Non-parametric Bayesian Methods in Machine Learning

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Outline

- (My) Bayesian philosophy
- Gaussian Processes for Regression and Classification (Monday)
  - GP preliminaries
  - *Application 1*: typing on touch-screens
  - Classification (including semi-supervised)
  - *Application 2*: clinical (dis)-agreement
- Dirichlet Process flavoured Cluster Models (Tuesday)
  - DP preliminaries
  - *Application 3*:Idenfitying metabolites
  - *Application 4*:Cluster models for multiple data views
- Summary

Relevant publications

- The four applications are described in the following papers:
  - Uncertain Text Entry on Mobile Devices Weir et. al, CHI 2014
  - Investigating the Disagreement Between Clinicians' Ratings of Patients in ICUs Rogers et. al 2013, IEEE Trans Biomed Health Inform
  - MetAssign: Probabilistic annotation of metabolites from LC–MS data using a Bayesian clustering approach Daly et. al, Bioinformatics, under review
  - Infinite factorization of multiple non-parametric views Rogers et. al, Machine Learning 2009

About me

- I'm not a statistican by training (don't ask me to prove anything!).
- Education:
  - Undergraduate Degree: Electrical and Electronic Engineering (Bristol)
  - PhD: Machine Learning Techniques for Microarray Analysis (Bristol)
- Currently:
  - Lecturer: Computing Science
  - Research Interests: Machine Learning and Applied Statistics in Computational Biology and Human-Computer Interaction (HCI)

Lecture 1: Bayesian Inference

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Bayesian Inference

Standard setup:

- We have some data $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- We have a model $p(\mathbf{X}|\mathbf{\Theta})$
- We define a prior $p(\mathbf{\Theta})$

Bayesian Inference

Standard setup:

- We have some data $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- We have a model $p(\mathbf{X}|\mathbf{\Theta})$
- We define a prior $p(\mathbf{\Theta})$
- We use Bayes rule (and typically lots of computation) to compute (or estimate) the posterior:

$$p(\mathbf{\Theta}|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{\Theta})p(\mathbf{\Theta})}{p(\mathbf{X})}$$

Why Be Bayesian?

Why Be Bayesian?

- Ability to incoroporate prior information?

Why Be Bayesian?

- Ability to incoroporate prior information?
- Ability to compute posterior densities (combine prior with likelihood)?

Why Be Bayesian?

- Ability to incoroporate prior information?
- Ability to compute posterior densities (combine prior with likelihood)?
- Ability to compare models via marginal likelihood?

Why Be Bayesian?

- Ability to incoroporate prior information?
- Ability to compute posterior densities (combine prior with likelihood)?
- Ability to compare models via marginal likelihood?
- For me: the ability to integrate out model parameters completely...

Why be Bayesian?

- We're often not interested in parameter values
- We're normally interested in something that is a function of the parameter values e.g.:
  - Within Machine Learning (ML) we are often interested in making predictions (predicing y_* from $\mathbf{x}_*$).
  - This will often require values of some parameters $\mathbf{\Theta}$
  - Being Bayesian allows us to *average* over uncertainity in parameters when making predictions:

$$p(y_*|\mathbf{x}_*, \mathbf{X}) = \int p(y_*|\mathbf{x}_*, \mathbf{\Theta}) p(\mathbf{\Theta}|\mathbf{X}) \, d\mathbf{\Theta}$$

- This for me, is the biggest Bayesian selling point!

Lecture 2: Gaussian Process Basics

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Gaussian Processes

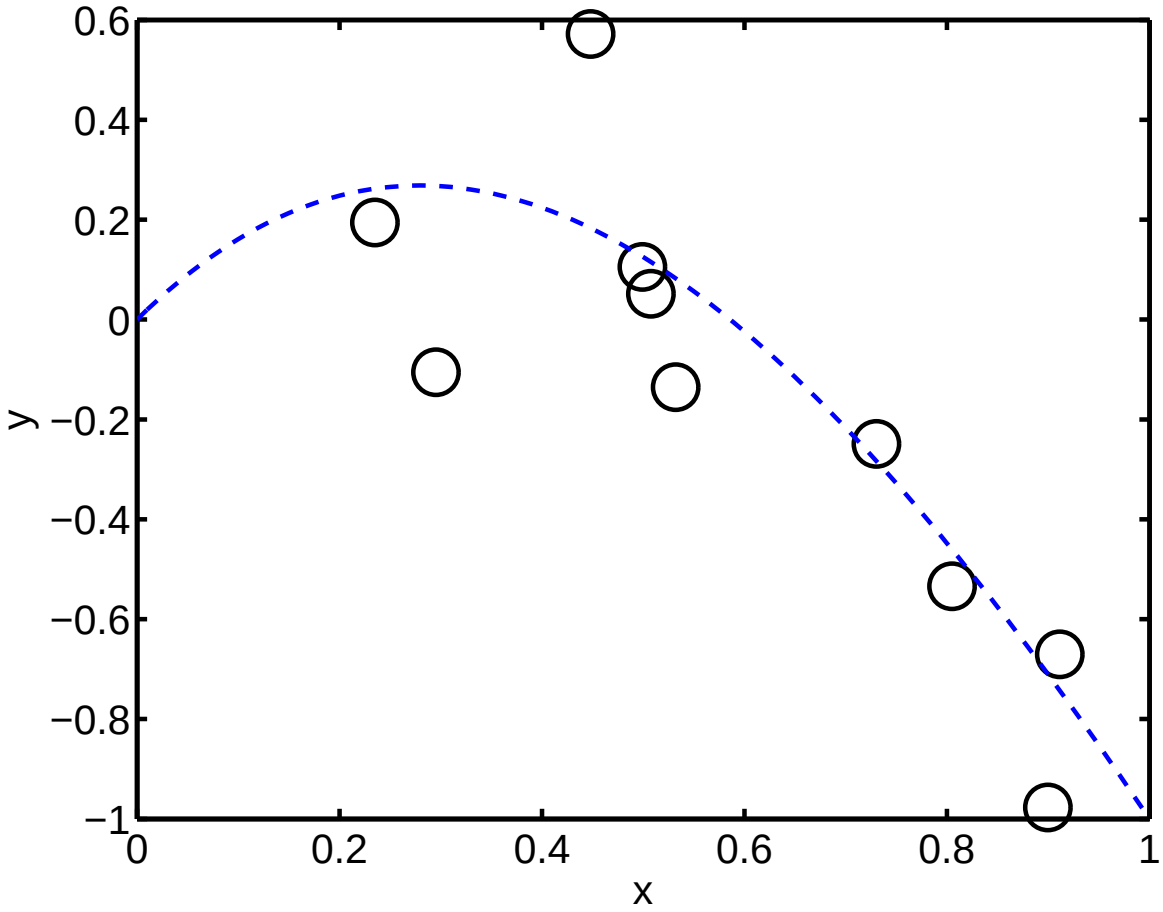


Figure 1 : A familiar problem: learn the underlying function (blue) from the observed data (crosses).

A parametric approach?

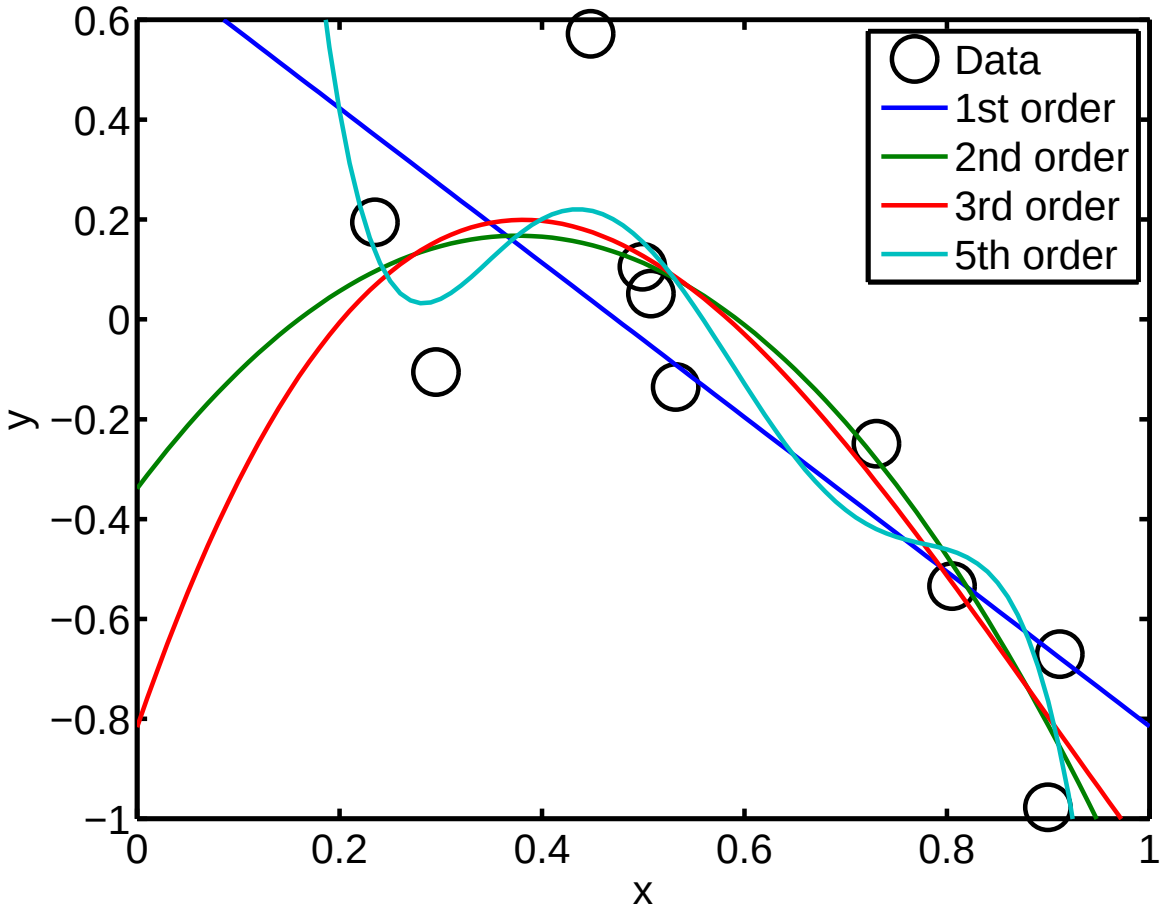


Figure 2 : Polynomials fitted by least squares.

It's easy to under and over-fit. What if we have no idea of the parametric form of the function?

A non-parametric approach - Gaussian Processes

- Rather than forcing us to choose a particular parametric form, a Gaussian Process (GP) allows us to place a prior distribution directly on *functions*
- With a GP prior we can:
  - Sample functions from the prior
  - Incorporate data to get a *posterior* distribution over functions
  - Make predictions

Some formalities

- We observe N training points, each of which consists of a set of features $\mathbf{x}_n$ and a target y_n.
- We can stack all of the y_n into a vector and $\mathbf{x}_n$ into a matrix:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}$$

GP definition

- The GP assumes that the vector of *all possible* y_n is a draw from a Multi-Variate Gaussian (MVG).
- We don't observe *all possible* values (if we did, we wouldn't need to make predictions!)

GP definition

- The GP assumes that the vector of *all possible* y_n is a draw from a MVG.
- We don't observe *all possible* values (if we did, we wouldn't need to make predictions!)
- But the marginal densities of a MVG are also MVGs so the subset we observe are also a draw from a MVG.

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$$

- i.e. if we have N training points, we're dealing with an N-dimensional MVG

GP definition

- The GP assumes that the vector of *all possible* y_n is a draw from a MVG.
- We don't observe *all possible* values (if we did, we wouldn't need to make predictions!)
- But the marginal densities of a MVG are also MVGs so the subset we observe are also a draw from a MVG.

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$$

- i.e. if we have N training points, we're dealing with an N-dimensional MVG
- With mean $\boldsymbol{\mu}$ (normally 0) and covariance $\mathbf{C}$

GP definition

- The GP assumes that the vector of *all possible* y_n is a draw from a MVG.
- We don't observe *all possible* values (if we did, we wouldn't need to make predictions!)
- But the marginal densities of a MVG are also MVGs so the subset we observe are also a draw from a MVG.

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$$

- i.e. if we have N training points, we're dealing with an N-dimensional MVG
- With mean $\boldsymbol{\mu}$ (normally 0) and covariance $\mathbf{C}$
- $\mathbf{x}_n$ looks to have disappeared – we find it inside $\mathbf{C}$

GP definition

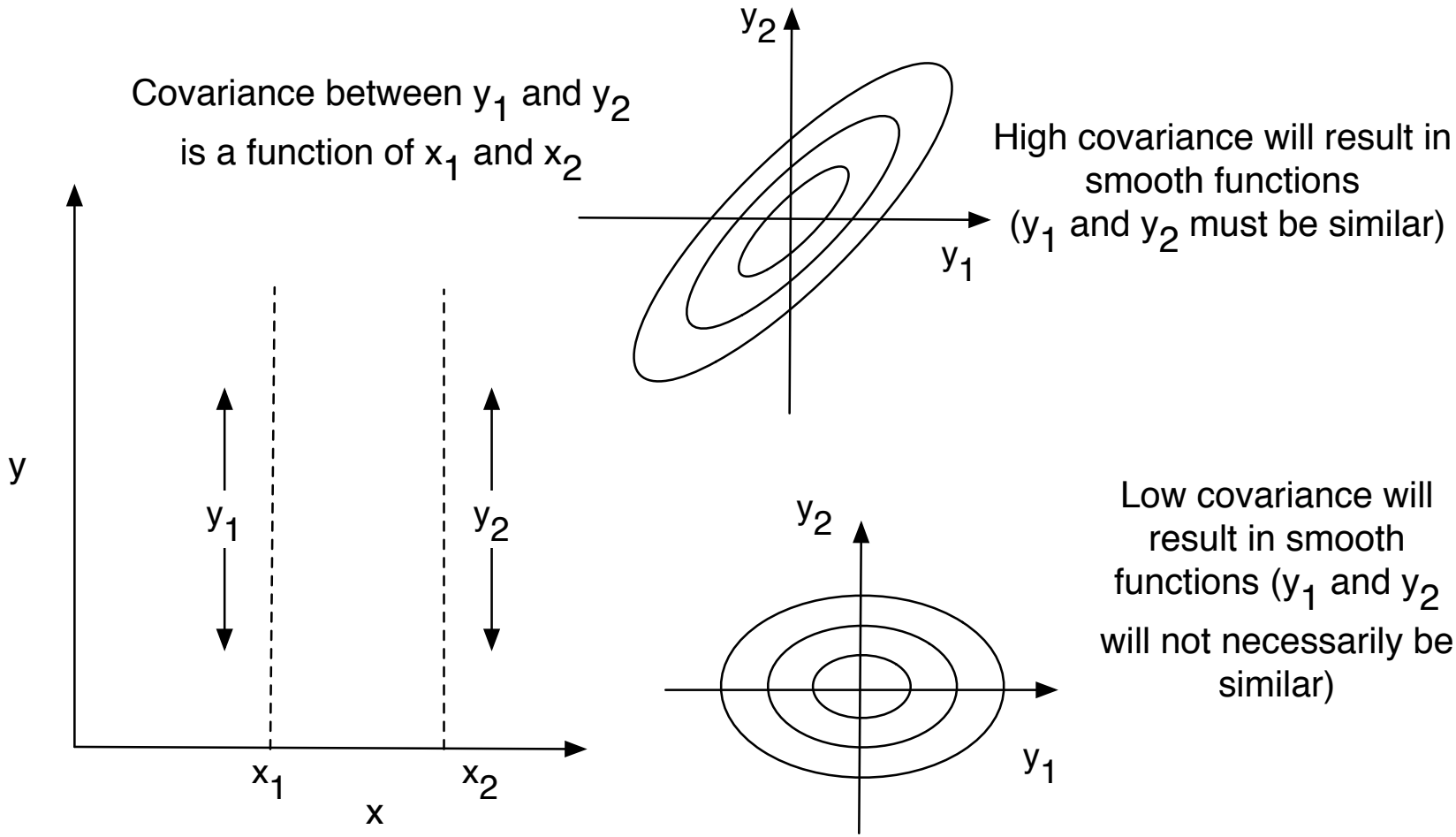


Figure 3 : Schematic of GP prior for two function values.

Covariance functions

- By choosing a covariance function, we are making an assumption on the *smoothness* of the regression function.
- Common choices:
  - Linear: $C(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2$
  - RBF: $C(\mathbf{x}_1, \mathbf{x}_2) = \exp\left\{-0.5\gamma||\mathbf{x}_1 - \mathbf{x}_2||^2\right\}$
  - And many, many more.
- More details: http://www.gaussianprocess.org/gpml/
  - (Free) book
  - Code

GP posterior

- We assume that all possible y_n are distributed as MVG
- Any subset are also MVG
- If we condition on some observed values of y, the density of the others is still MVG
- Joint density of observed ($\mathbf{y}$) and un-observed ($\mathbf{y}_*$):

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

- And then use standard results for Gaussian conditionals:

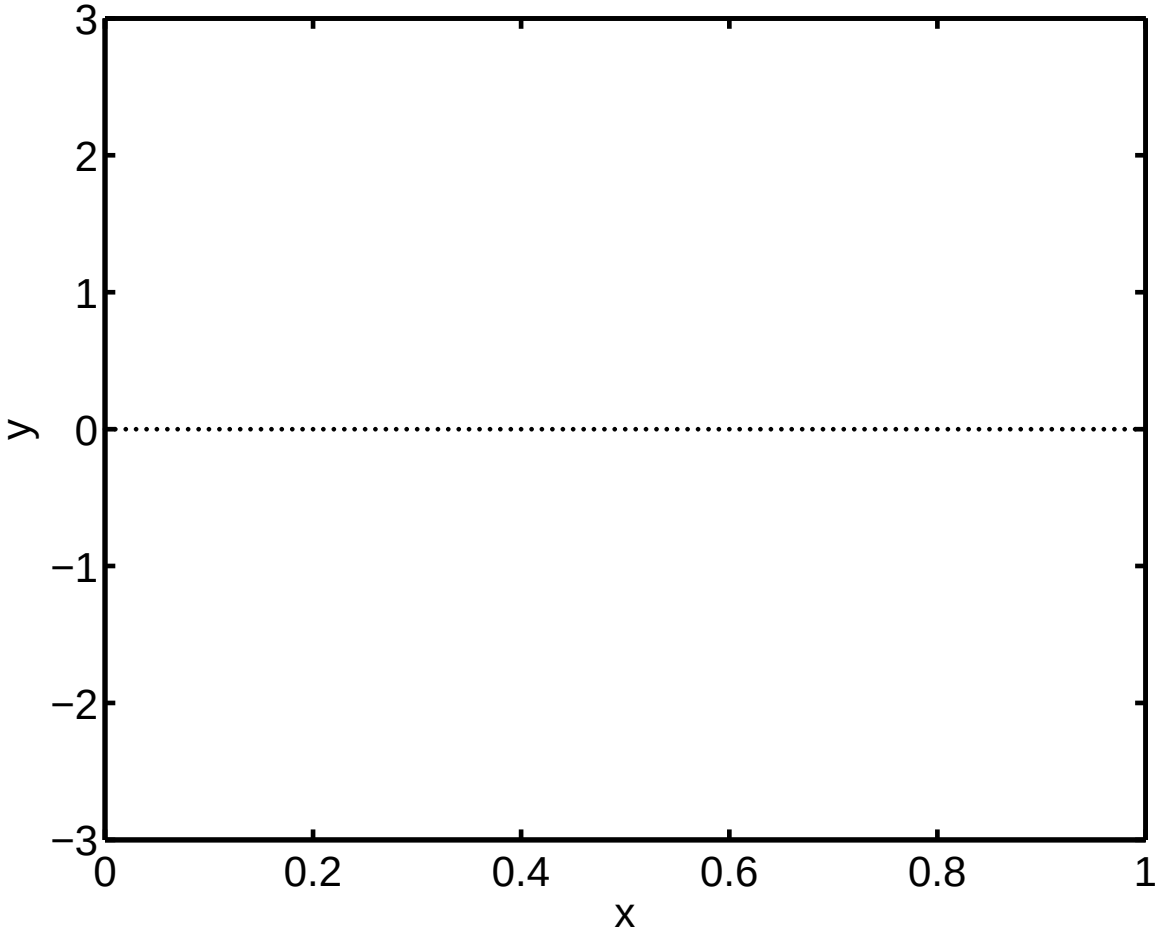
$$p(\mathbf{y}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*) = \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

where

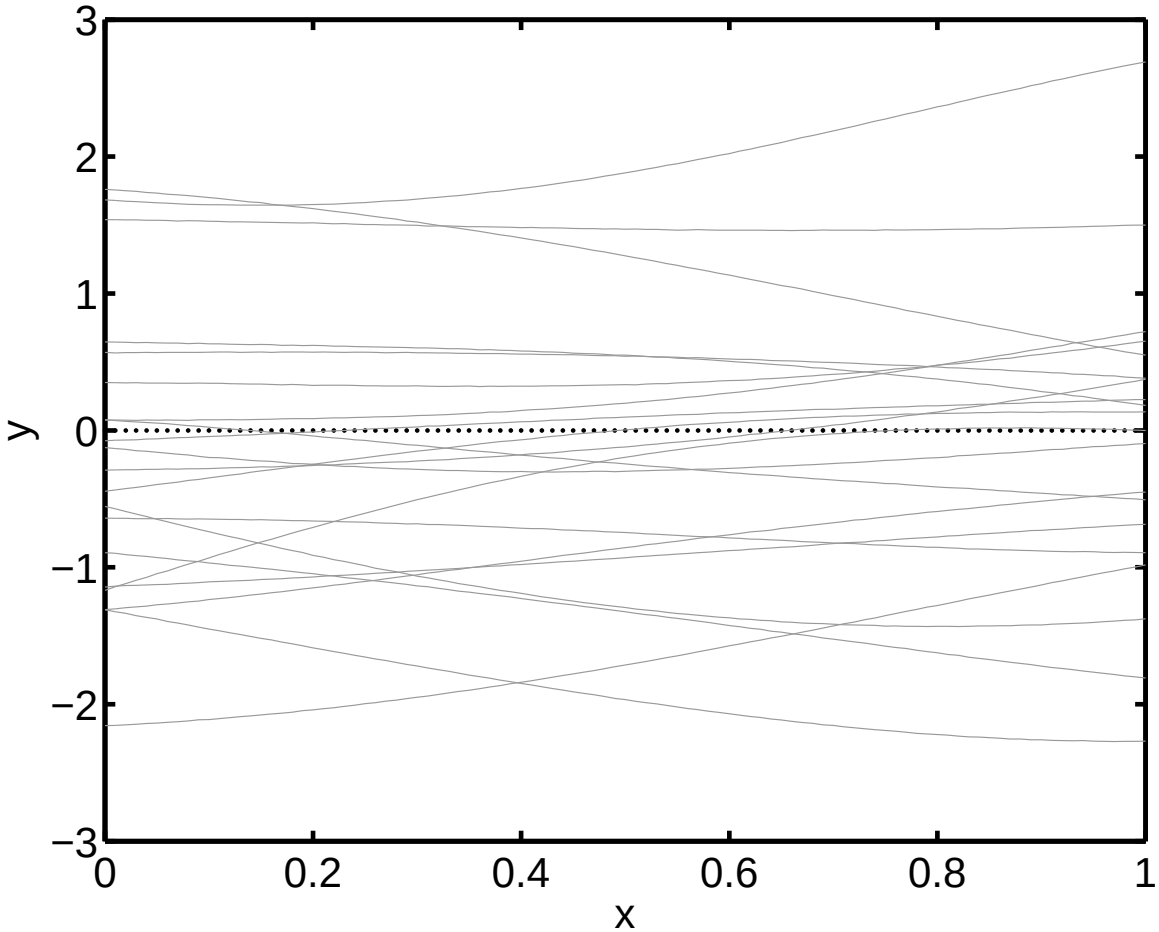
$$\begin{aligned} \boldsymbol{\mu}_* &= C(\mathbf{X}_*, \mathbf{X})C(\mathbf{X}, \mathbf{X})^{-1}\mathbf{y} \\ \sigma_*^2 &= C(\mathbf{X}_*, \mathbf{X}_*) - C(\mathbf{X}_*, \mathbf{X})C(\mathbf{X}, \mathbf{X})^{-1}C(\mathbf{X}, \mathbf{X}^*) \end{aligned}$$

http://orion.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf

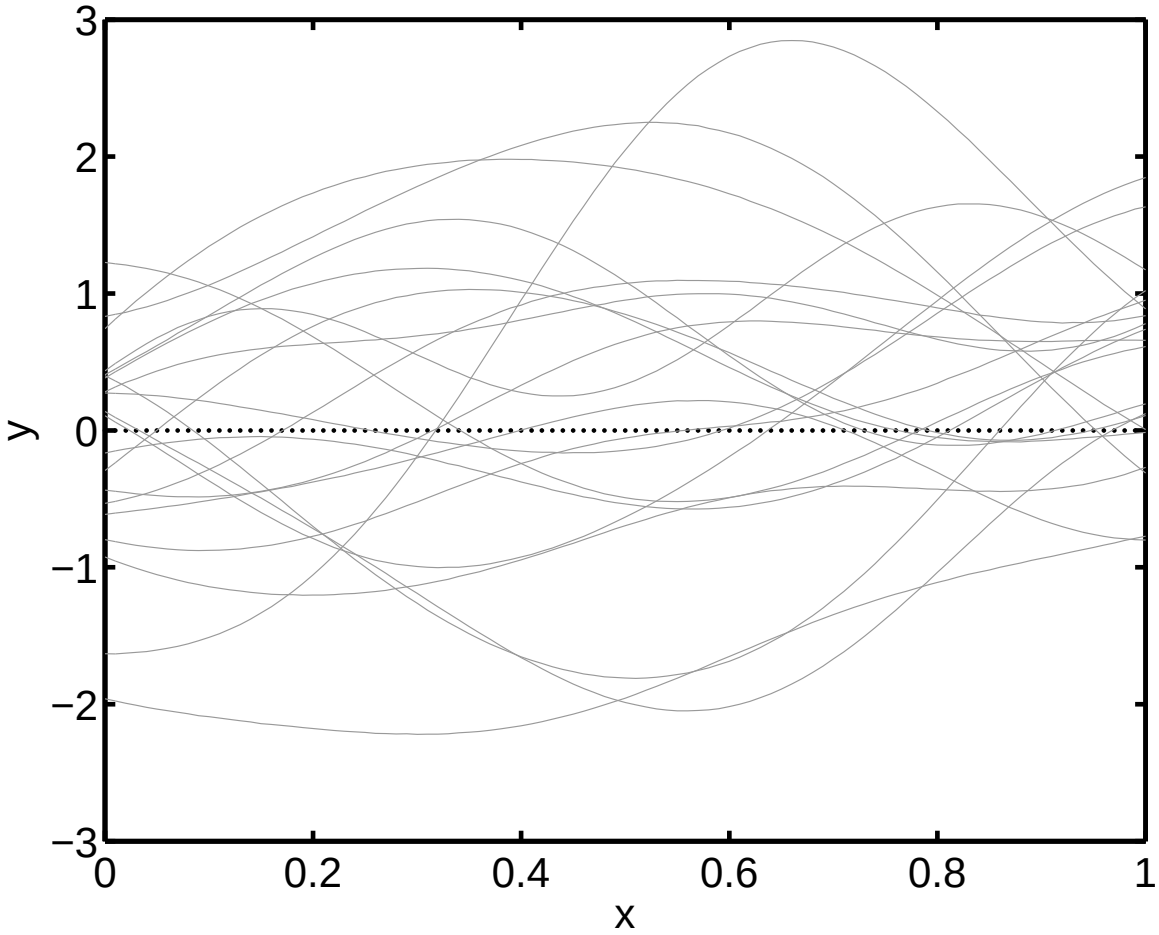
GP example



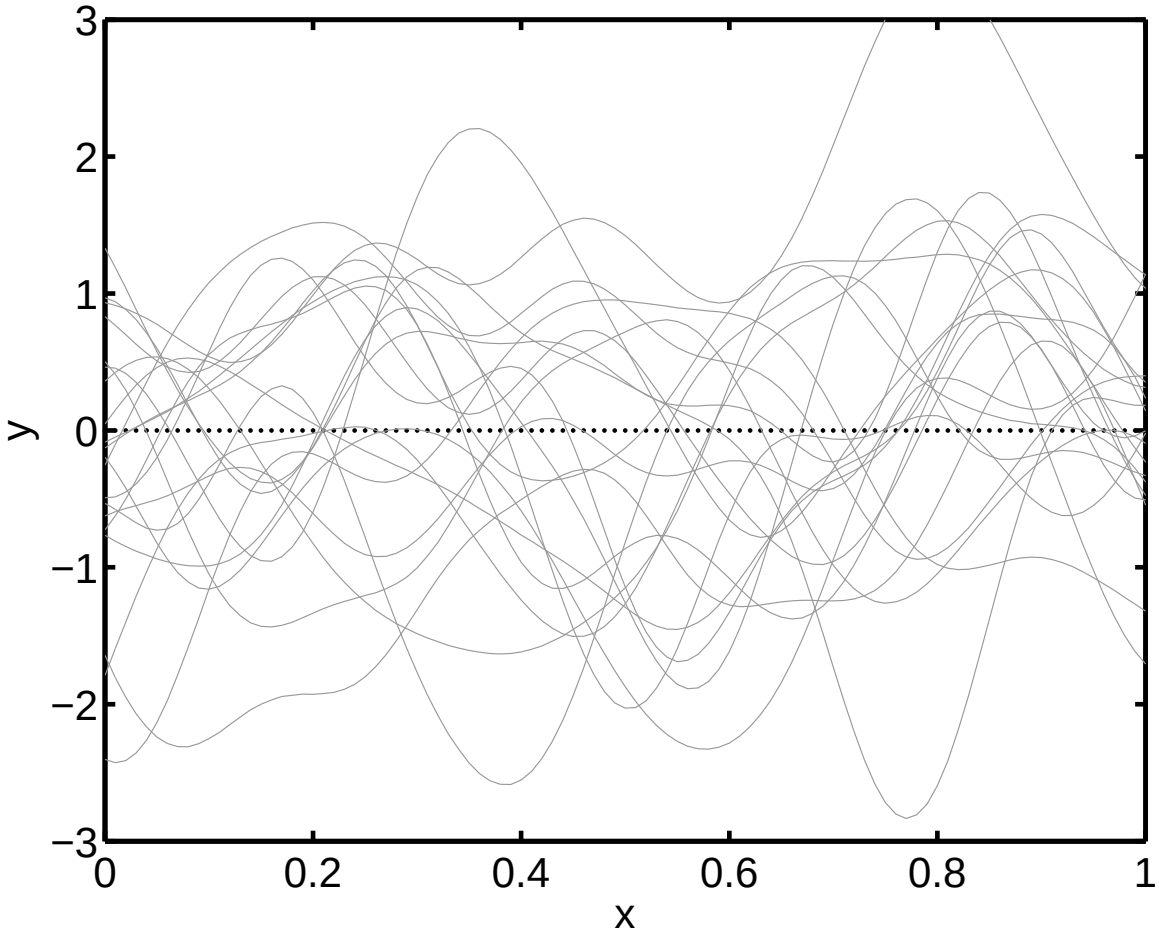
- Start by defining which points we're *interested* in



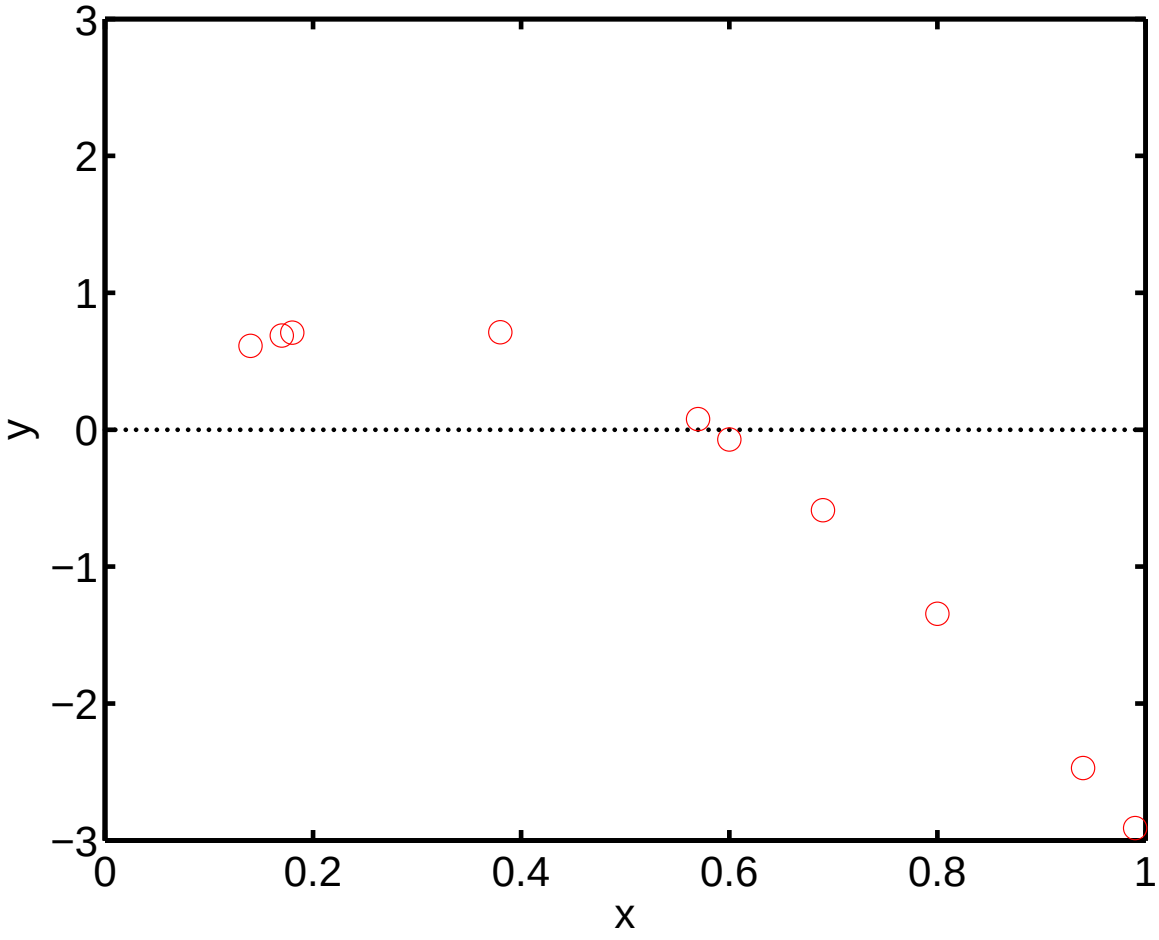
- We can sample function values at these points from the prior
- As the hyper-parameter changes, the smoothness changes
- Sometimes it's best to draw a continuous line



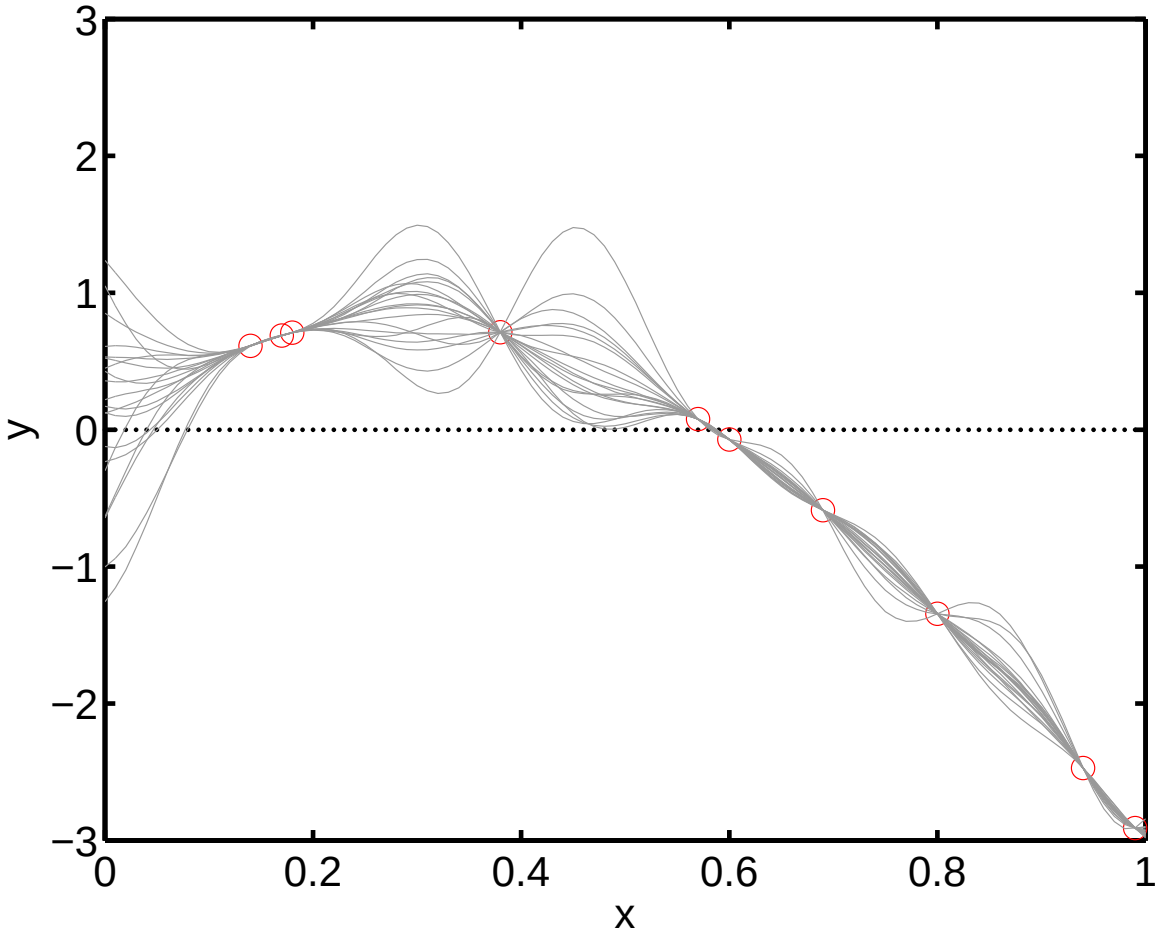
- We can sample function values at these points from the prior
- As the hyper-parameter changes, the smoothness changes
- Sometimes it's best to draw a continuous line



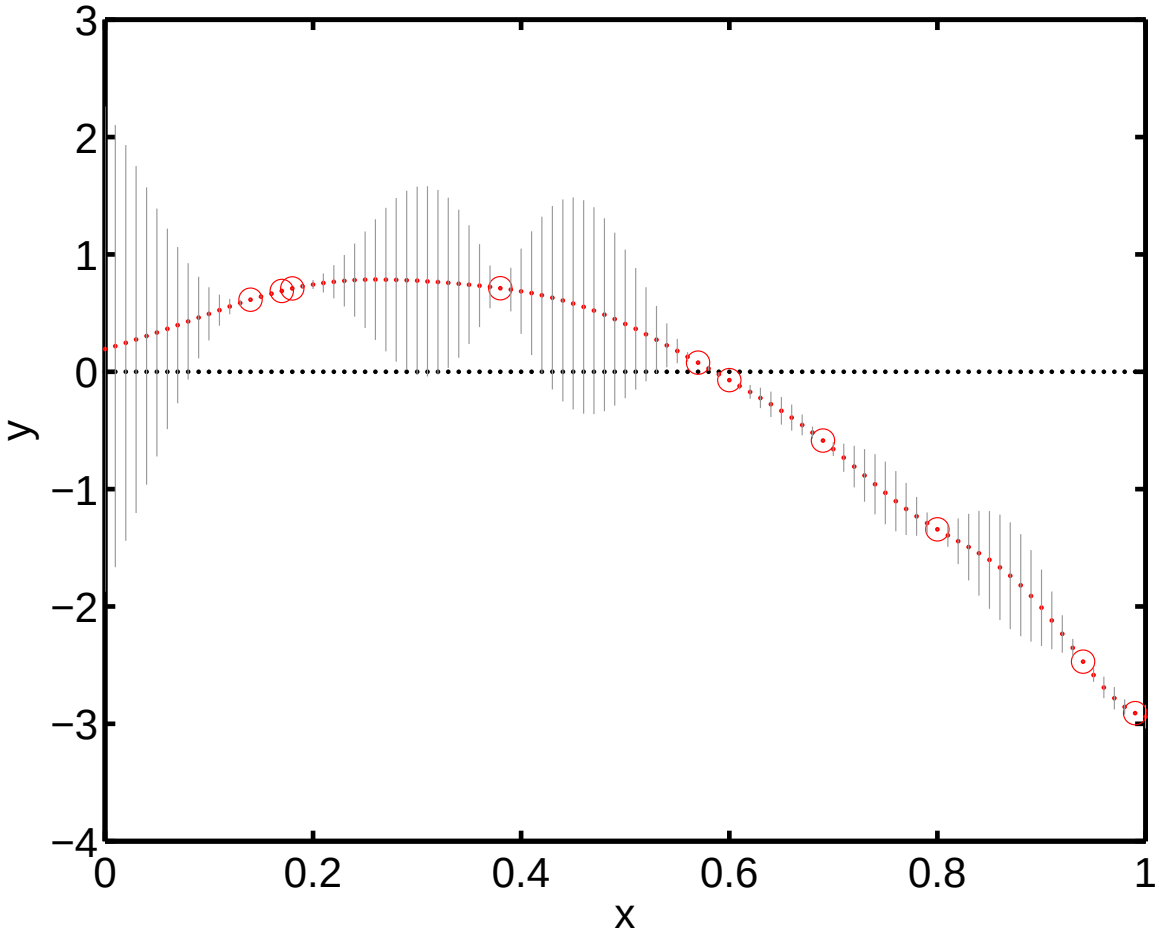
- We can sample function values at these points from the prior
- As the hyper-parameter changes, the smoothness changes
- Sometimes it's best to draw a continuous line



- Observe values of y at some of the points



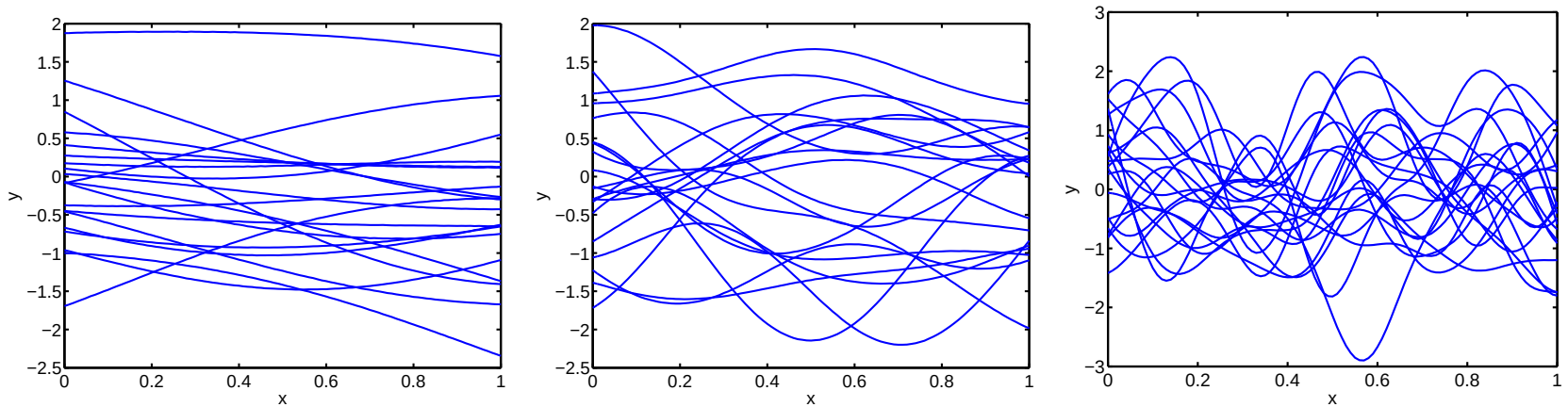
- Sample from the posterior, conditioned on the observed data



- Marginal predictive mean plus/minus 3 SD

Hyper-parameters

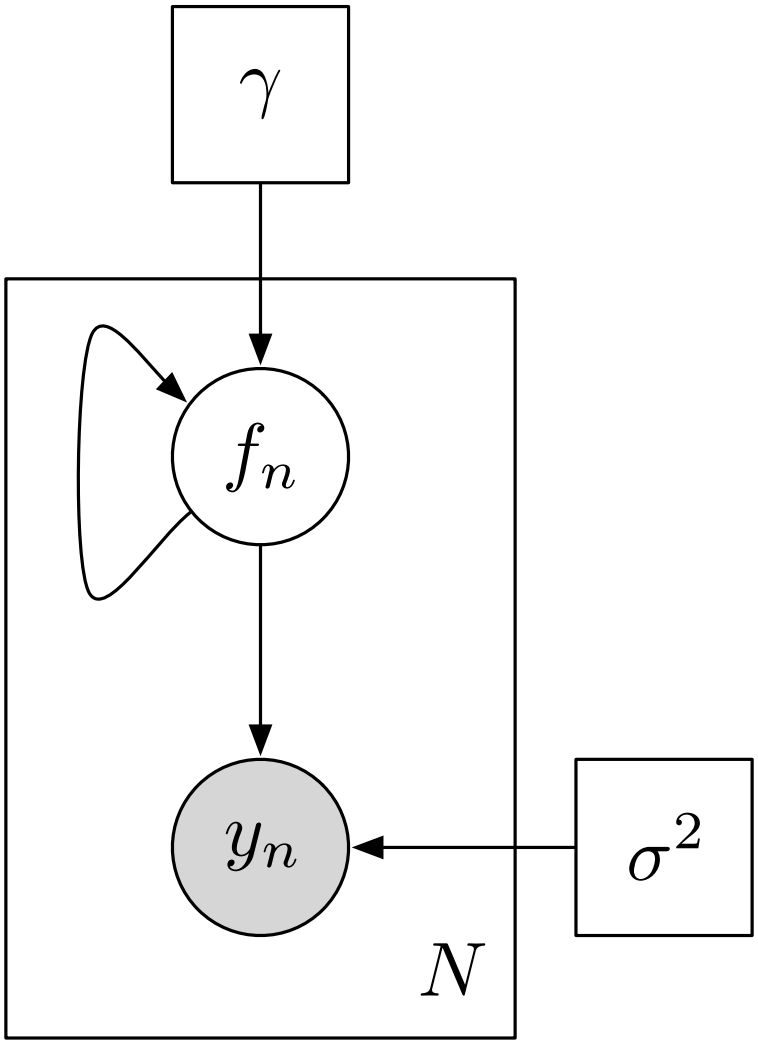
$$C(\mathbf{x}_1, \mathbf{x}_2) = \exp\left\{-0.5\gamma||\mathbf{x}_1 - \mathbf{x}_2||^2\right\}$$



(a) $\gamma = 1$ (b) $\gamma = 10$ (c) $\gamma = 100$

Figure 4 : Varying hyper-parameters in an RBF covariance varies the smoothness of the function.

Adding observation noise

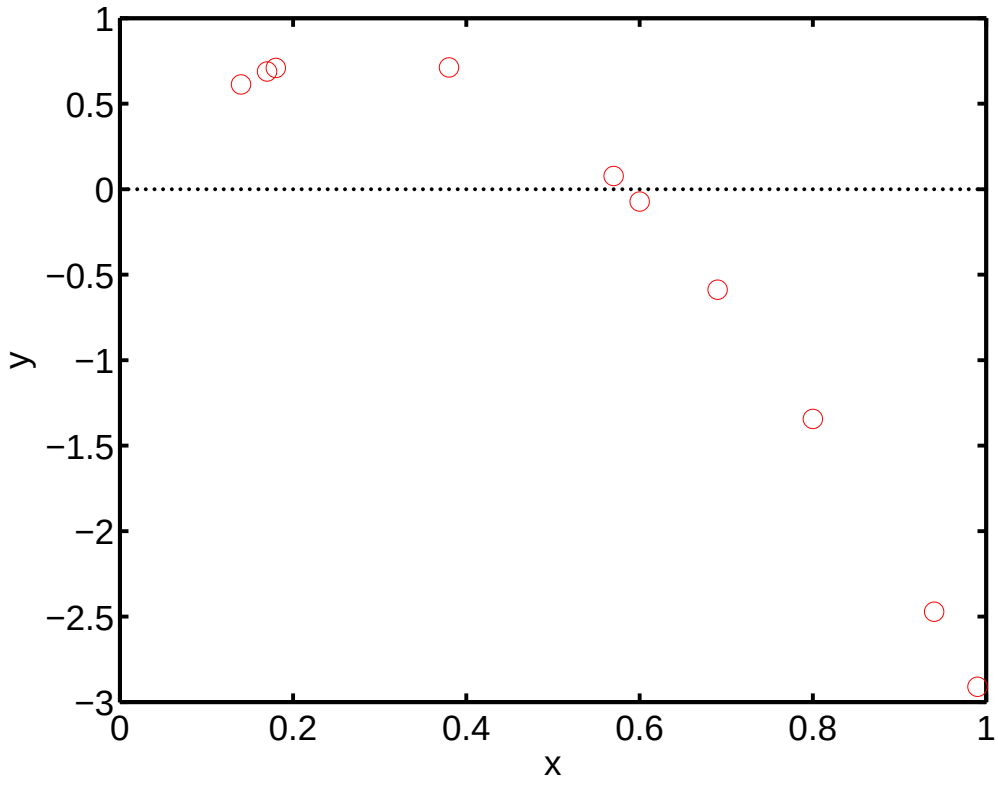


$$
\begin{aligned}
\mathbf{f} &\sim \mathcal{N}(\mathbf{0}, \mathbf{C}) \\
y_n &\sim \mathcal{N}(f_n, \sigma^2) \\
\mathbf{y} &\sim \mathcal{N}(\mathbf{0}, \mathbf{C} + \sigma^2 \mathbf{I})
\end{aligned}
$$

Observation noise

- Posterior of can be derived as:

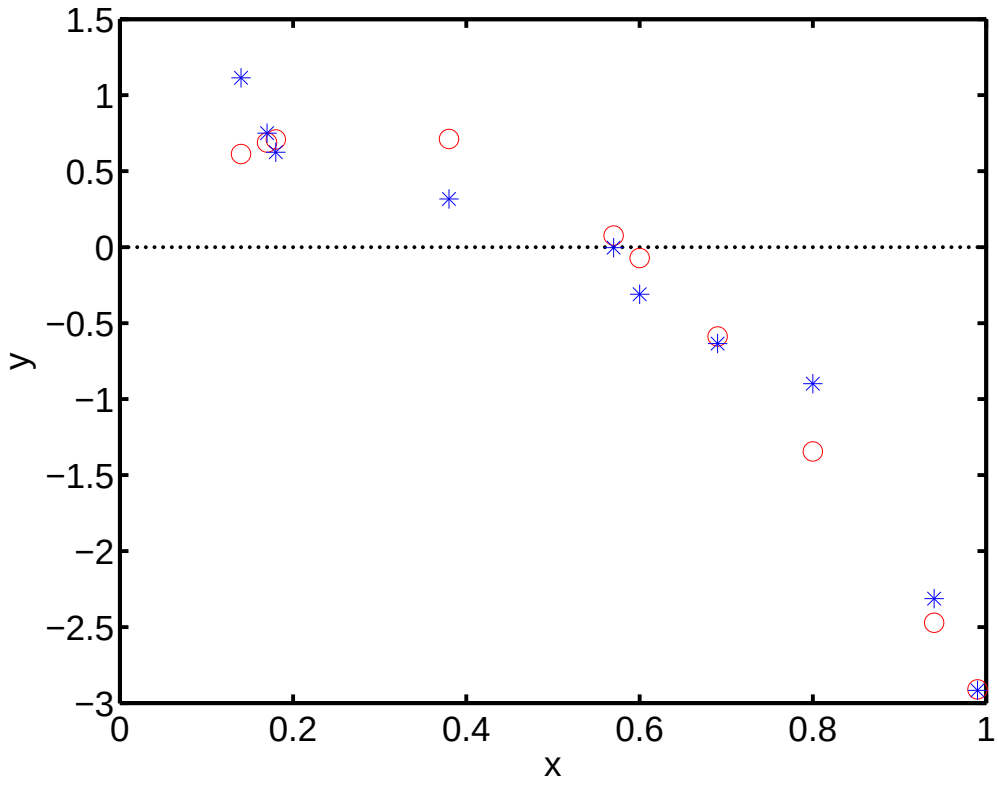
$$
\begin{aligned}
p(\mathbf{f}|\mathbf{y}) &\propto \mathcal{N}(\mathbf{f}|(0), \mathbf{C})\mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}) \\
&= \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f) \\
\boldsymbol{\Sigma}_f &= (\mathbf{C}^{-1} + \sigma^{-2}\mathbf{I})^{-1} \\
\boldsymbol{\mu}_f &= \sigma^{-2}\Sigma_f\mathbf{y}
\end{aligned}
$$



Observation noise

- Posterior of can be derived as:

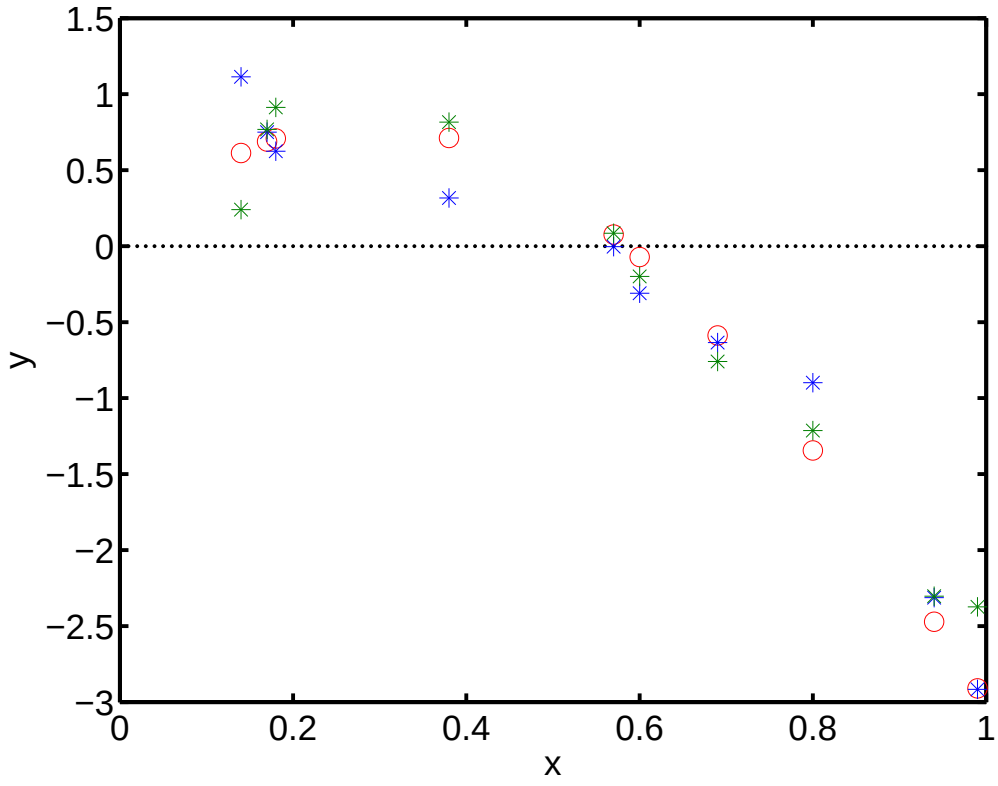
$$
\begin{aligned}
p(\mathbf{f}|\mathbf{y}) &\propto \mathcal{N}(\mathbf{f}|(0), \mathbf{C})\mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}) \\
&= \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f) \\
\boldsymbol{\Sigma}_f &= (\mathbf{C}^{-1} + \sigma^{-2}\mathbf{I})^{-1} \\
\boldsymbol{\mu}_f &= \sigma^{-2}\Sigma_f\mathbf{y}
\end{aligned}
$$



Observation noise

- Posterior of can be derived as:

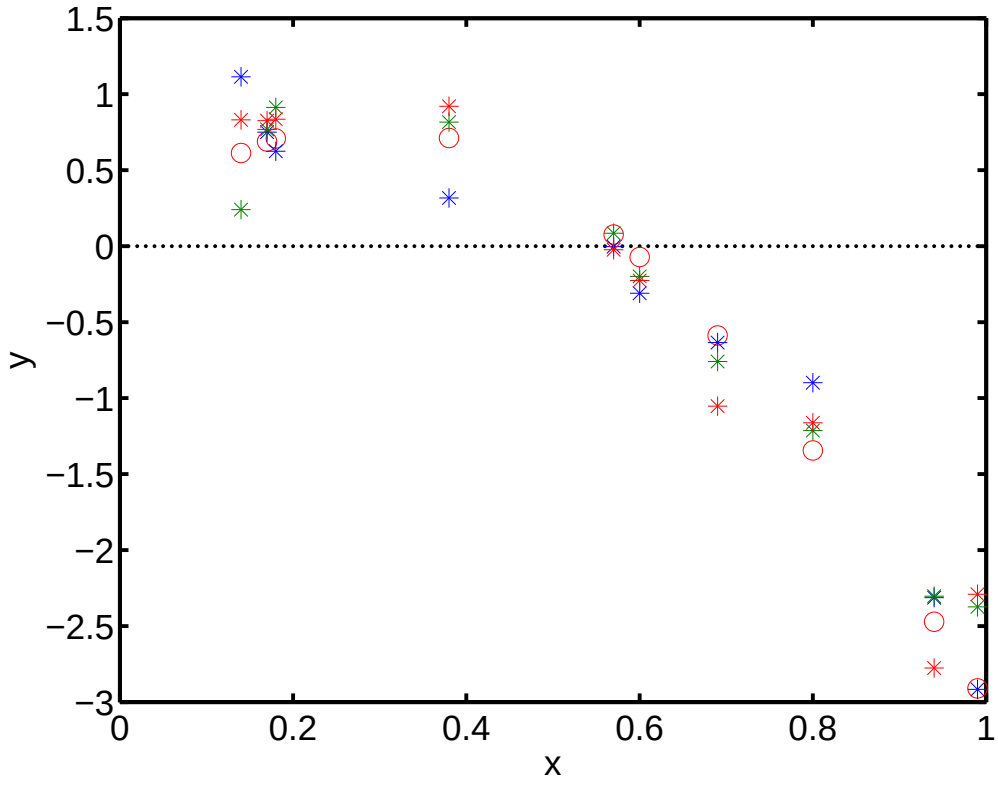
$$
\begin{aligned}
p(\mathbf{f}|\mathbf{y}) &\propto \mathcal{N}(\mathbf{f}|(0), \mathbf{C})\mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}) \\
&= \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f) \\
\boldsymbol{\Sigma}_f &= (\mathbf{C}^{-1} + \sigma^{-2}\mathbf{I})^{-1} \\
\boldsymbol{\mu}_f &= \sigma^{-2}\Sigma_f\mathbf{y}
\end{aligned}
$$



Observation noise

- Posterior of can be derived as:

$$
\begin{aligned}
p(\mathbf{f}|\mathbf{y}) &\propto \mathcal{N}(\mathbf{f}|(0), \mathbf{C})\mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}) \\
&= \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f) \\
\boldsymbol{\Sigma}_f &= (\mathbf{C}^{-1} + \sigma^{-2}\mathbf{I})^{-1} \\
\boldsymbol{\mu}_f &= \sigma^{-2}\Sigma_f\mathbf{y}
\end{aligned}
$$



Predictions with observation noise

- As before, first construct the joint model (note, predicting $\mathbf{f}_*$)

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} C(\mathbf{X},\mathbf{X}) + \sigma^2\mathbf{I} & C(\mathbf{X},\mathbf{X}_*) \\ C(\mathbf{X}_*,\mathbf{X}) & C(\mathbf{X}_*,\mathbf{X}_*) \end{bmatrix}\right)$$

Predictions with observation noise

- As before, first construct the joint model (note, predicting $\mathbf{f}_*$)

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} C(\mathbf{X},\mathbf{X}) + \sigma^2\mathbf{I} & C(\mathbf{X},\mathbf{X}_*) \\ C(\mathbf{X}_*,\mathbf{X}) & C(\mathbf{X}_*,\mathbf{X}_*) \end{bmatrix}\right)$$

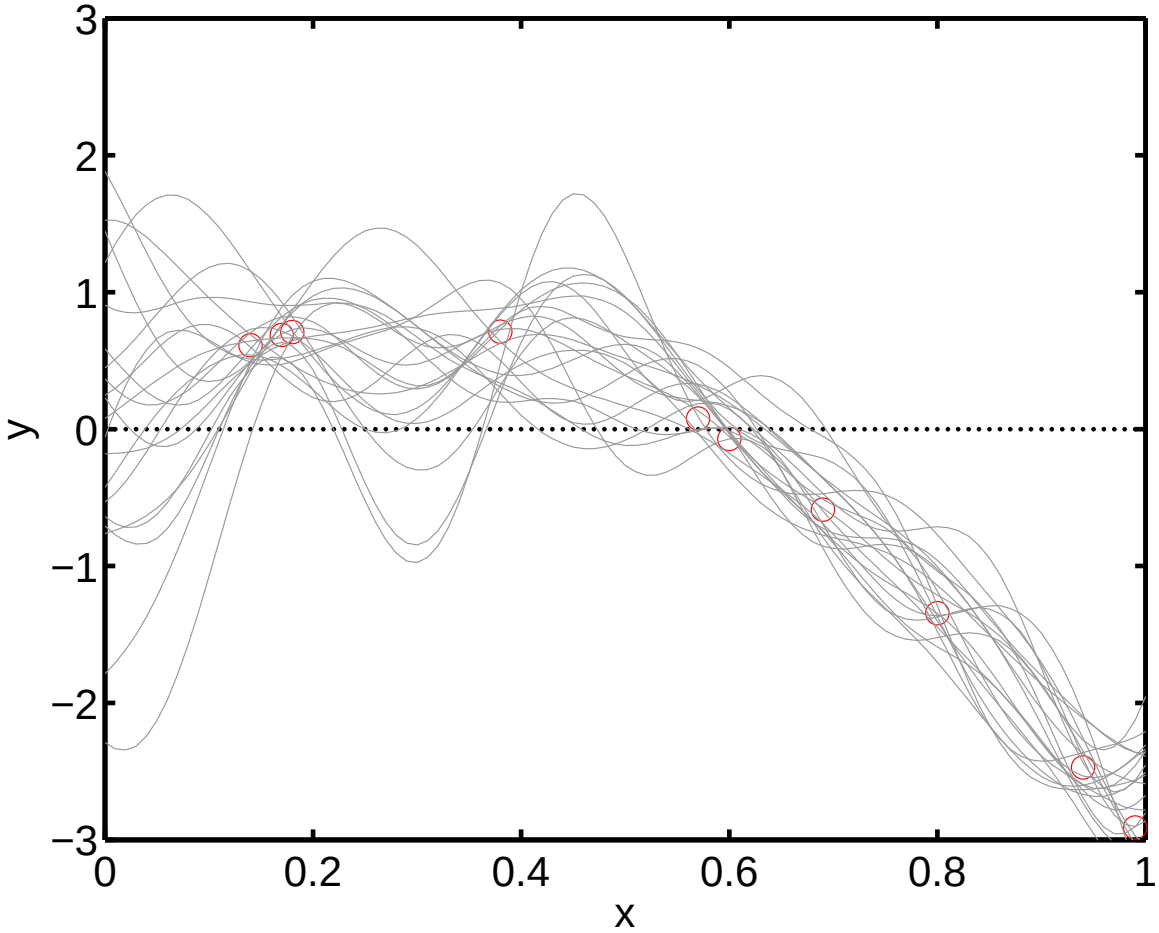
- And then compute the conditionals:

$$p(\mathbf{f}_*|\mathbf{y},\mathbf{X},\mathbf{X}_*,\sigma^2) = \mathcal{N}(\boldsymbol{\mu}_*,\boldsymbol{\Sigma}_*)$$

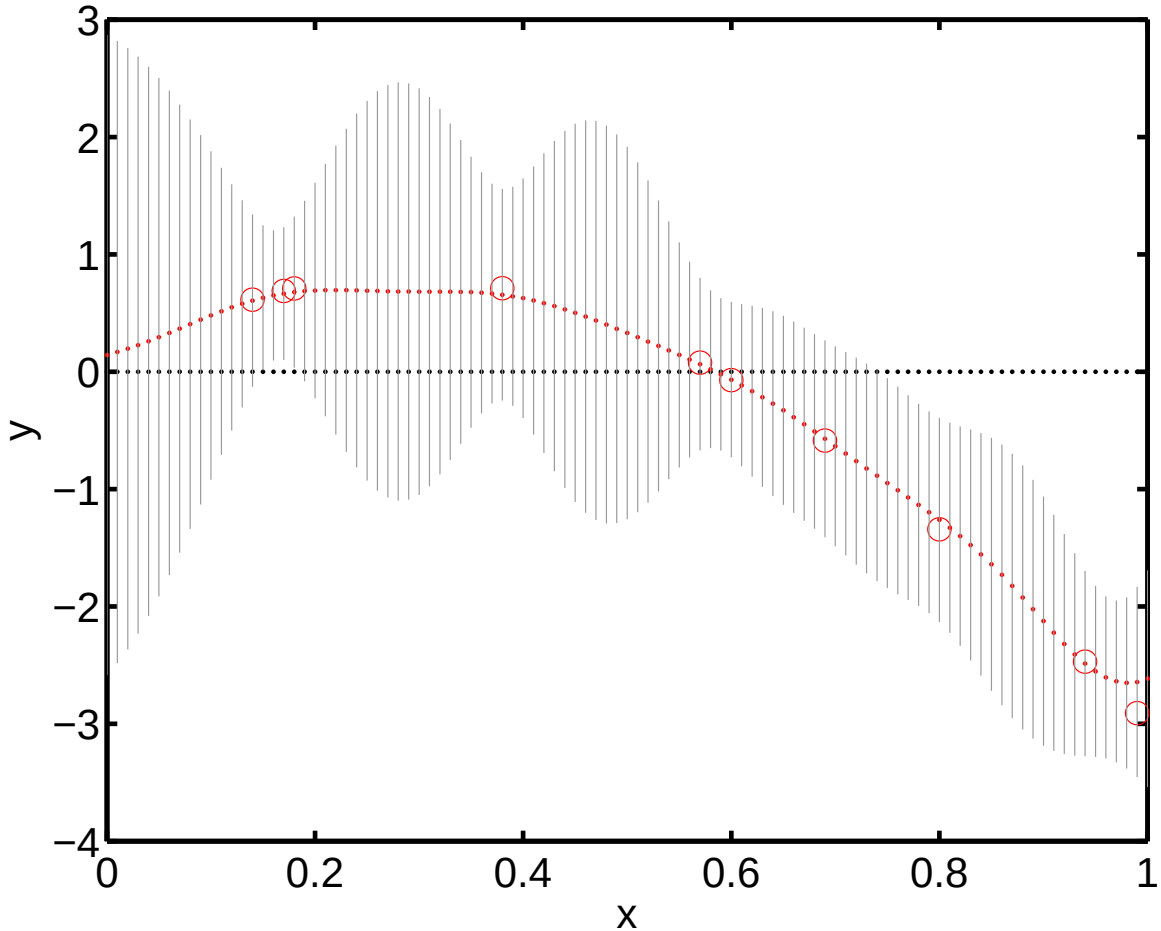
where

$$\begin{aligned} \boldsymbol{\mu}_* &= C(\mathbf{X}_*,\mathbf{X})\left[C(\mathbf{X},\mathbf{X}) + \sigma^2\mathbf{I}\right]^{-1}\mathbf{y} \\ \boldsymbol{\Sigma}_* &= C(\mathbf{X}_*,\mathbf{X}_*) - C(\mathbf{X}_*,\mathbf{X})\left[C(\mathbf{X},\mathbf{X}) + \sigma^2\mathbf{I}\right]^{-1}K(\mathbf{X},\mathbf{X}^*) \end{aligned}$$

Predictive samples with observation noise



- Posterior functions no longer have to go through training data



- Marginal predictive mean plus / minus 3 sd.

GP Exercise

TASK [1]

- Experiment with the `gp_task.m` script (in Matlab or Octave)
- Try:
  - Generating data from different functions
  - Using different covariance/noise parameters
  - Using different covariance functions (there are lots in `kernel.m`)
- You'll also need `gausssamp.m` and `kernel.m`

Optimising hyper-parameters

- Can use standard methods (e.g. Cross-validation)
- Or maximise the marginal likelihood

$$\underset{\sigma^2,\gamma}{\text{argmax}} \quad p(\mathbf{y}|\mathbf{X})$$

- $p(\mathbf{y}|\mathbf{X})$ available analytically (by marginalising):

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^T(\mathbf{C}+\sigma^2\mathbf{I})^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{C}+\sigma^2\mathbf{I}| - \frac{N}{2}\log 2\pi$$

- Most implementations (e.g. gpml) provide marginal likelihood maximisation routines

Summary

- GPs offer a flexible alternative to parametric regression
- If observation noise is Gaussian, all required densities are available analytically

Lecture 3: Application: Touchscreen typing

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Typing on touchscreens

- Most people have smartphones
- Most smartphones have touchscreens
- Touchscreens are small
- Keyboards on touchscreens are small
- Typing on them is hard!
  - . . . but people type on them a lot

Background 1: Why is it hard?

- Occlusion of target by finger
- 'fat finger' problem
- Small targets
- Demo: http://bit.ly/1nBws97

Background 1: Why is it hard?

- Occlusion of target by finger
- 'fat finger' problem
- Small targets
- Demo: http://bit.ly/1nBws97
- Quite a bit of work in this area:
  - Holz and Baudisch
  - Henze (100,000,000 taps)
- Collecting data is fairly easy

Background 2: All users are different

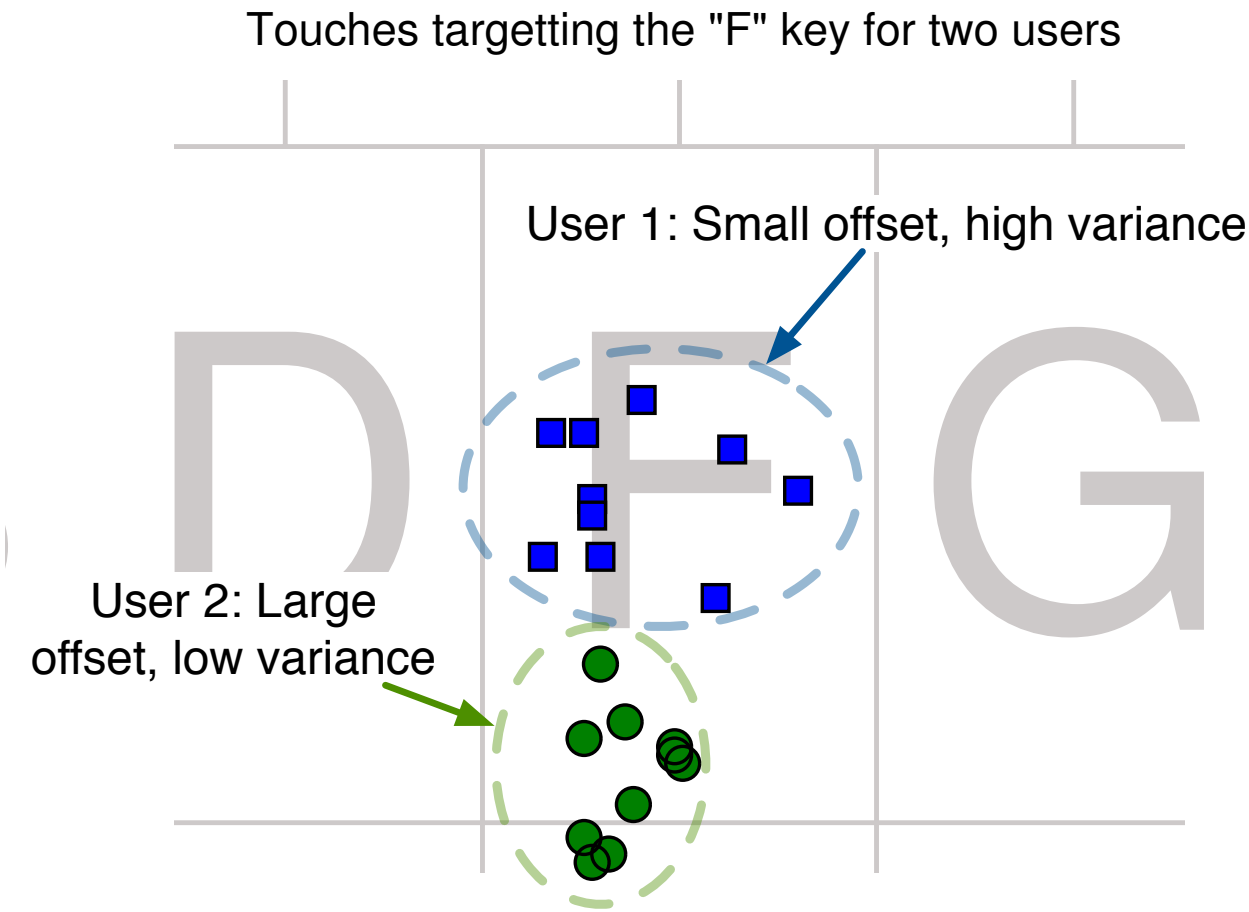


Figure 5 : Touches recorded by two users aiming for the 'F' key. User 2 has high bias and low variance, user 1 has low bias and high variance.

Background 3: Current systems (maybe?)

- Touch is boxed into nearest key.
- Key ID is passed to a Statistical Language Model (SLM).
- SLM is made up of probabilities of observing certain character strings (from large text corpora).
- SLM can swap characters to make the character string more likely.
  - e.g. 'HELLP $\rightarrow$ HELLO'

Our idea

- There is a lot of uncertainty present in touch (bias and variance)
- Boxing a touch into a key is probably bad
- Why can't we pass a *distribution* to the SLM?
  - Pass the uncertainty onwards
  - Being Bayesian!

Our idea

- There is a lot of uncertainty present in touch (bias and variance)
- Boxing a touch into a key is probably bad
- Why can't we pass a *distribution* to the SLM?
  - Pass the uncertainty onwards
  - Being Bayesian!
- Can use a user specific GP regression model to predict target from input touch.

The model

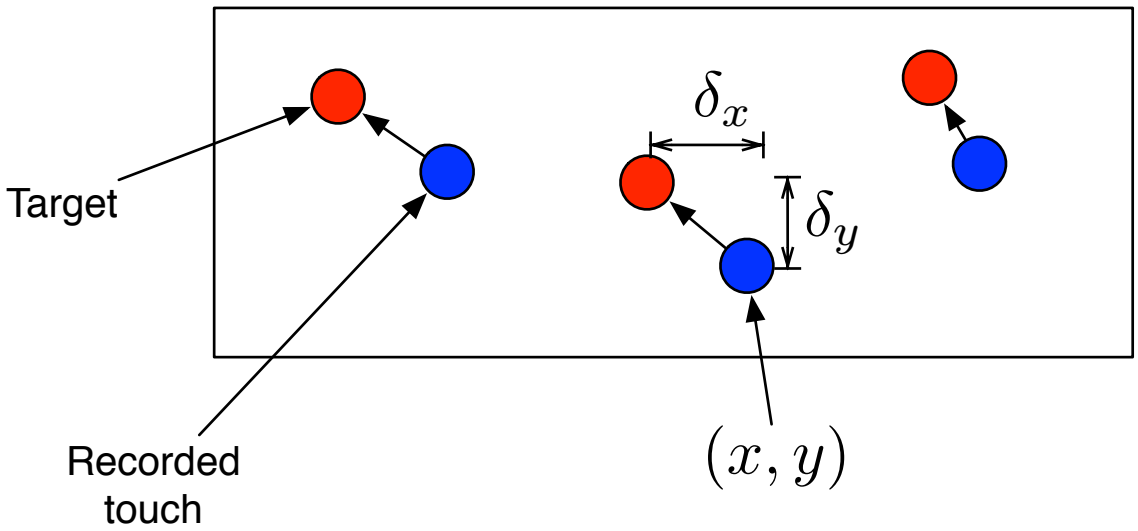
- We use independent GP regressions for predicting x and y offsets.
- Training data:
  - Each user typed phrases provided to them.
  - Data: the x, y location of the recorded touch (i.e. $\mathbf{x}_n = [x_n, y_n]^T$). Target: the center of the intended key minus the touch (i.e. the offset).

The model

- We use independent GP regressions for predicting x and y offsets.
- Training data:
  - Each user typed phrases provided to them.
  - Data: the x, y location of the recorded touch (i.e. $\mathbf{x}_n = [x_n, y_n]^T$). Target: the center of the intended key minus the touch (i.e. the offset).
- Used a GP with zero mean and a composite covariance:

$$C(\mathbf{x}_1, \mathbf{x}_2) = a\mathbf{x}_1^T\mathbf{x}_2 + (1-a)\exp\{-\gamma||\mathbf{x}_1 - \mathbf{x}_2||^2\}$$

The model



$$\boldsymbol{\delta}_x \sim GP(\mathbf{0}, \mathbf{C})$$
$$\boldsymbol{\delta}_y \sim GP(\mathbf{0}, \mathbf{C})$$
$$\mathbf{C}_{nm} = f((x_n, y_n), (x_m, y_m))$$

- x and y offsets both depend on x and y position of touch
- Have also used raw capacitive sensor data as input
  - Potentially better (more info) but only accessible on some devices

System cartoon

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

System cartoon

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

System cartoon

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

System cartoon

Figure 6 : Train GPs to predict the intended touch from an input touch. The flexibility of GPs means that the mean and covariance of the offset can vary across the keyboard.

System cartoon

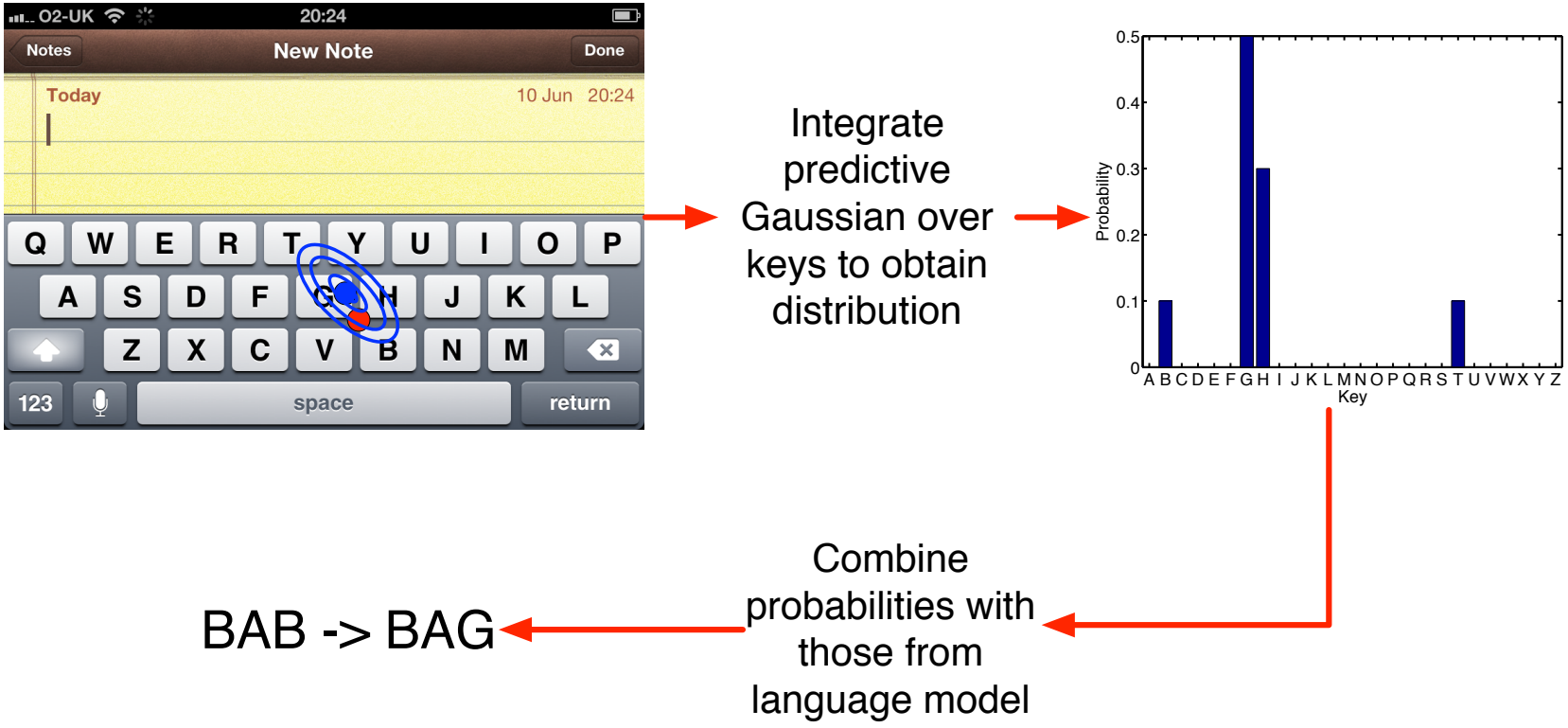


Figure 7 : The complete system

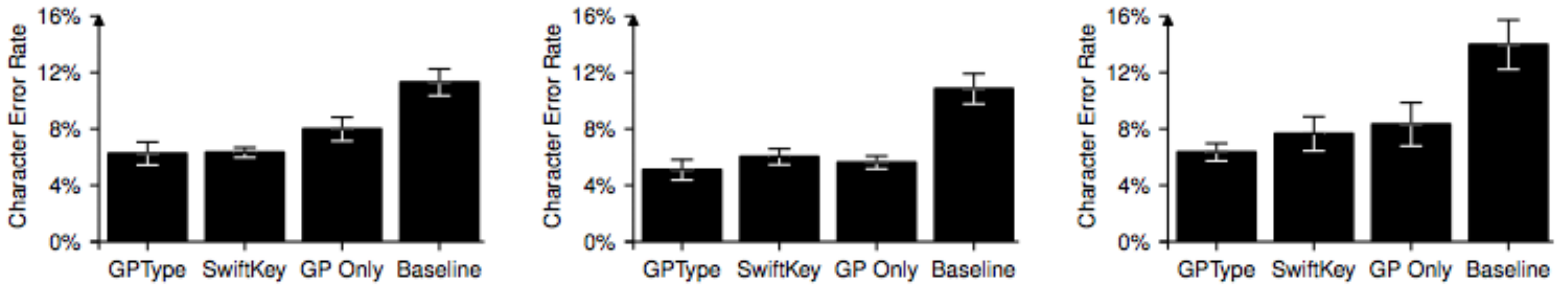
Video

- http://www.youtube.com/watch?v=llQI5gV5l74

The experiment

- 10 participants
- Calibration data collected for each
  - Note: calibration task matters
- each did 3$\times$ 45 minute sessions, typing whilst sitting, standing and walking. [more details in paper]
- Compared:
  - GPtype (our system), Swiftkey (commercial Android keyboard), GP only (just offset, no SLM), baseline (boxing, no SLM).

Results



(a) Sitting (b) Standing (c) Walking

Figure 4. Character error rates for the two keyboards we evaluated, separated by mobility condition (Study 2). Plots show mean and standard error across all participants. The baseline method represents the literal keys touched, while GP Only shows the keys hit after the mean GP offset is applied.

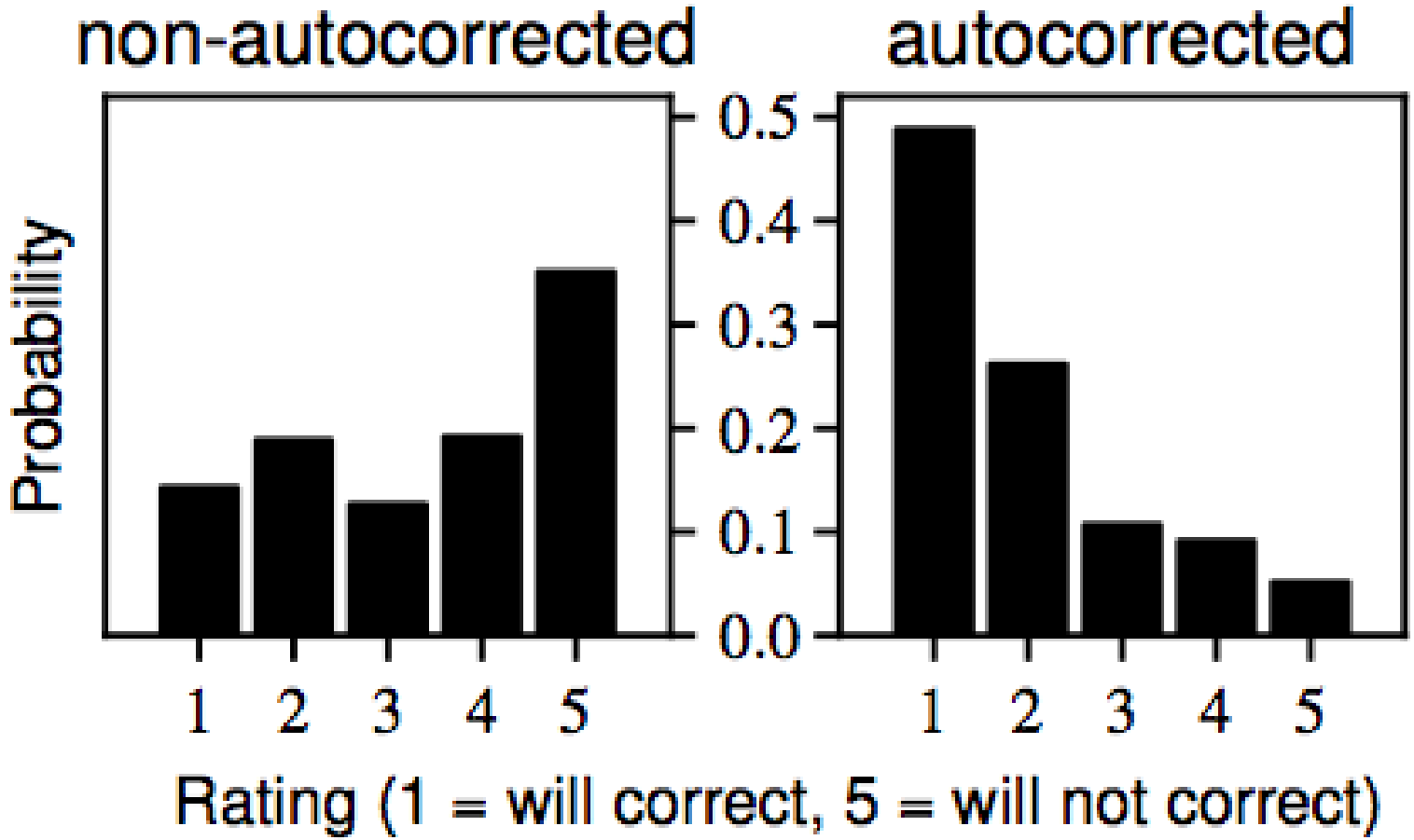
Figure 8 : Results of GPType experiment

- GPType marginally (stat sig) better than Swiftkey.
  - A **lot** of people work on SwiftKey
- Baseline awful!

Explicit uncertainity control

- In GPType, uncertainity is handled implicitly
- As user typing becomes more uncertain, more power given to language model
- Could users *explicitly* control this?
  - Certain inputs: no SLM control (slang, names, etc)
  - Uncertain inputs: high SLM control
- Use pressure to control certainty:
  - High pressure: high certainty
  - Low pressure: low certainty

Do users know when SLM will fail?



- Users given phrases and asked whether they thought autocorrect would change them incorrectly
- Users quite good at understanding SLM failings

ForceType

- Modified Synaptics Forcepad
- Pressure mapped to Gaussian variance (no GP)
- System explained to users
- Users type phrases with and without forcetype

ForceType: Results

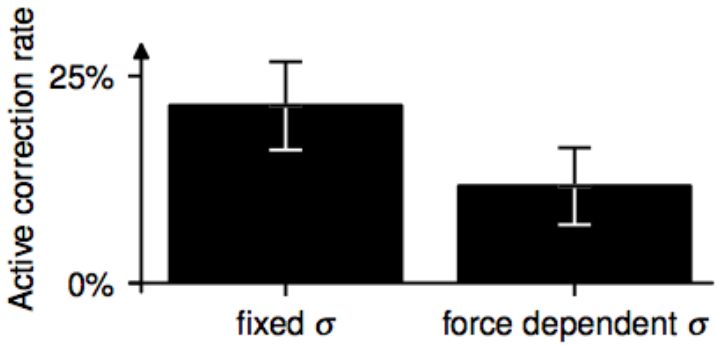


Figure 8. ForceType requires significantly fewer active corrections from users when entering text. Required corrections dropped by ≈ 10 percentage points. Errors bars are 1 sd.

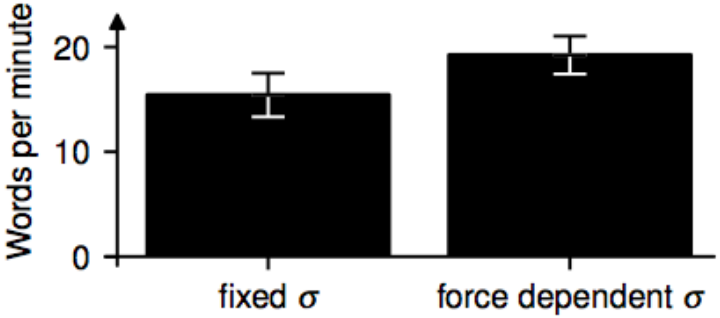


Figure 9. ForceType enabled users to enter phrases > 20% faster. A significant increase over the baseline. Errors bars are 1 sd.

- Forcetype reduced number of corrections performed by users (top)
- Forcetype improved overall text entry rate

Conclusions

- GP regression is key to the approach: we make no parametric assumptions (what would they be?)
- . . . and get probabilistic predictions
- . . . that can be fed to the SLM – (un)certainity is passed to the SLM
- Performance is promising

Conclusions

- GP regression is key to the approach: we make no parametric assumptions (what would they be?)
- . . . and get probabilistic predictions
- . . . that can be fed to the SLM – (un)certainity is passed to the SLM
- Performance is promising
- Can also use pressure to provide explicit uncertainty control

Conclusions

- GP regression is key to the approach: we make no parametric assumptions (what would they be?)
- . . . and get probabilistic predictions
- . . . that can be fed to the SLM – (un)certainity is passed to the SLM
- Performance is promising
- Can also use pressure to provide explicit uncertainty control
- More info:
  - http://www.youtube.com/watch?v=llQI5gV5l74
  - http://pokristensson.com/pubs/WeirEtAlCHI2014.pdf
  - Acknowledgements: Daryl Weir, Per Ola Kristensson, Keith Vertanen, Henning Pohl

Lecture 4: GPs for classification and ordinal regression via the auxiliary variable trick

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

GPs for Classification and ordinal regression

- What if our observation model is non-Gaussian?
  - Classification:

$$P(y_n = 1|f_n) = \int_{-\infty}^{f_n} \mathcal{N}(z|0,1)\ dz = \phi(f_n)$$

  - Logistic Regression:

$$P(y_n = k|f_n) = \phi(b_{k+1}) - \phi(b_k)$$

  - etc
- Analytical inference is no longer possible
- I'll cover how to do inference in these models and extensions with the *auxiliary variable trick*

Binary classification

- Problem setup: we observe N data / target pairs $(\mathbf{x}_n, y_n)$ where $y_n \in \{0, 1\}$
- Place a GP prior on a set of latent variables f_n

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$$

- Use the probit likelihood:

$$P(y_n = 1|f_n) = \phi(f_n) = \int_{-\infty}^{f_n} \mathcal{N}(z|0, 1)\ dz$$

- Inference in this form is hard

Auxiliary Variable Trick

- Re-write the probit function:

$$
\begin{aligned}
P(y_n = 1|f_n) &= \int_{-\infty}^{f_n} N(z|0,1)\ dz \\
&= \int_{-\infty}^{0} N(z|-f_n,1)\ dz \\
&= \int_{0}^{\infty} N(z|f_n,1)\ dz \\
&= \int_{-\infty}^{\infty} \delta(z>0)\mathcal{N}(z|f_n,1)\ dz
\end{aligned}
$$

where $\delta(expr)$ is 1 if $expr$ is true, and 0 otherwise.

Auxiliary Variable Trick

- If we define $P(y_n = 1|z_n) = \delta(z_n > 0)$ then we have:

$$P(y_n = 1|f_n) = \int_{-\infty}^{\infty} P(y_n = 1|z_n)p(z_n|f_n)\ dz_n$$

- and could therefore remove the integral to obtain a model including z_n:

$$p(y_n = 1, z_n|f_n) = P(y_n = 1|z_n)p(z_n|f_n)$$

- Doing inference in this model (i.e. with additional variables z_n) is much easier (but still not analytically tractable)
- Note: $P(y_n = 0|z_n) = \delta(z_n < 0)$

Example - Gibbs sampling for binary classification

- An easy way to perform inference in the augmented model is via Gibbs sampling
- Sample $z_n|f_n, y_n$:

$$\begin{aligned} p(z_n|f_n, y_n = 0) &\propto \delta(z_n < 0)\mathcal{N}(z_n|f_n, 1) \\ p(z_n|f_n, y_n = 1) &\propto \delta(z_n < 1)\mathcal{N}(z_n|f_n, 1) \end{aligned}$$

Example - Gibbs sampling for binary classification

- An easy way to perform inference in the augmented model is via Gibbs sampling
- Sample $z_n|f_n, y_n$:

$$\begin{aligned} p(z_n|f_n, y_n = 0) &\propto \delta(z_n < 0)\mathcal{N}(z_n|f_n, 1) \\ p(z_n|f_n, y_n = 1) &\propto \delta(z_n < 1)\mathcal{N}(z_n|f_n, 1) \end{aligned}$$

- Sample $\mathbf{f}|\mathbf{z}, \mathbf{C}$

$$p(\mathbf{f}|\mathbf{z}, \mathbf{C}) = \mathcal{N}(\boldsymbol{\mu}_f, \boldsymbol{\Sigma}_f)$$

where

$$\boldsymbol{\Sigma}_f = \left(\mathbf{I} + \mathbf{C}^{-1}\right)^{-1}, \quad \boldsymbol{\mu}_f = \boldsymbol{\Sigma}_f^{-1}\mathbf{z}$$

- Repeat ad infinitum

Example - Gibbs sampling for binary classification

- To make predictions:
  - At each sampling step, do a (noise-free) GP regression using the current sample of $\mathbf{f}$ to get a density over f_* (Details in a previous slide).
  - Sample a specific realisation of f_* from this density.
  - Compute $\phi(f_*)$ (or sample a z_* and then record whether it's > 0 or not)
  - Average this value over all Gibbs sampling iterations!

Example - binary classification

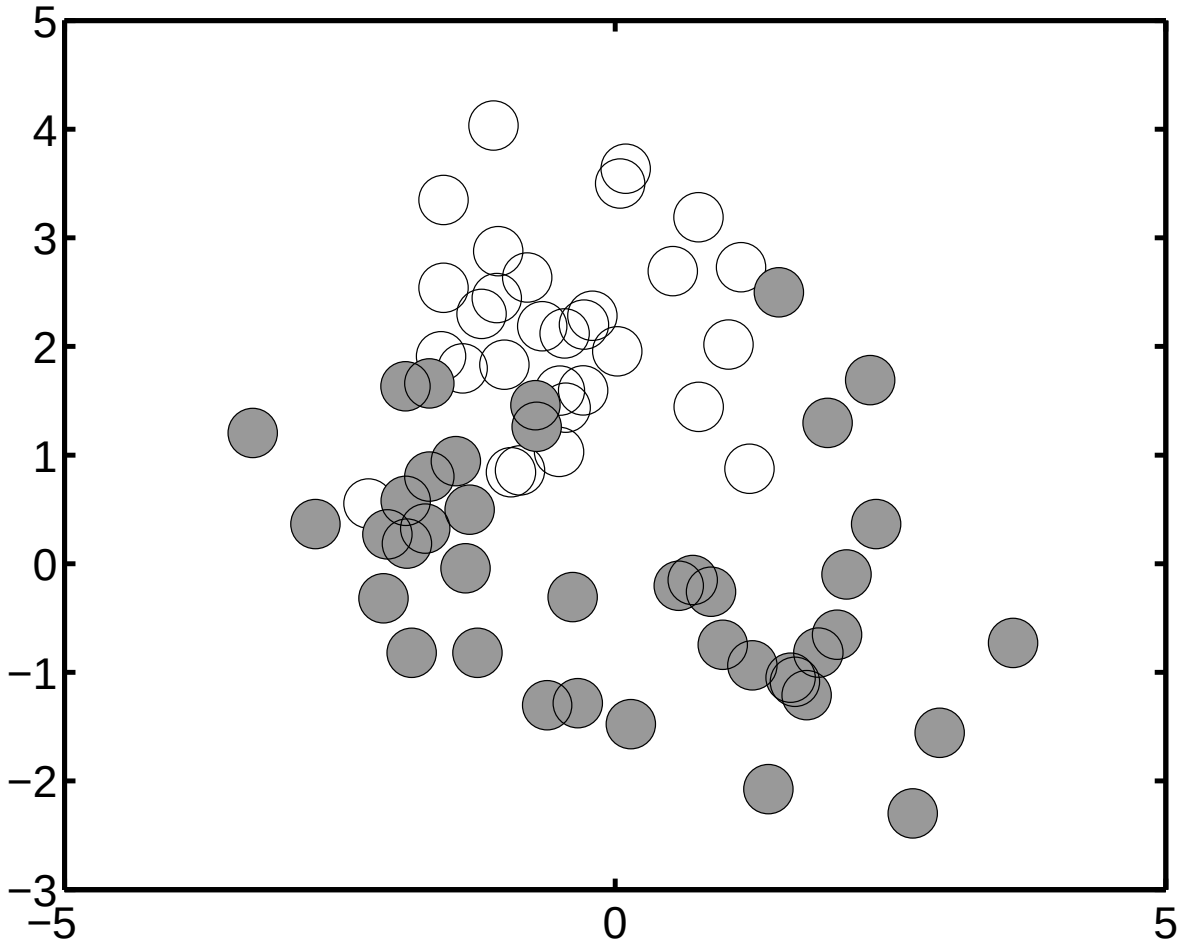


Figure 9 : Some simple classification data

Example - binary classification

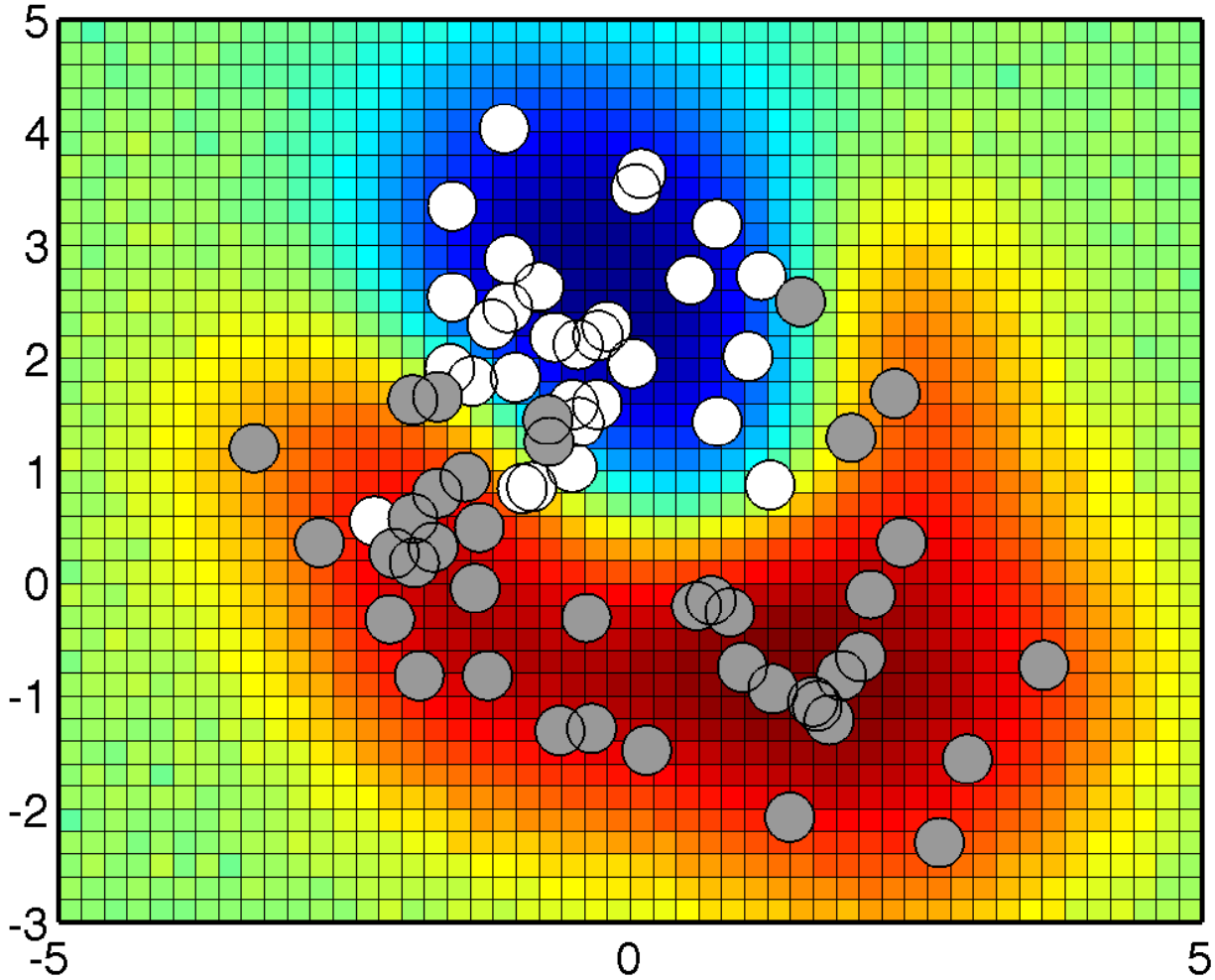


Figure 10 : Predictive probabilities averaged over 1000 Gibbs samples using an RBF covariance. As γ is increased, the model overfits.

Example - binary classification

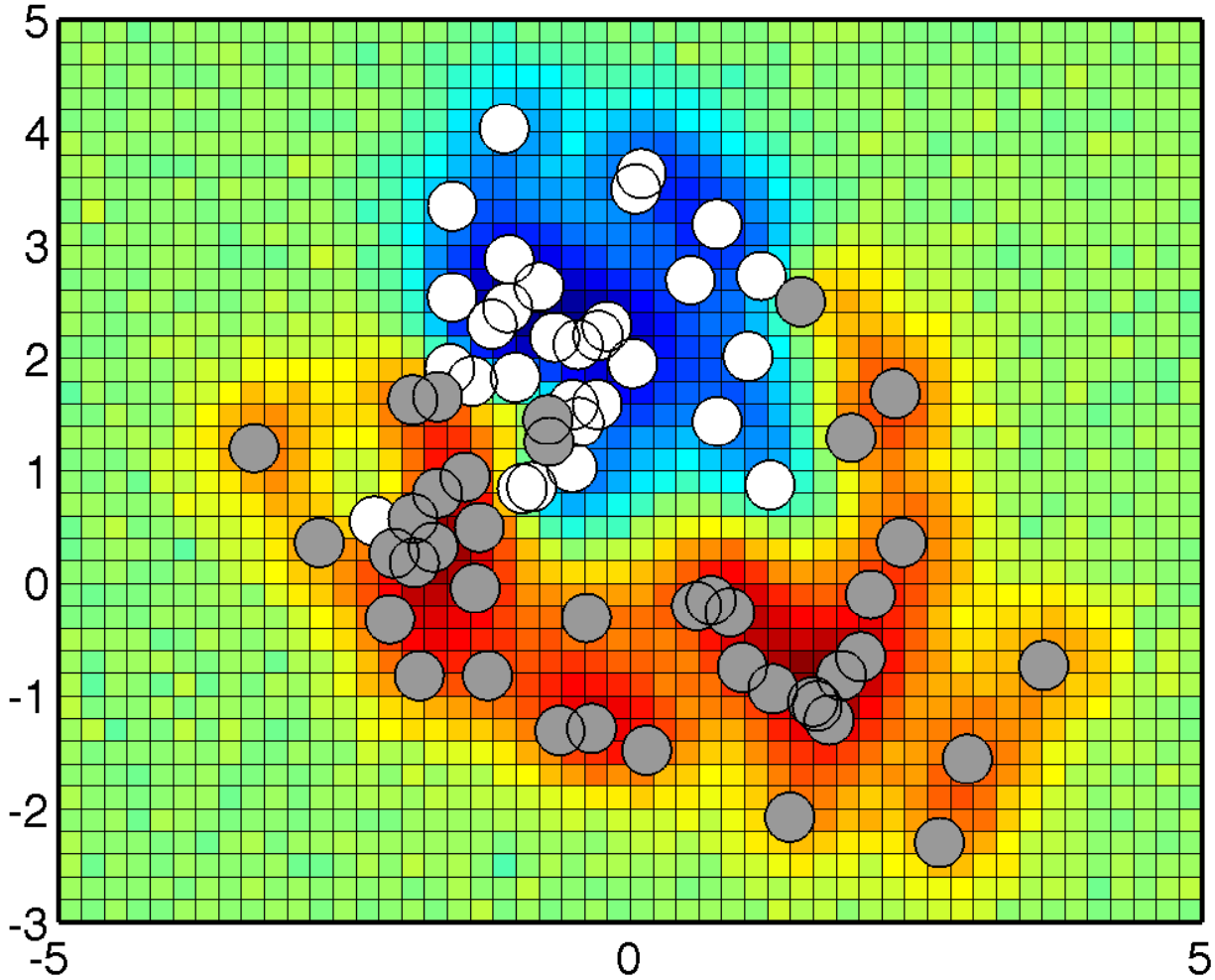


Figure 10 : Predictive probabilities averaged over 1000 Gibbs samples using an RBF covariance. As γ is increased, the model overfits.

Example - binary classification

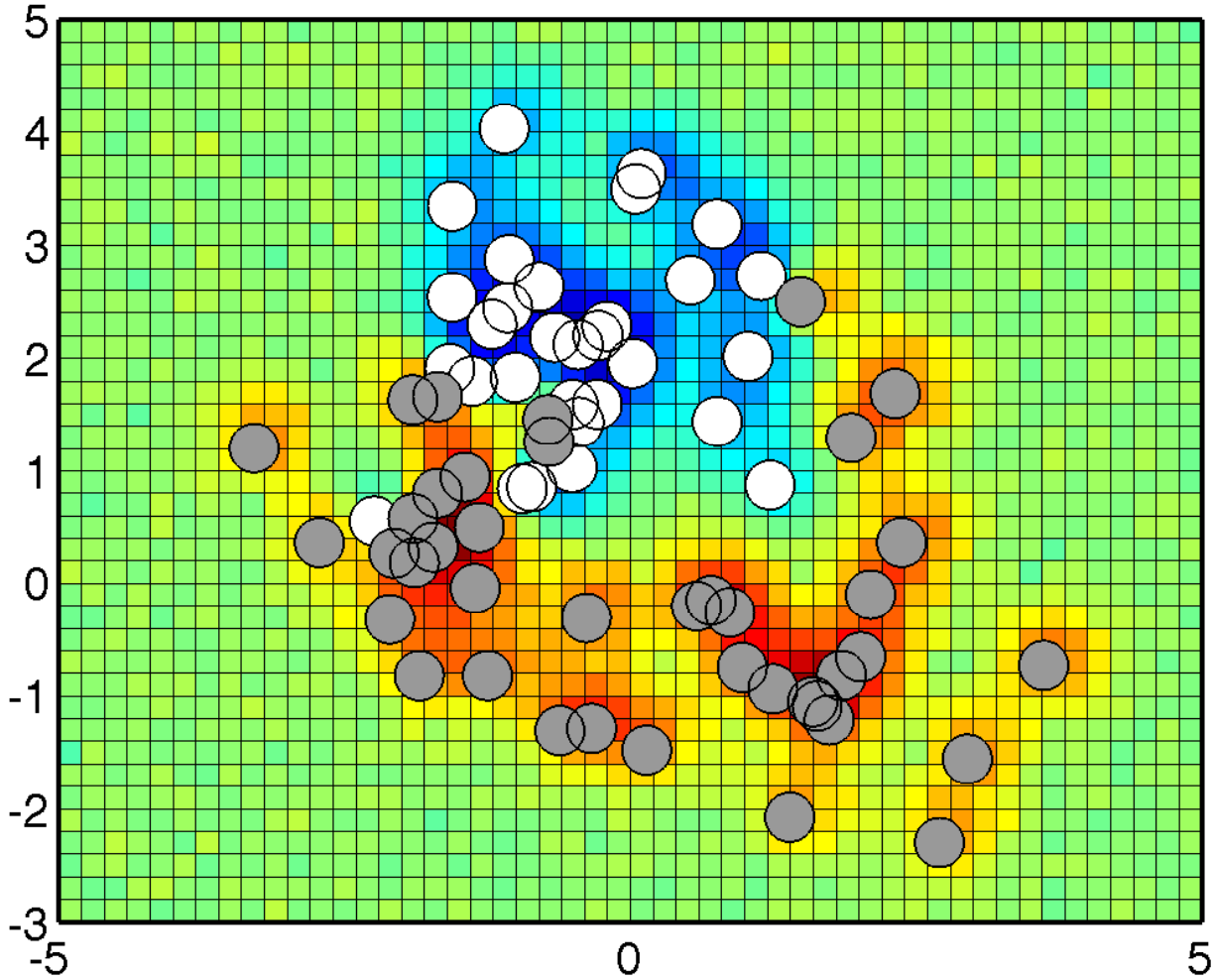


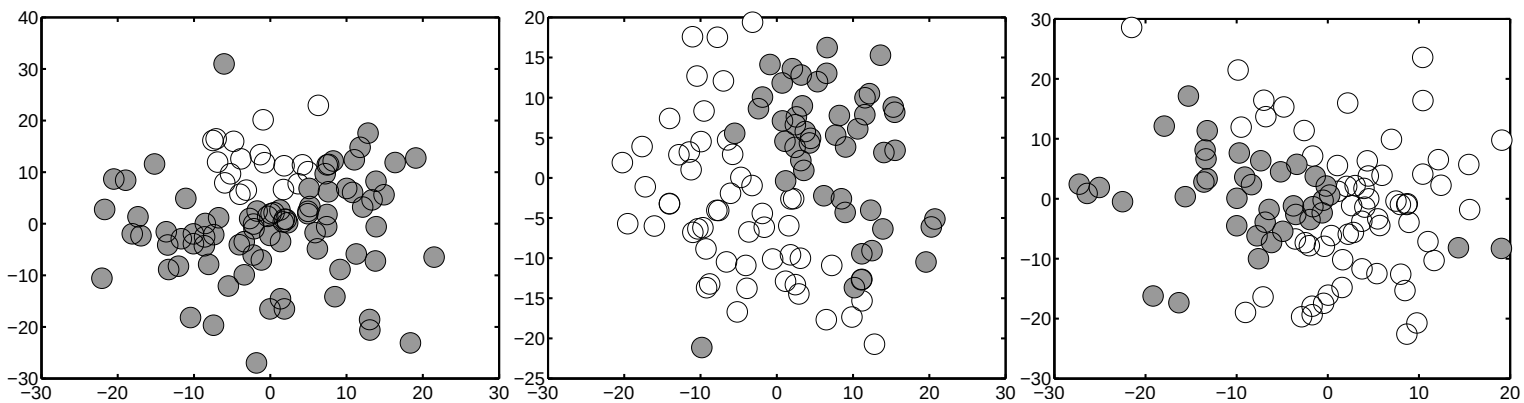
Figure 10 : Predictive probabilities averaged over 1000 Gibbs samples using an RBF covariance. As γ is increased, the model overfits.

Note

- Inference:
  - Gibbs sampling isn't the only option
  - A popular alternative is Variational Bayes

Note 2 – The Generative Process

- Sometimes it's useful to think of the generative process defined by the model.
- In this case, to generate N values of y_n given the associated x_n:
  - Sample $\mathbf{f}$ from a GP with mean $\mathbf{0}$ and Covariance matrix $\mathbf{C}$.
  - For each $n = 1 \ldots N$:
    - Sample $z_n \sim \mathcal{N}(f_n, 1)$
    - If $z_n > 0$ set $y_n = 1$, otherwise $y_n = 0$.
- Some examples:



GP classification exercise

TASK [2]

- Explore GP binary classification with auxiliary variables using `gp_class_task.m`
- Try:
  - Generating data from different distributions
  - Varying covariance function and parameters
  - Taking more posterior samples
- You will also need `plotClassdata.m` and `kernel.m`

A more general idea

- Models of this form:
  - $\mathbf{f} \sim GP$
  - $z_n \sim \mathcal{N}(f_n, 1)$
  - $P(y_n|z_n) = \delta(f(z_n))$
- Can be used for more than just binary classification.

A more general idea

- Models of this form:
  - $\mathbf{f} \sim GP$
  - $z_n \sim \mathcal{N}(f_n, 1)$
  - $P(y_n|z_n) = \delta(f(z_n))$
- Can be used for more than just binary classification.
- Ordinal Regression:
  - $P(y_n = k|z_n)$ is now chopped at both ends:

$$P(y_n = k|z_n) = \delta(b_k < z_n < b_{k+1})$$

  - Gibbs distribution for z_n therefore involves a Gaussian truncated at both ends.

A more general idea

- Models of this form:
  - $\mathbf{f} \sim GP$
  - $z_n \sim \mathcal{N}(f_n, 1)$
  - $P(y_n|z_n) = \delta(f(z_n))$
- Can be used for more than just binary classification.
- Ordinal Regression:
  - $P(y_n = k|z_n)$ is now chopped at both ends:

$$P(y_n = k|z_n) = \delta(b_k < z_n < b_{k+1})$$

  - Gibbs distribution for z_n therefore involves a Gaussian truncated at both ends.
- As well as multi-class and semi-supervised classification...

Multi-class classification

- The previous treatment can be extended to multiple classes.
- For a problem with K classes:
  - K GP priors, K N-dimensional latent vectors $\mathbf{f}_k$.
  - $N \times K$ auxiliary variables $z_{nk} \sim \mathcal{N}(f_{nk}, 1)$
  - And:

$$P(y_n = k | z_{n1}, \ldots, z_{nK}) = \delta(z_{nk} > z_{ni} \quad \forall i \neq k)$$

Multi-class classification

- The previous treatment can be extended to multiple classes.
- For a problem with K classes:
  - K GP priors, K N-dimensional latent vectors $\mathbf{f}_k$.
  - $N \times K$ auxiliary variables $z_{nk} \sim \mathcal{N}(f_{nk}, 1)$
  - And:

$$P(y_n = k | z_{n1}, \ldots, z_{nK}) = \delta(z_{nk} > z_{ni} \quad \forall i \neq k)$$

- Gibbs sampling is similar to the binary case:
  - Only tricky bit is efficently sampling from a K-dimensionl MVG truncated such that the kth element is largest.

Multi-class classification

- The previous treatment can be extended to multiple classes.
- For a problem with K classes:
  - K GP priors, K N-dimensional latent vectors $\mathbf{f}_k$.
  - $N \times K$ auxiliary variables $z_{nk} \sim \mathcal{N}(f_{nk}, 1)$
  - And:

$$P(y_n = k | z_{n1}, \ldots, z_{nK}) = \delta(z_{nk} > z_{ni} \quad \forall i \neq k)$$

- Gibbs sampling is similar to the binary case:
  - Only tricky bit is efficently sampling from a K-dimensionl MVG truncated such that the kth element is largest.
- Details of a Variational Bayes inference scheme in: Girolami and Rogers 2006

Multi-class Example

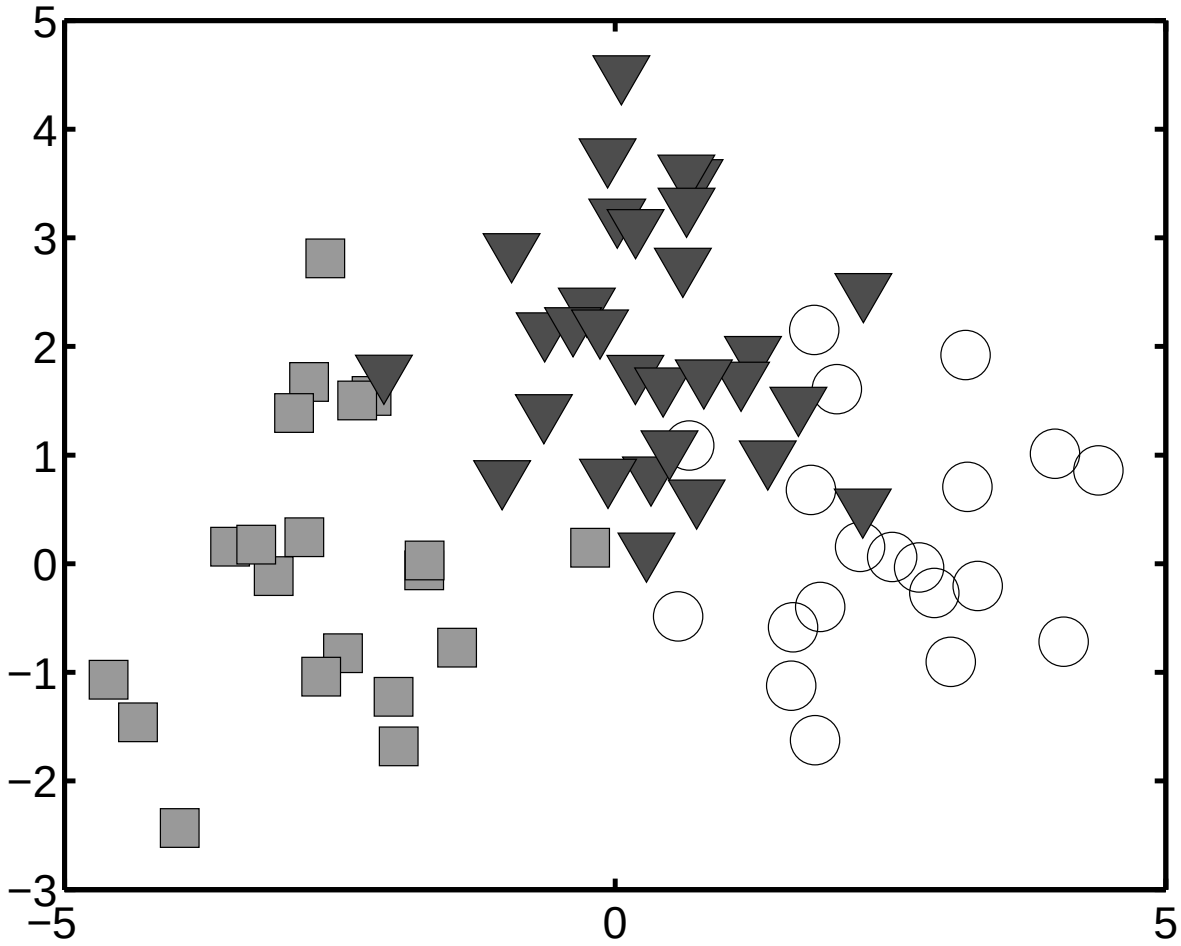


Figure 11 : Multi-class classification example. RBF covariance, $\gamma = 1$.

Multi-class Example

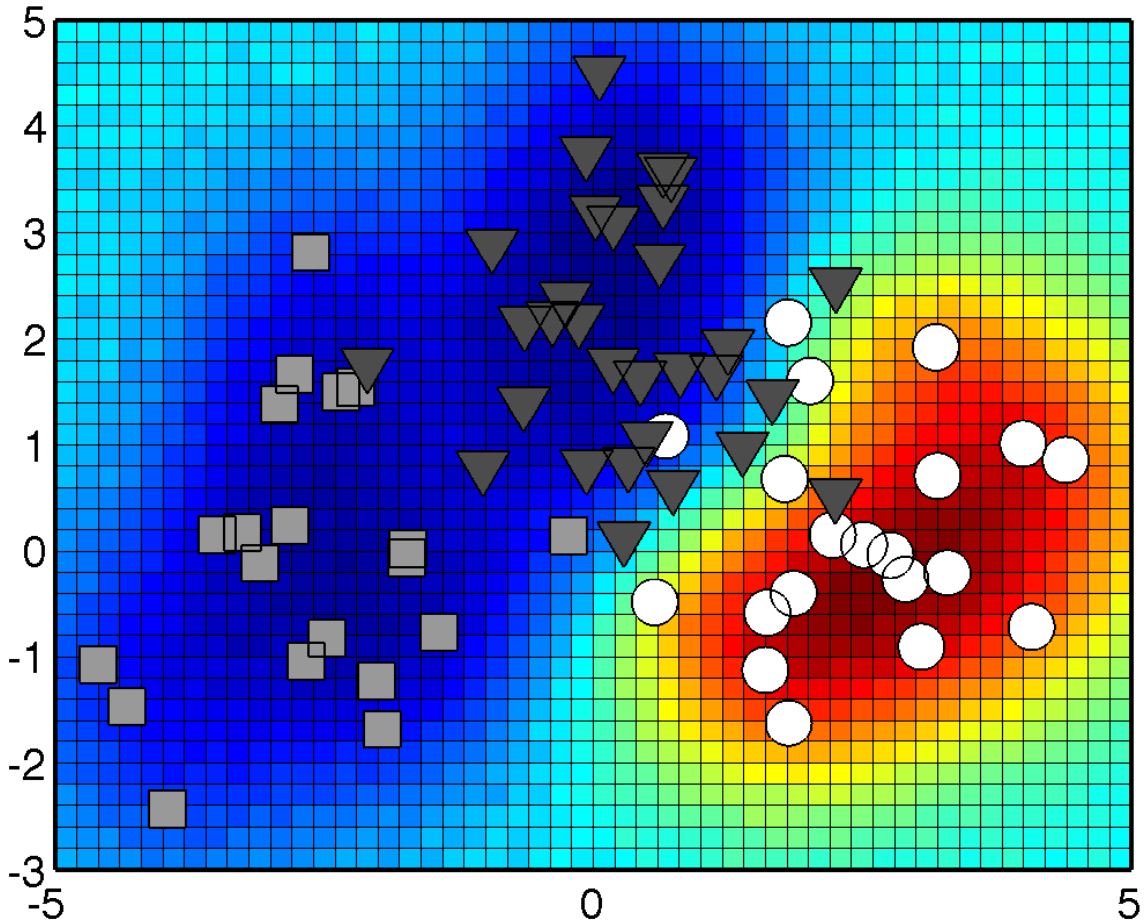


Figure 11 : Multi-class classification example. RBF covariance, $\gamma = 1$.

Multi-class Example

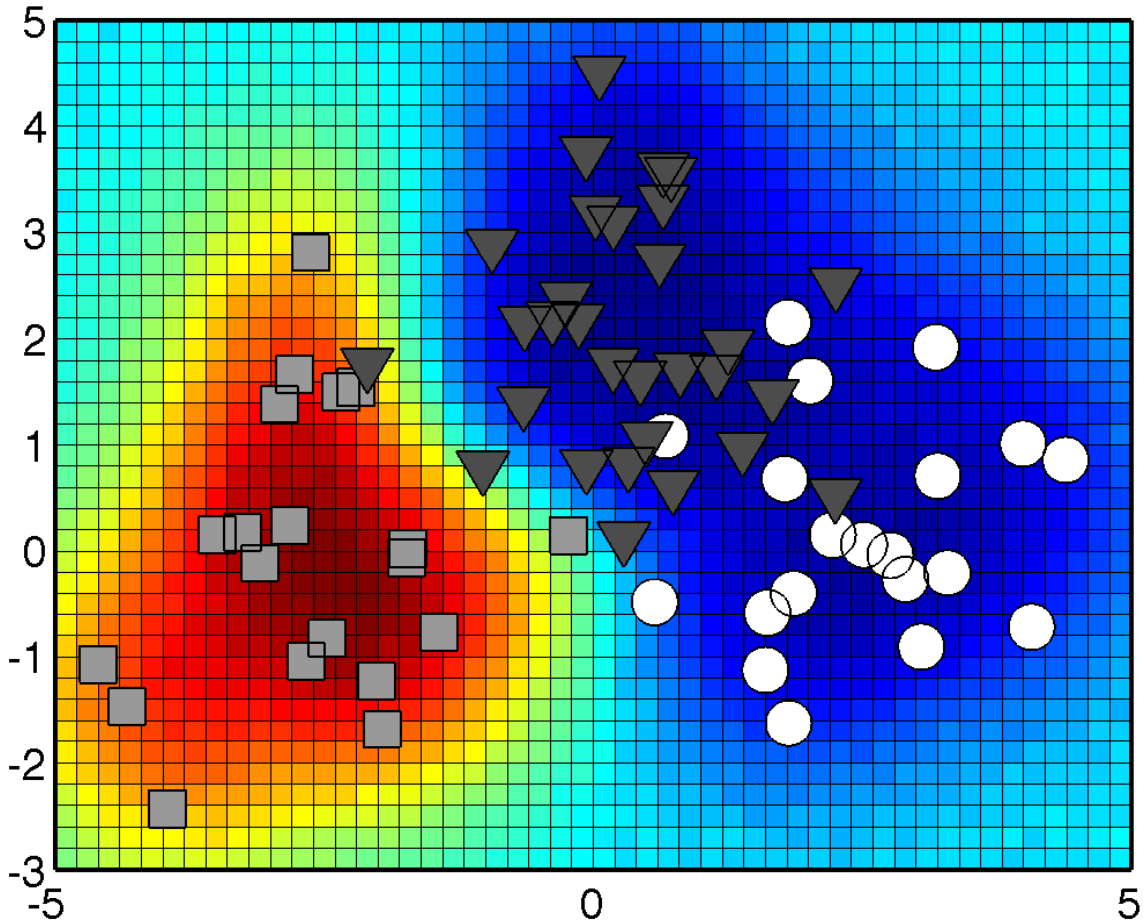


Figure 11 : Multi-class classification example. RBF covariance, $\gamma = 1$.

Multi-class Example

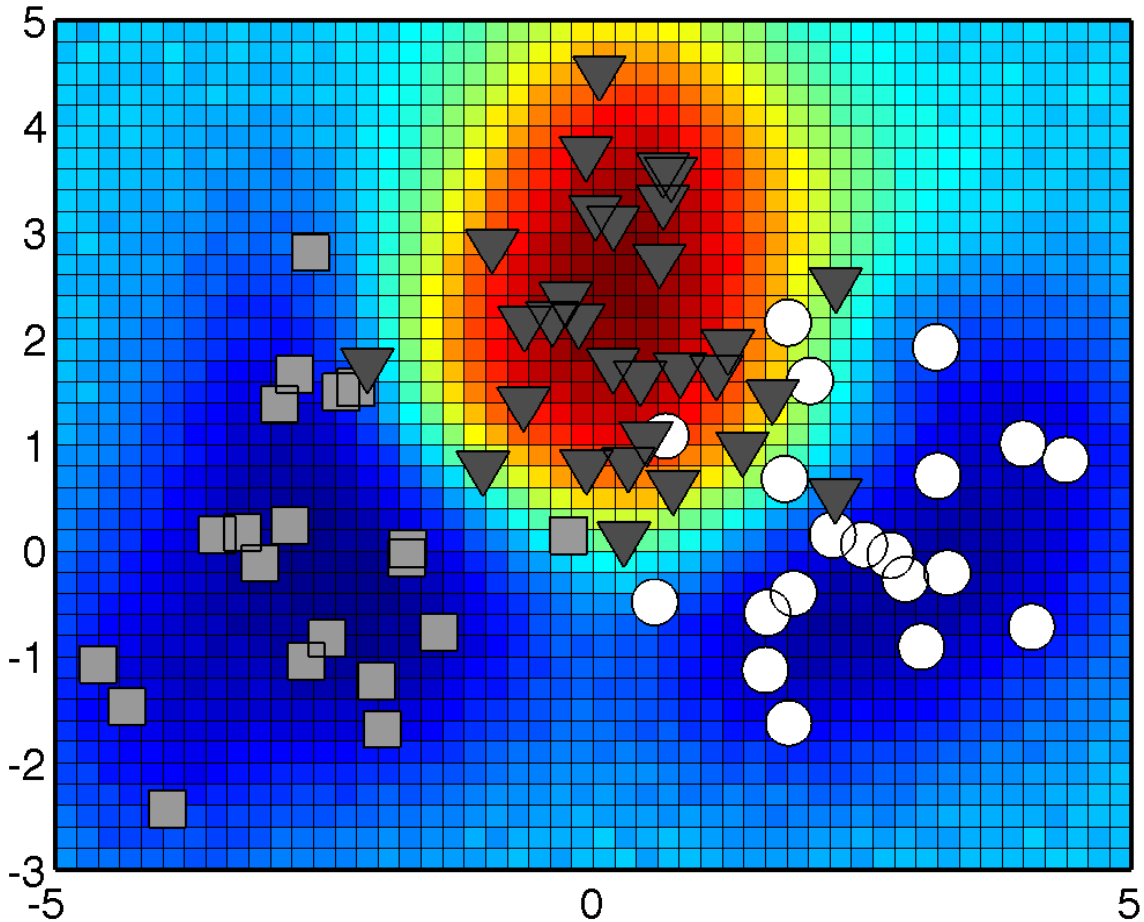


Figure 11 : Multi-class classification example. RBF covariance, $\gamma = 1$.

Semi-supervised Classification

- In some domains, only a subset of data are labeled [e.g. image classification]

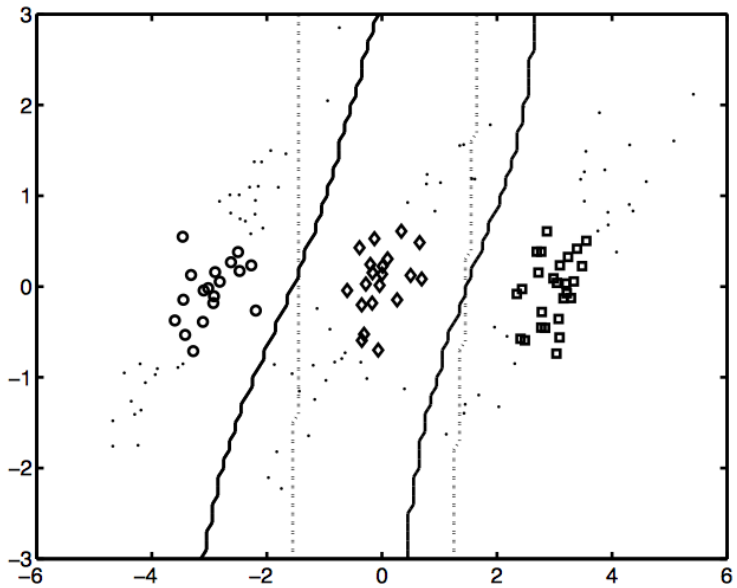


Figure 12 : A toy semi-supervised classification problem.

- Can be overcome using the Null Category Noise Model (NCNM) Lawrence and Jordan 2004

The NCNM

- Going back to binary classification, the auxiliary variable trick can be visualised:

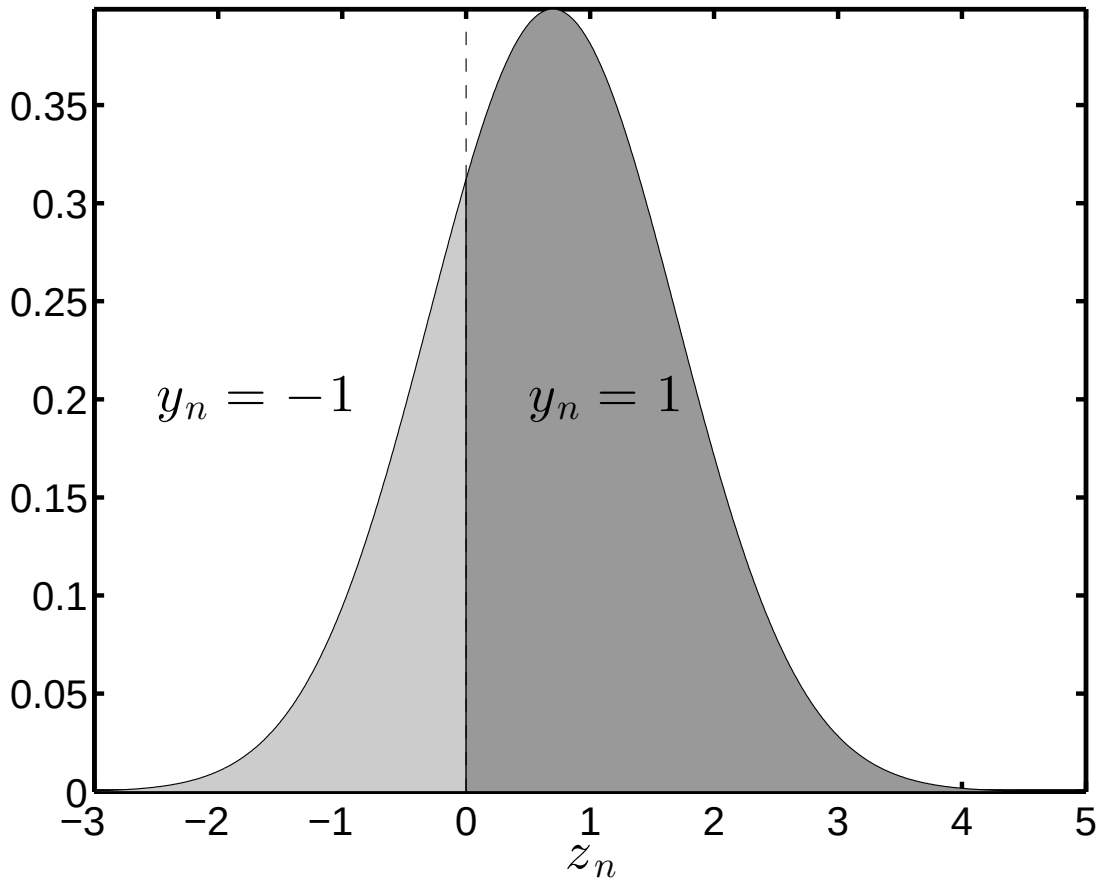


Figure 13 : Visualisation of the auxiliary variable trick. The Gaussian has mean f_n. Note that I'm not calling the classes ± 1.

The NCNM

- To include unlabeled data, we add a third category, for $y_n = 0$:

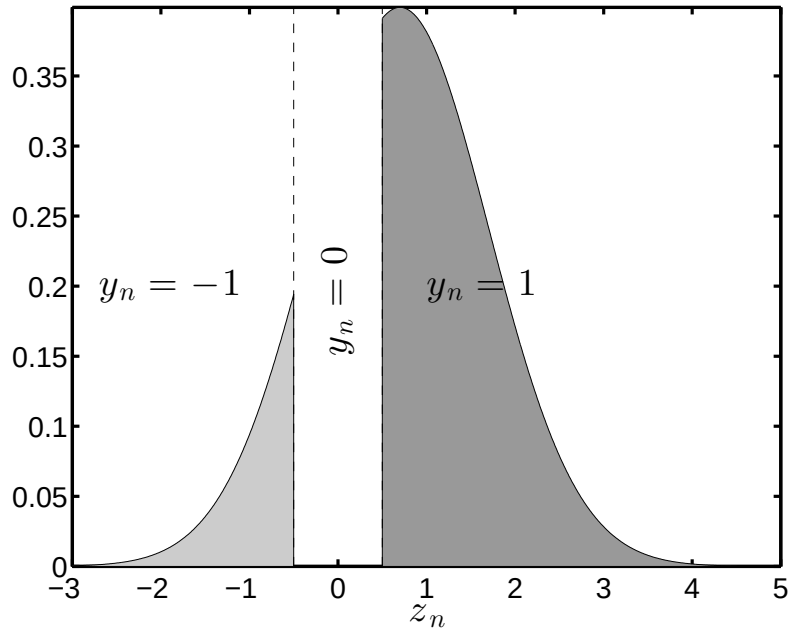


Figure 14 : Visualisation of the NCNM with a null region of width 1.

$$p(y_n|z_n) = \begin{cases} \delta(z_n < -a) & y_n = -1 \\ \delta(z_n > a) & y_n = 1 \\ \delta(z_n > -a) - \delta(z_n > a) & y_n = 0 \end{cases}$$

The NCNM

- The final step is to introduce another set of latent variables.
  - $g_n = 0$ if y_n is observed (i.e. labeled) and $g_n = 1$ otherwise.
- And enforce the constraint that no unlabeled points can exist in the null region:

$$P(y_n = 0 | g_n = 1) = 0$$

The NCNM

- The final step is to introduce another set of latent variables.
  - $g_n = 0$ if y_n is observed (i.e. labeled) and $g_n = 1$ otherwise.
- And enforce the constraint that no unlabeled points can exist in the null region:

$$P(y_n = 0 | g_n = 1) = 0$$

- This has the effect of introducing an empty region around the decision boundary
  - i.e. pushing the decision boundary into regions of empty space

The NCNM

- The final step is to introduce another set of latent variables.
  - $g_n = 0$ if y_n is observed (i.e. labeled) and $g_n = 1$ otherwise.
- And enforce the constraint that no unlabeled points can exist in the null region:

$$P(y_n = 0 | g_n = 1) = 0$$

- This has the effect of introducing an empty region around the decision boundary
  - i.e. pushing the decision boundary into regions of empty space
- Inference:
  - Gibbs sampling is the same as the binary case except $z_n | f_n, g_n = 1$.
  - This is a mixture of two truncated Gaussians – sample the component, and then sample z_n.

NCNM Example

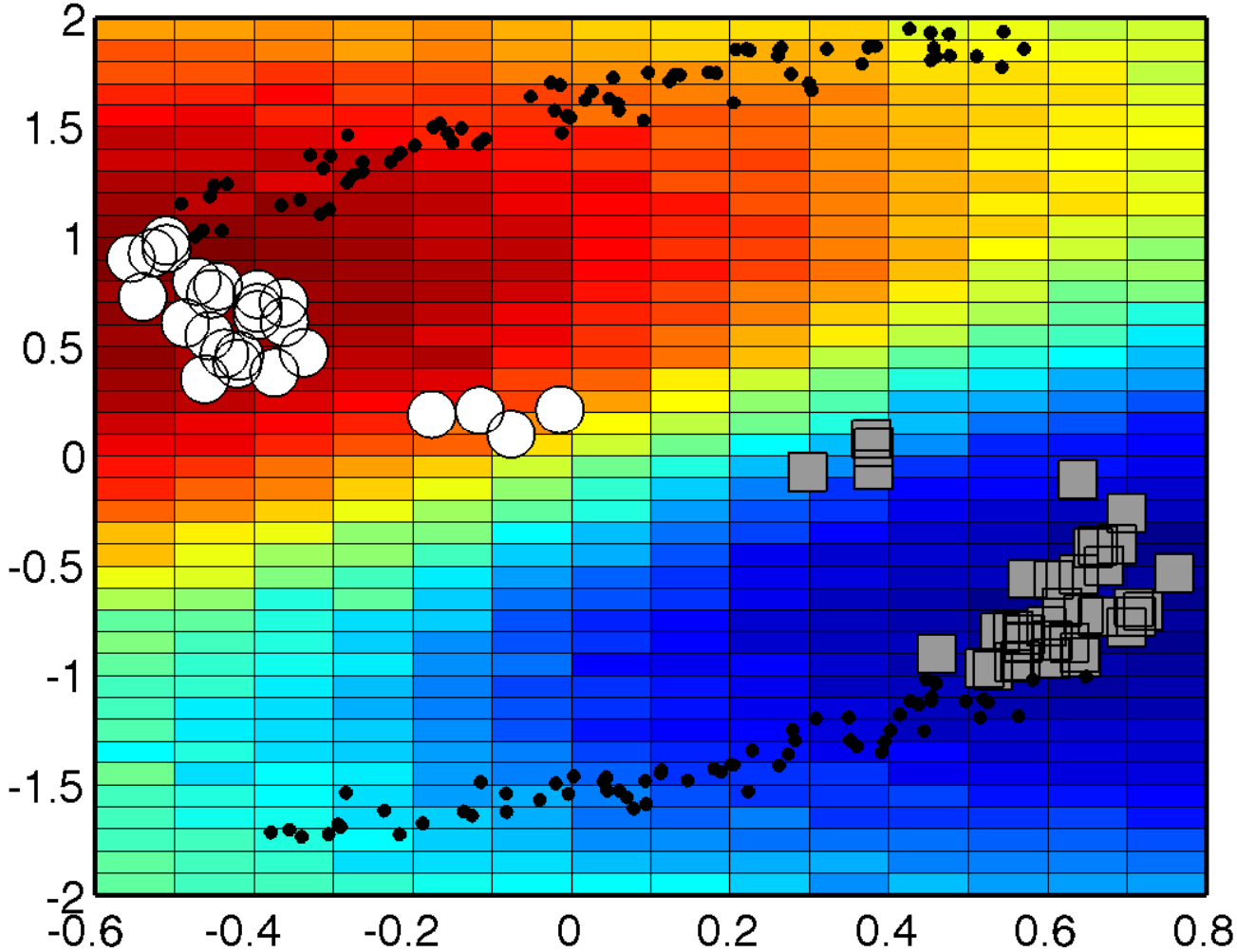


Figure 15 : Standard GP classification (unlabeled data ignored)

NCNM Example

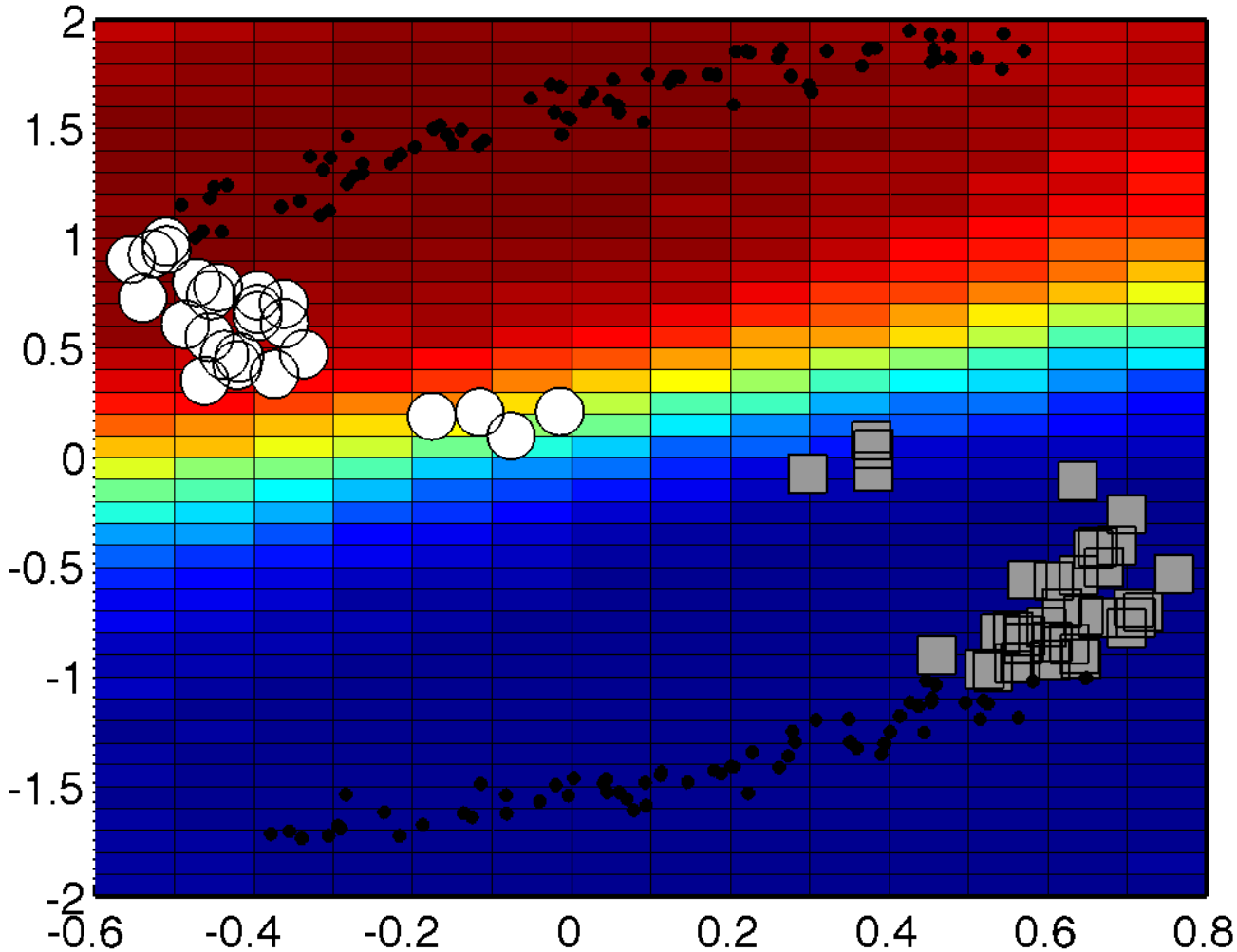


Figure 16 : NCNM GP classification

NCNM Exercise

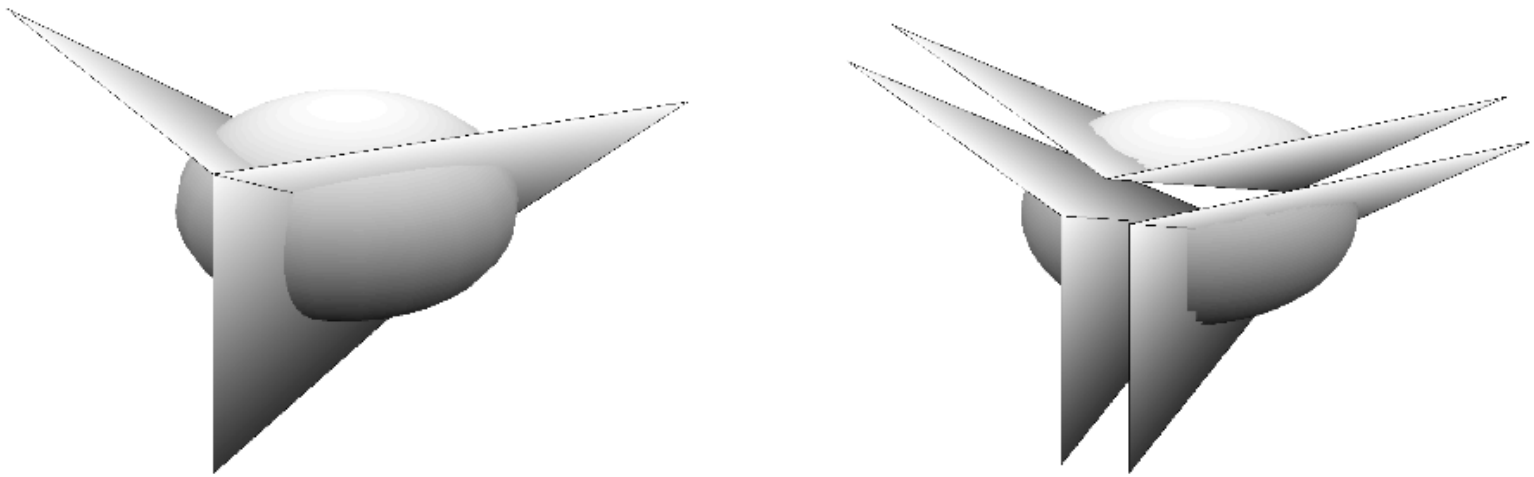
TASK [3]

- Experiment with the NCNM using `gp_ncnm_task.m`
- Setting `a=0` results in the standard model
- Setting `a>0` uses the NCNM
- It's not always easy to get the results you want to see!

Multi-class NCNM

- This idea can be extended to the multi-class setting.
- See Rogers and Girolami 2007

$$P(y_n = k|z_{n1}, \ldots, z_{nK}) = \begin{cases} \delta(z_{nk} > z_{ni} + a \quad \forall i \neq k) & y_n > 0 \\ 1 - \sum_j \delta(z_{nj} > z_{ni} + a \quad \forall i \neq j) & y_n = 0 \end{cases}$$



(a) A visualisation of the truncation caused by the standard multi-class probit model

(b) A visualisation of the truncation caused by the multi-class probit model with a null region

Figure 17 : Visualisation of truncation

Summary

- GP priors aren't restricted to regression.
- Analytical solutions aren't possible
- Auxiiliary Variable Trick makes inference (via Gibbs sampling or Variational Bayes) straightforward for:
  - Binary classification
  - Ordinal regression
  - Multi-class classification
  - Semi-supervised classification (binary and mutli-class)
  - As well as others (e.g. binary PCA)

Lecture 5: Application: Clinical Ratings

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Clinicians disagree in AandE

- Patients in Accident and Emergency (A&E) are continually monitored.
  - Heart rate
  - Blood pressure
  - Temperature
  - etc

Clinicians disagree in AandE

- Patients in A&E are continually monitored.
  - Heart rate
  - Blood pressure
  - Temperature
  - etc
- Based on these hourly observations, clinicians (in a Glasgow hospital) give each patient an ordinal rating
  - A (healthy(ish)), B, C, D, E, F (critical)

Clinicians disagree in AandE

- Patients in A&E are continually monitored.
  - Heart rate
  - Blood pressure
  - Temperature
  - etc
- Based on these hourly observations, clinicians (in a Glasgow hospital) give each patient an ordinal rating
  - A (healthy(ish)), B, C, D, E, F (critical)
- These ratings are *subjective*
  - How do clinicians disagree? (variance? bias?)
- More details of this work in Rogers et al 2013 and Rogers et al 2010

Data

- $c = 1 \ldots C$ clinicians.
- $p = 1 \ldots P$ patients.
- For patient p, we have T_p observations at times $\mathbf{t}_p = [t_{p1}, \ldots, t_{pT_p}]^T$.
- $y^p_{\tau c}$ is rating at $t_{p\tau}$ ($\{A, B, C, D, E\}$).

The model

- Assumptions:
  - We assume that there is an unobserved continuous latent health function for each patient
  - Each clinician observes this and and corrupts it in two ways:
    - Adds Gaussian noise
    - Adds a constant offset (optimist, pessimist)
  - Corrupted health is then binned to give category

The model

- Assumptions:
  - We assume that there is an unobserved continuous latent health function for each patient
  - Each clinician observes this and and corrupts it in two ways:
    - Adds Gaussian noise
    - Adds a constant offset (optimist, pessimist)
  - Corrupted health is then binned to give category
- The model:
  - Patient health is modelled as a GP.
  - Corrupted health is the auxiliary variable (q) – one set of auxiliary variables per clinician
  - q is binned to produce rating.
  - Note that $p(q|f)$ has been generalised from standard normal.

Model

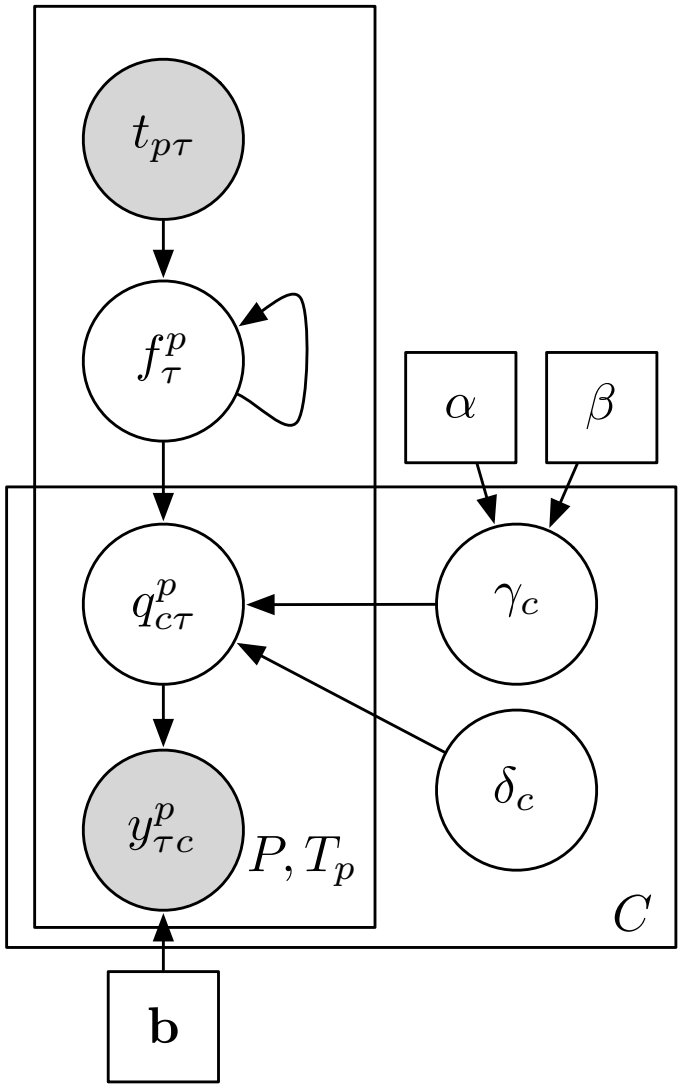


Figure 18 : Plates diagram

$$
\begin{aligned}
\mathbf{f}^p &\sim \mathcal{N}(\mathbf{0}, \mathbf{C}^p) \quad \text{[health]} \\
\delta_c &\sim \mathcal{N}(0, 1) \quad \text{[offset]} \\
\gamma_c &\sim \mathcal{G}(\alpha, \beta) \quad \text{[precision]} \\
q^p_{c\tau} &\sim \mathcal{N}(f^p_\tau + \delta_c, \gamma_c^{-1}) \\
P(y^p_{\tau c} = k) &= \delta(b_k < q^p_{c\tau} < b_{k+1})
\end{aligned}
$$

Previously auxiliary variables were $z_n \sim \mathcal{N}(f_n, 1)$. This model adds clinician-specific offsets and precisions: $q^p_{c\tau} \sim \mathcal{N}(f^p_\tau + \delta_c, \gamma_c^{-1})$.

Example data generation

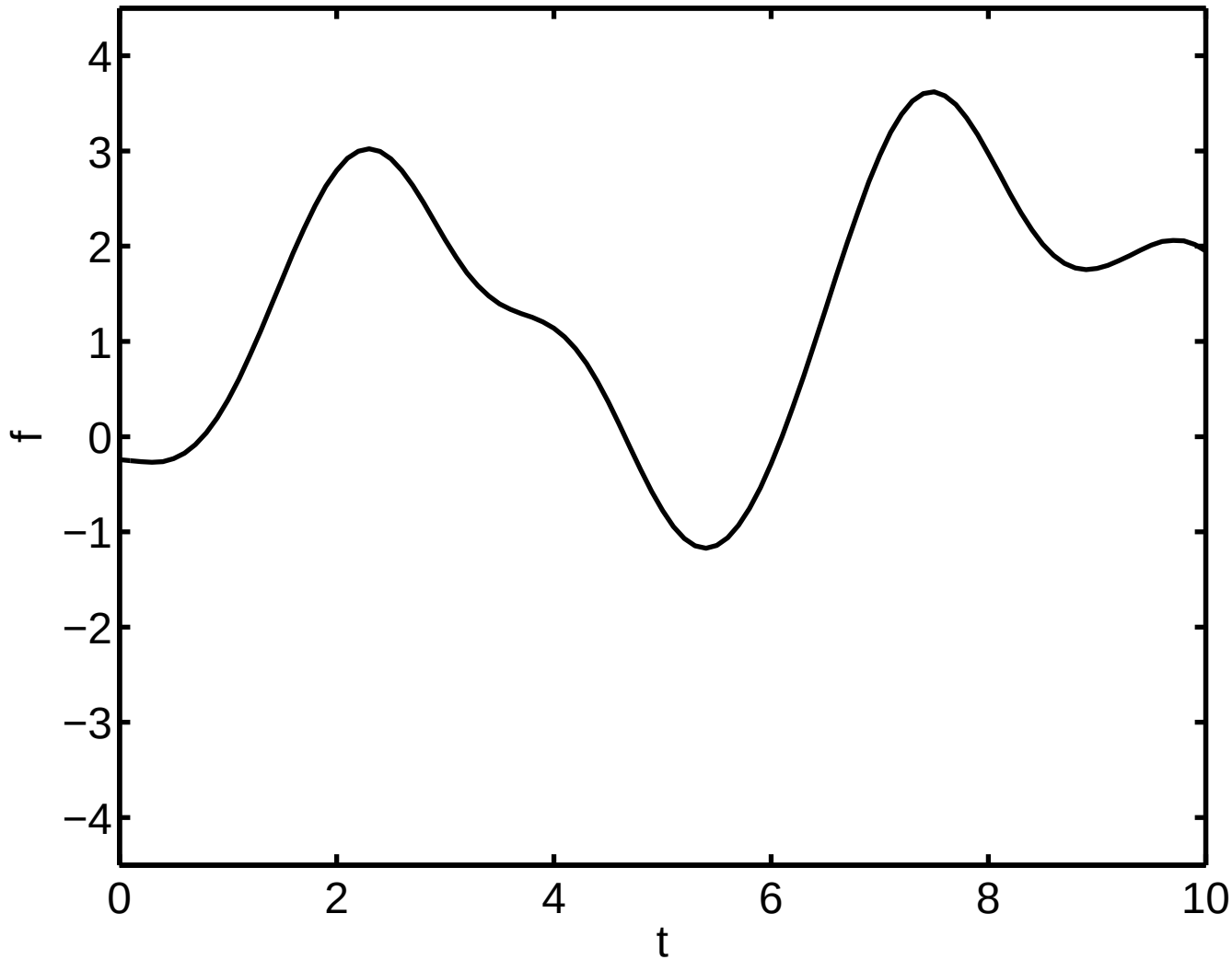


Figure 19 : Example of the generative process described by the model for three clinicans.

Example data generation

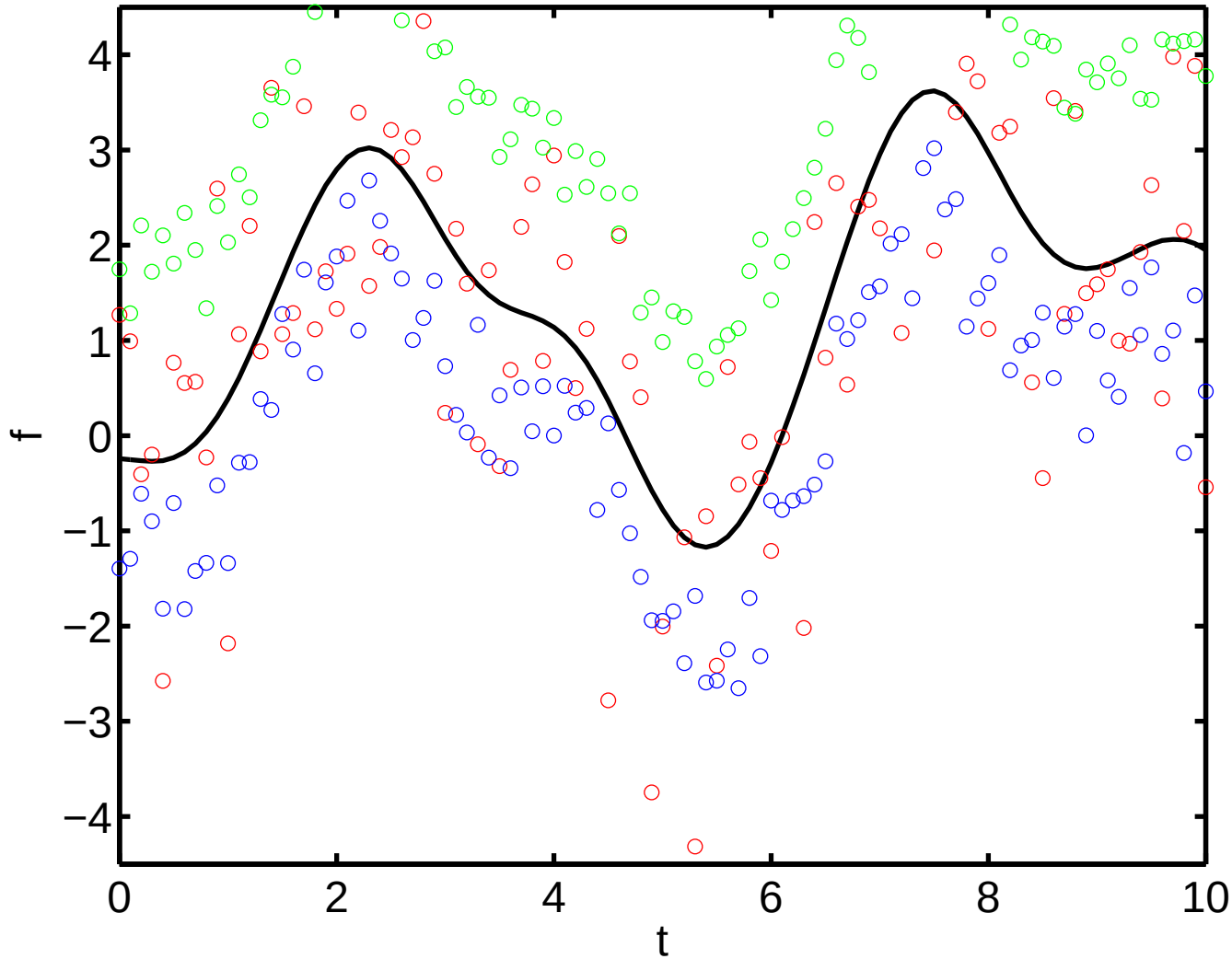


Figure 19 : Example of the generative process described by the model for three clinicans.

Example data generation

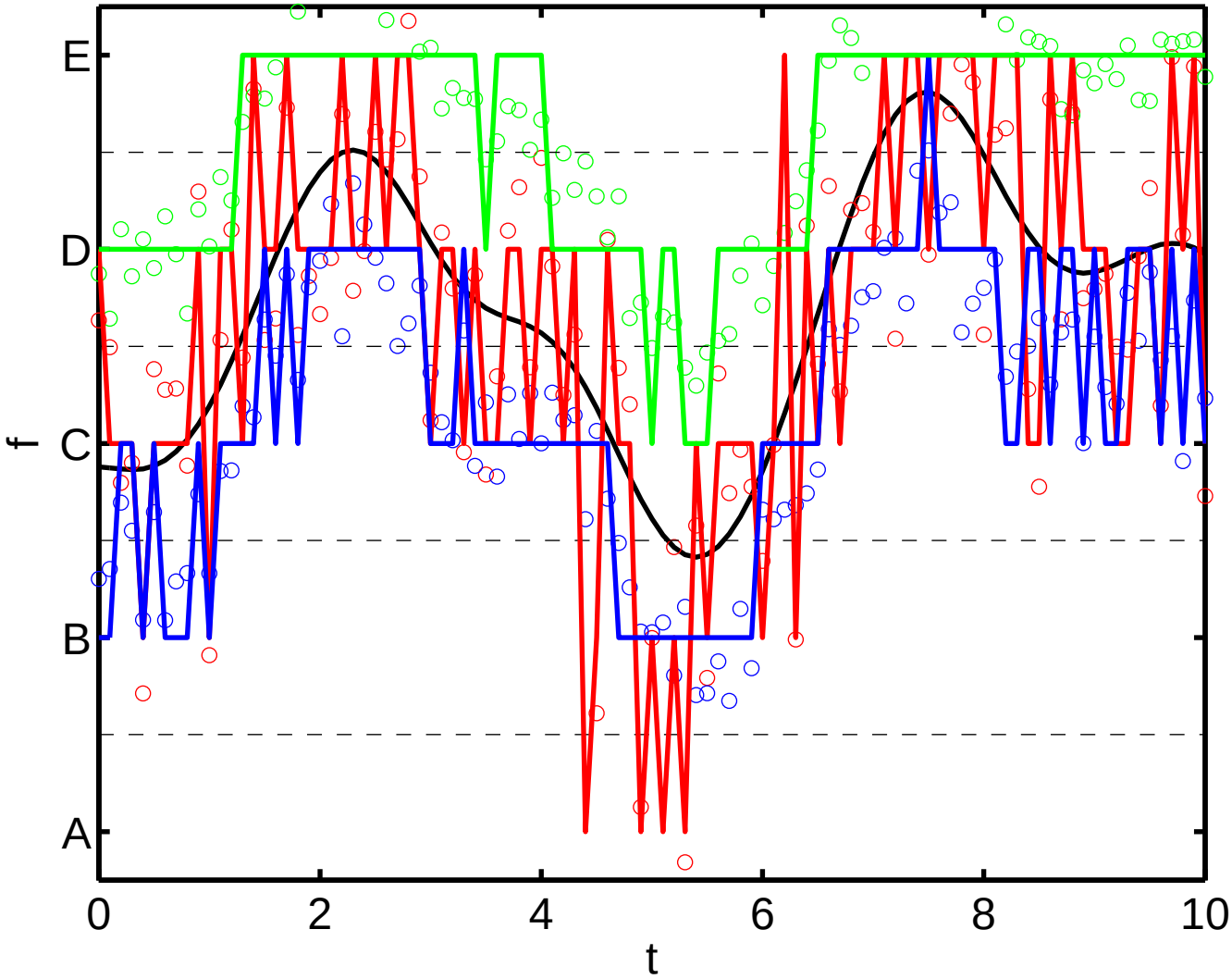


Figure 19 : Example of the generative process described by the model for three clinicans.

Model inference

- Gibbs sampling is straightforward
  - The latent health function for each patient.

$$p(\mathbf{f}^p|\ldots) = \mathcal{N}(\mathbf{f}^p|\boldsymbol{\mu}_{\mathbf{f}^p}, \boldsymbol{\Sigma}_{\mathbf{f}^p})$$

where:

$$\boldsymbol{\Sigma}_{\mathbf{f}^p} = \left((\mathbf{C}^p)^{-1} + \sum_c \gamma_c \mathbf{I}\right)^{-1}, \quad \boldsymbol{\mu}_{\mathbf{f}^p} = \boldsymbol{\Sigma}_{\mathbf{f}^p} \sum_c \gamma_c (\mathbf{q}^p_{c\cdot} - \delta_c)$$

  - The auxiliary variables:

$$p(q^p_{c\tau}|y^p_{c\tau} = k, \ldots) \propto \delta(b_k < q^p_{c\tau} < b_{k+1}) \mathcal{N}(q^p_{c\tau}|f^p_\tau + \delta_c, \gamma_c^{-1})$$

- Gibbs sampling continued:
  - The offset and precision for each clinician:

$$p(\delta_c|\ldots) = \mathcal{N}(\delta_c|\mu_c, \sigma_c^2), \quad p(\gamma_c|\ldots) = \mathcal{G}(a_c, b_c)$$

where:

$$\sigma_c^2 = (1 + N_c)^{-1}, \quad \mu_c = \gamma_c \sigma_c^2 \sum_p \sum_\tau (q_{c\tau}^p - f_\tau^p)$$

and:

$$a_c = \alpha + N_c/2, \quad b_c = \beta + \frac{1}{2} \sum_p \sum_\tau (q_{c\tau}^p - f_\tau^p)^2$$

and N_c is the total number of observations for clinician c.

Inference and Interpretation

- The offset and precision tell us how the clinicians disagree.
  - Low precision indicates a clinician who is very unpredictable (w.r.t the rest)
  - High offset indicates a precision who consistently rates higher or lower than the norm.

Inference and Interpretation

- The offset and precision tell us how the clinicians disagree.
  - Low precision indicates a clinician who is very unpredictable (w.r.t the rest)
  - High offset indicates a precision who consistently rates higher or lower than the norm.
- Identifiability: offset for one clinician fixed to 0.
- Covariance parameter for GP inferred via Metropolis-Hastings (see details in Rogers et al 2013)

Results

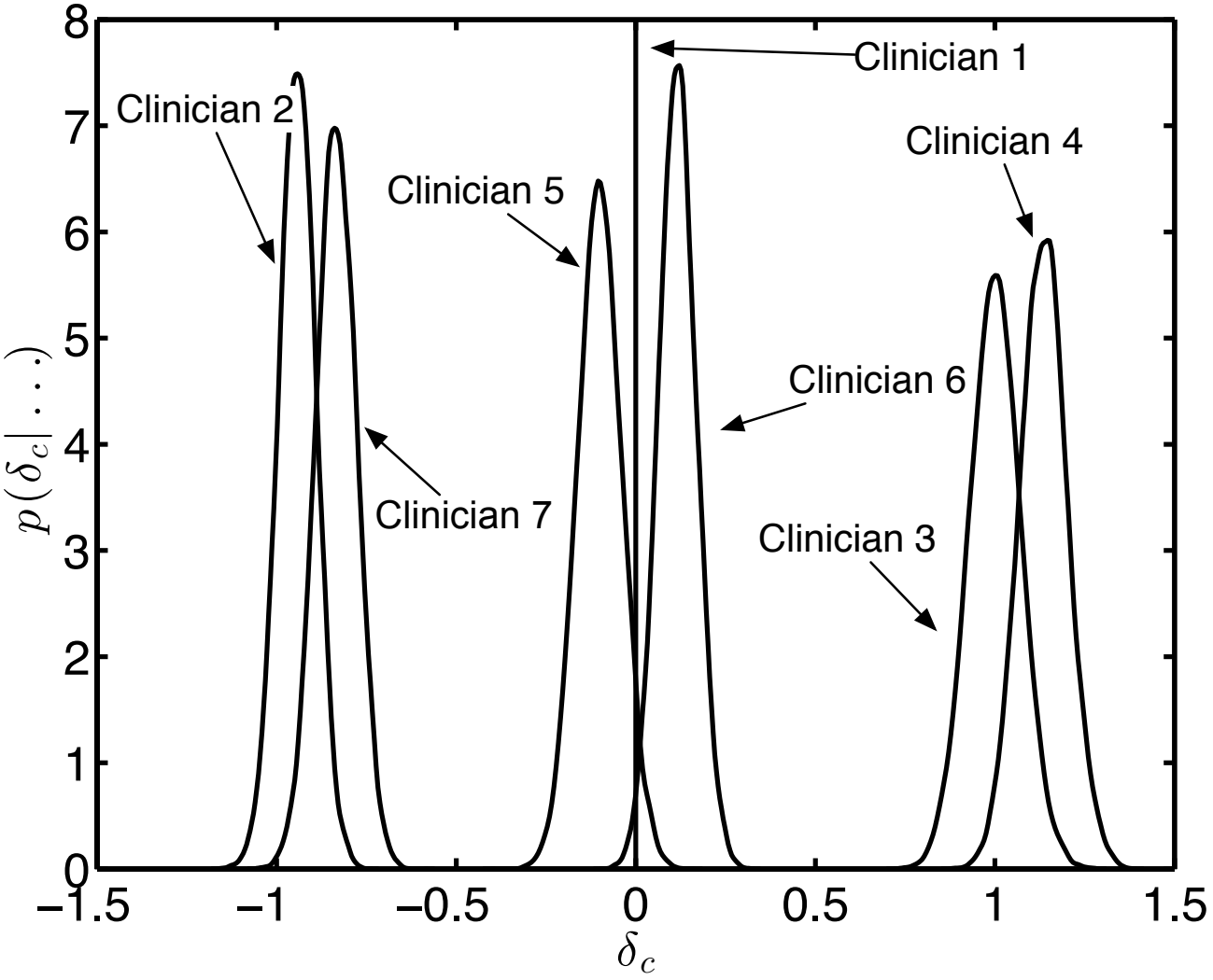
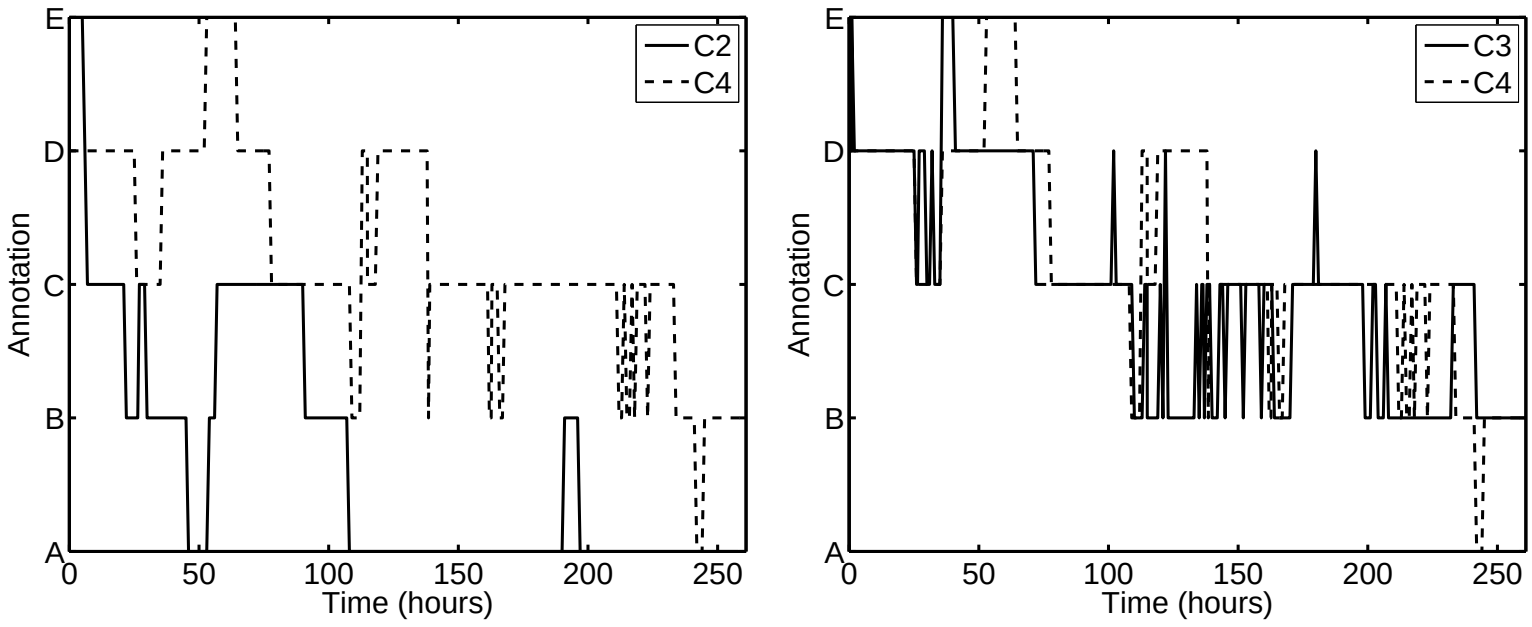


Figure 20 : Marginal offset posteriors

Results



(a) Clinicians 2 and 4

(b) Clinicians 3 and 4

Figure 21 : Inferred offsets make sense on inspection of the data.

Results

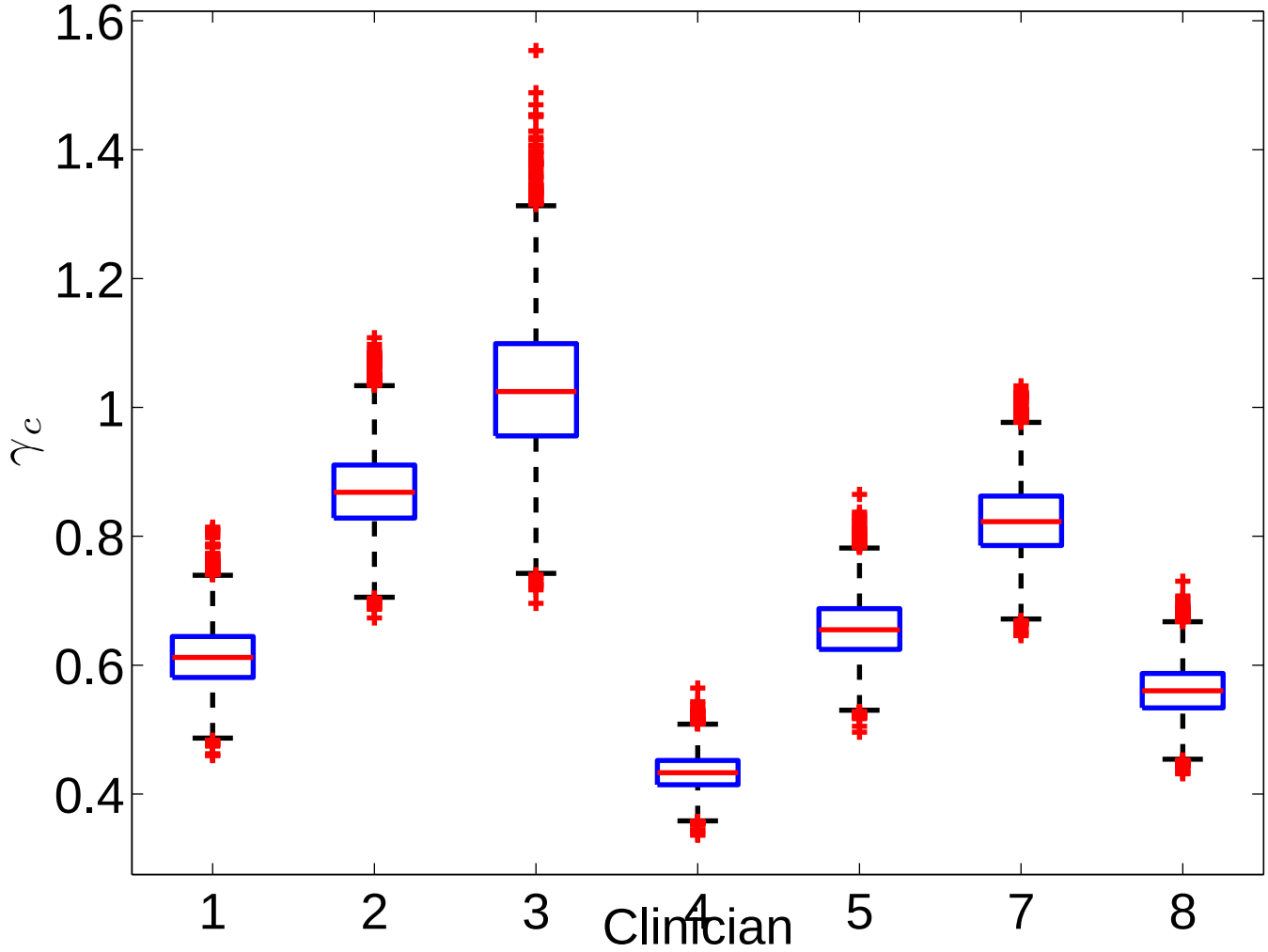
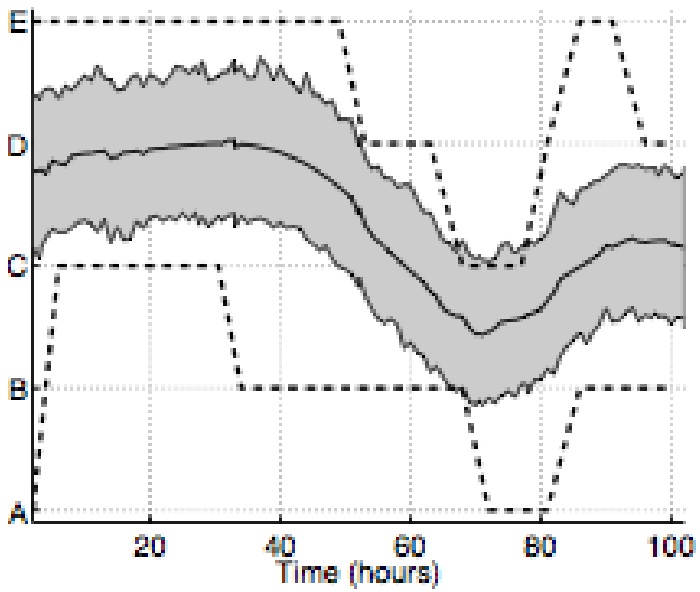


Figure 22 : Marginal precision posteriors. Clinicians 1, 4, and 8 appear to be the least consistent (wrt the majority)

Results

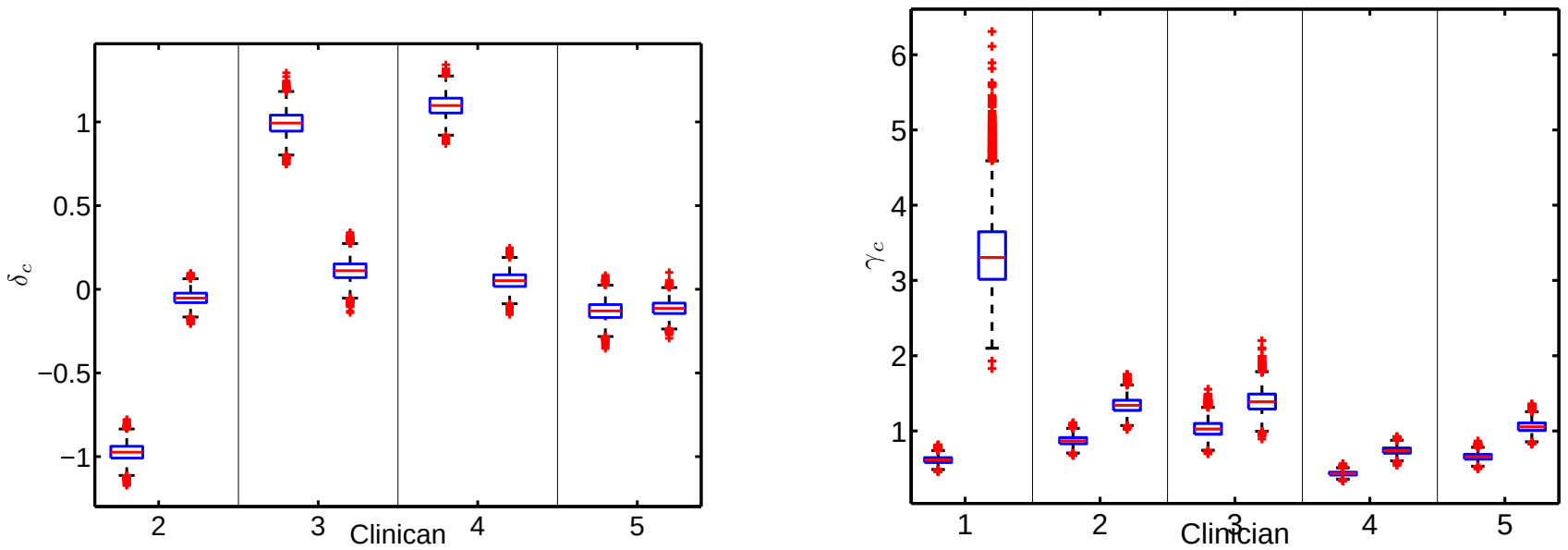


- Posterior health function for one patient
- Shaded area shows boundaries of posterior samples
- Dashed lines show maximum and minimum clinician ratings
  - Note the range: initially patient rated both A and E!

INSIGHT

- After the initial annotation, clinicians went through INSIGHT procedure.
- The goal was to make ratings more consistent.
- If it succeeded, we should see a reduction in offset and increase in precision in the post-INSIGHT data.
- Note: only 5 clinicians remained after INSIGHT

Post-INSIGHT results

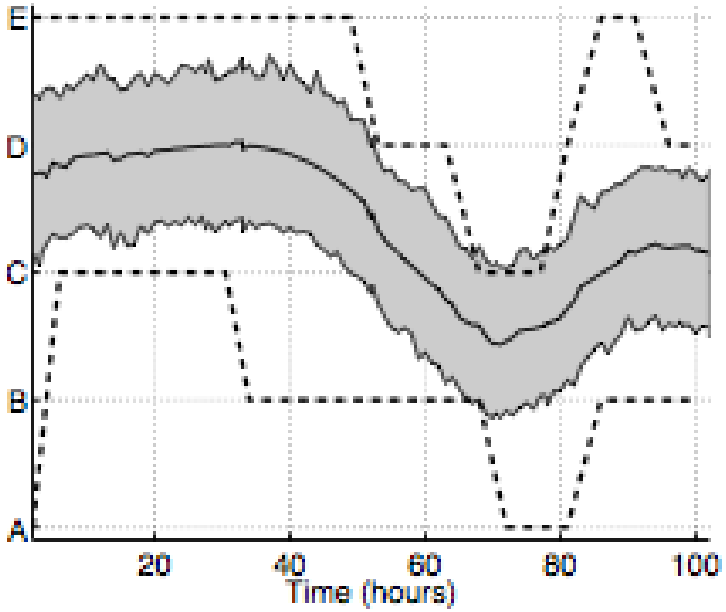


(a) Offsets

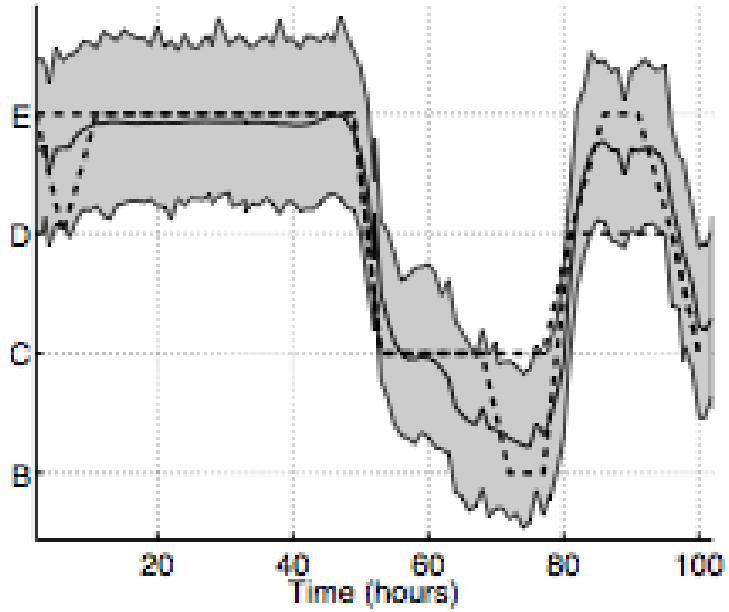
(b) Precisions

Figure 23 : Offsets and precision before and after INSIGHT. Offsets get closer to 0, whilst precision increase suggesting greater agreement amongsth clinicians.

Post-INSIGHT results



(a) Before INSIGHT

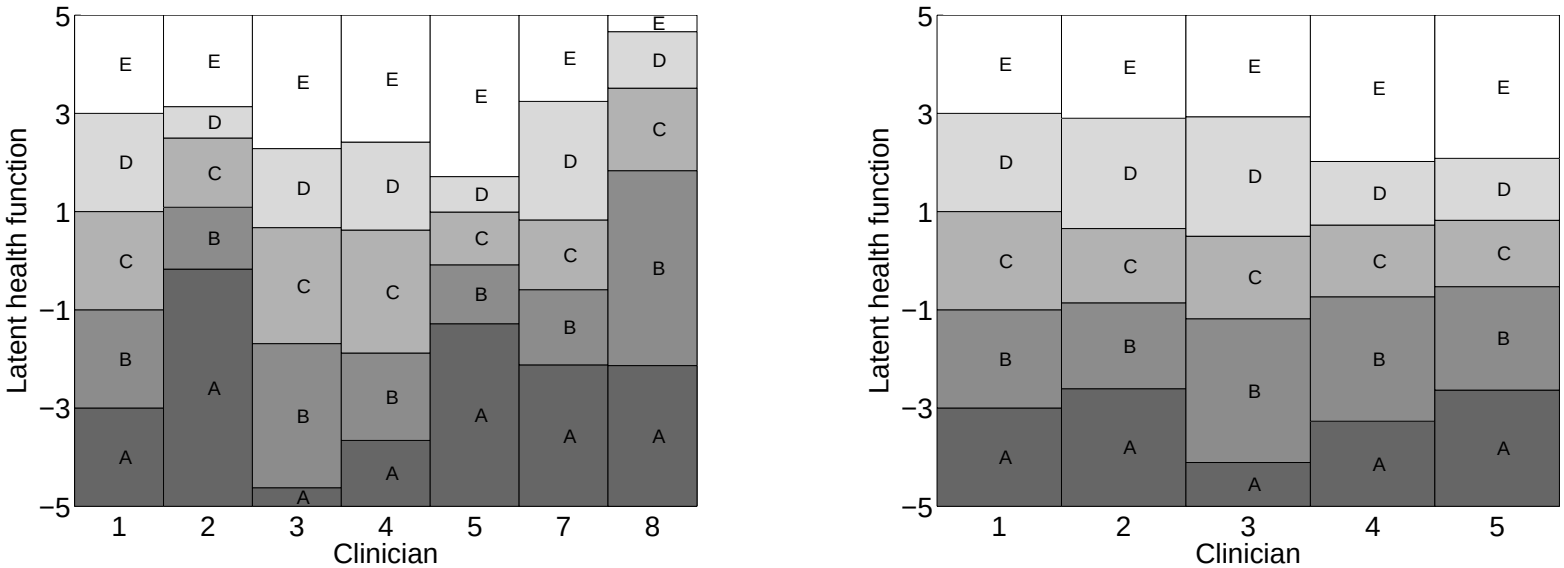


(b) After INSIGHT

- Patient health function before and after INSIGHT
- Less smooth after INSIGHT
- Range of ratings much reduced

Inferring category boundaries

- So far, it has been assumed that all categories are the same size (i.e. the elements of $\mathbf{b}$ are equally spaced).
- We can also infer these (with fixed end-points and $\delta_c = 0$).
- Removes uniform prior assumption over categories.



(a) Before INSIGHT

(b) After INSIGHT

Figure 24 : Posterior mean cateogory boundaries.

Summary and Conclusions

- Model allows us to:
  - learn something about *how* clinicians disagree and how they rate.
  - assess the effectiveness of the INSIGHT procedure.

Summary and Conclusions

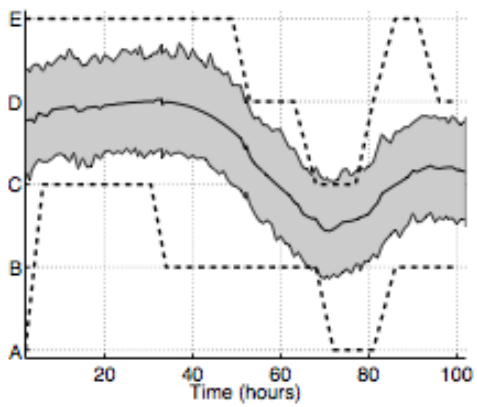
- Model allows us to:
  - learn something about *how* clinicians disagree and how they rate.
  - assess the effectiveness of the INSIGHT procedure.
- GP prior:
  - Flexible
  - Required no parametric assumptions about health function
  - Hyper-parameter (γ) was inferred in the model (could be patient-specific)

Summary and Conclusions

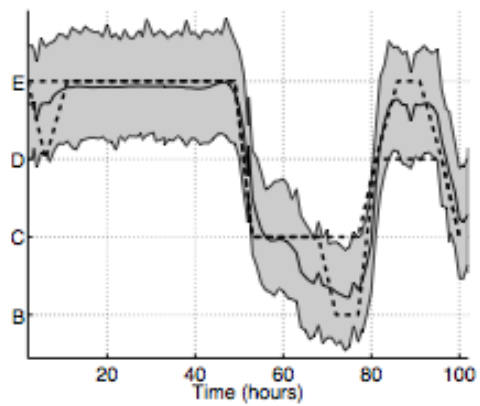
- Model allows us to:
  - learn something about *how* clinicians disagree and how they rate.
  - assess the effectiveness of the INSIGHT procedure.
- GP prior:
  - Flexible
  - Required no parametric assumptions about health function
  - Hyper-parameter (γ) was inferred in the model (could be patient-specific)
- Auxiliary Variable Trick:
  - Not restricted to a standard Gaussian centered on the GP variable.
  - Incorporated offset and precision without causing additional inference challenges.

Future work

- Incorporate measured covariates:
  - HR, BP, etc
  - Predictive model? (would need better ground truth)
- Patient-specific covariance parameters
- Non-stationary covariance parameters
  - Long periods of no change followed by short periods of fast change



(a) Before INSIGHT



(b) After INSIGHT

Lecture 6: Dirichlet Process Priors for Mixture Models

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Mixture Models

- A common strategy when faced with complex multi-modal data is to fit a *mixture model*.
- In general:

$$p(\mathbf{x}) = \sum_{k=1}^{K} P(k)p(\mathbf{x}|k)$$

- Where each component $p(\mathbf{x}|k)$ is some simple density, e.g. Gaussian
- Within this model, we must know K *a-priori*
- Can do inference with Expectation-Maximisation, Variational Bayes, Gibbs Sampling, etc
- Generative process for N data points:
  - For each datapoint, n:
    - Sample a component (k) according to $P(k)$.
    - Sample $\mathbf{x}_n \sim p(\mathbf{x}_n|k)$

Gibbs sampling for mixture models

- Assume that the kth mixture component has parameters θ_k.
- Define binary variables z_{nk} where $z_{nk} = 1$ if nth object is *in* kth component and zero otherwise.
- Define $\pi_k = P(k)$.
- Define prior density on θ_k: $p(\theta_k)$.
- For each iteration:
  - Sample each θ_k from $p(\theta_k|\ldots) \propto p(\theta_k) \prod_n p(\mathbf{x}_n|\theta_k)^{z_{nk}}$
  - For each object n:
    - Remove from its current component.
    - Sample a new component: $P(z_{nk} = 1|\ldots) \propto \pi_k p(\mathbf{x}_n|\theta_k)$

Being Bayesian

- We should treat $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]^T$ as a random variable.
- A suitable prior density is the Dirichlet:

$$p(\boldsymbol{\pi}_k) = \frac{\Gamma(\sum_k \beta)}{\prod_k \Gamma(\beta)} \prod_k \pi_k^{\beta_k - 1}$$

- (from now on, we'll assume $\beta_k = \alpha/K \quad \forall k$)

Being Bayesian

- We should treat $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]^T$ as a random variable.
- A suitable prior density is the Dirichlet:

$$p(\boldsymbol{\pi}_k) = \frac{\Gamma(\sum_k \beta)}{\prod_k \Gamma(\beta)} \prod_k \pi_k^{\beta_k - 1}$$

- (from now on, we'll assume $\beta_k = \alpha/K \quad \forall k$)
- We will also assume that (*a-priori*) the number of objects in each cluster ($c_k = \sum_n z_{nk}$) is multinomial with parameter $\boldsymbol{\pi}$:

$$p(\mathbf{c}|\boldsymbol{\pi}) \propto \prod_k \pi_k^{c_k}$$

Being Bayesian

- We can now compute the posterior density for $\boldsymbol{\pi}$. It's another Dirichlet:

$$p(\boldsymbol{\pi}|\mathbf{c}, \alpha) = \frac{\Gamma(\sum_k \alpha/K + c_k)}{\prod_k \Gamma(\alpha/K + c_k)} \prod_k \pi_k^{\alpha/K + c_k - 1}$$

Being Bayesian

- We can now compute the posterior density for $\boldsymbol{\pi}$. It's another Dirichlet:

$$p(\boldsymbol{\pi}|\mathbf{c},\alpha) = \frac{\Gamma(\sum_k \alpha/K + c_k)}{\prod_k \Gamma(\alpha/K + c_k)} \prod_k \pi_k^{\alpha/K + c_k - 1}$$

- We can now also compute the probablity that some new observation would be placed in class j:

$$\begin{aligned} P(z_{*j} &= 1|\mathbf{c},\alpha) = \int p(z_{*j}=1|\boldsymbol{\pi}) p(\boldsymbol{\pi}|\mathbf{c},\alpha)\ d\boldsymbol{\pi} \\ &= \int \pi_j p(\boldsymbol{\pi}|\mathbf{c},\alpha) \\ &= \frac{c_j + \alpha/K}{\alpha + \sum_k c_k} \end{aligned}$$

- (Need to know that $\Gamma(z+1) = z\Gamma(z)$)

Gibbs sampling again

- Going back to our Gibbs sampling, we can replace π_k with this expression:

$$P(z_{nk} = 1 | \ldots) \propto \frac{c_k + \alpha/K}{\alpha + \sum_j c_j} p(\mathbf{x}_n | \theta_k)$$

- Where the point being sampled shouldn't appear in any c_j (i.e. $\sum_j c_j = N - 1$)

Sampling from the prior

- We can ignore the data $\mathbf{x}_n$ for a while and just sample partitions from this prior:
- Start with N objects, all in one cluster.
- For each iteration:
  - For each object n:
    - Remove from component it is in and re-assign with probability:

$$P(z_{nk} = 1 | \ldots) = \frac{c_k + \alpha/K}{\alpha + \sum_j c_k}$$

Sampling from the prior

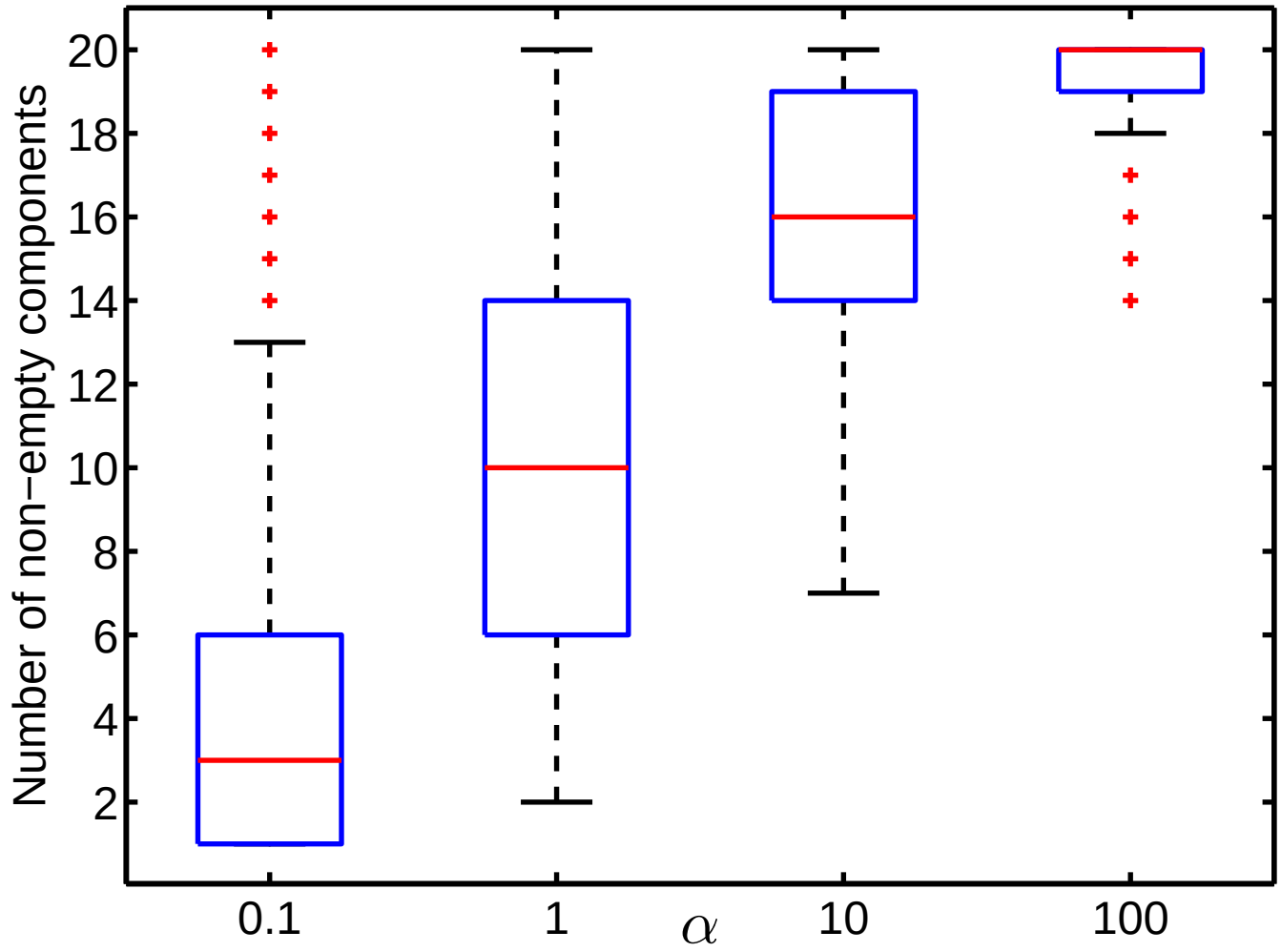


Figure 25 : Number of non-empty components as α is increased. $N = 100$ and $K = 20$. α controls how clustered the data are. Low α gives few populated clusters. Note: could have done this by sampling $\boldsymbol{\pi}$ and then sampling from $\boldsymbol{\pi}$)

$K \to \infty$

- What if we don't want to fix K?
- i.e. set $K = \infty$.

$$P(Z_{nk} = 1 | \ldots) = \frac{c_k + \alpha/K}{\alpha + \sum_j c_j} = \frac{c_k}{\alpha + \sum_j c_j}$$

- This is the probability for one of the *finite* number of clusters that are currently occupied.

$K \to \infty$

- What if we don't want to fix K?
- i.e. set $K = \infty$.

$$P(Z_{nk} = 1 | \ldots) = \frac{c_k + \alpha/K}{\alpha + \sum_j c_j} = \frac{c_k}{\alpha + \sum_j c_j}$$

- This is the probability for one of the *finite* number of clusters that are currently occupied.
- The probability of assigning to one of the *infinite* number of unoccupied clusters must be:

$$P(Z_{nk*} = 1 | \ldots) = 1 - \sum_k \frac{c_k}{\alpha + \sum_j c_j} = \frac{\alpha}{\alpha + \sum_j c_j}$$

$K \to \infty$

- What if we don't want to fix K?
- i.e. set $K = \infty$.

$$P(Z_{nk} = 1 | \ldots) = \frac{c_k + \alpha/K}{\alpha + \sum_j c_j} = \frac{c_k}{\alpha + \sum_j c_j}$$

- This is the probability for one of the *finite* number of clusters that are currently occupied.
- The probability of assigning to one of the *infinite* number of unoccupied clusters must be:

$$P(Z_{nk*} = 1 | \ldots) = 1 - \sum_k \frac{c_k}{\alpha + \sum_j c_j} = \frac{\alpha}{\alpha + \sum_j c_j}$$

- A nice paper describing this: Rasmussen 2000

Sampling from the $K = \infty$ prior

- Start with N objects all in one cluster
- For each iteration:
  - For each object n:
    - Remove from current component (if it leaves an empty component, delete it)
    - Re-assign according to:

$$P(z_{nk} = 1 | \ldots) = \begin{cases} \frac{c_k}{\alpha + \sum_j c_j} & k \text{ currently occupied} \\ \frac{\alpha}{\alpha + \sum_j c_j} & \text{new } k \end{cases}$$

    - if object in a new component, create one.

Sampling from the $K = \infty$ prior

- Start with N objects all in one cluster
- For each iteration:
  - For each object n:
    - Remove from current component (if it leaves an empty component, delete it)
    - Re-assign according to:

$$P(z_{nk} = 1 | \ldots) = \begin{cases} \frac{c_k}{\alpha + \sum_j c_j} & k \text{ currently occupied} \\ \frac{\alpha}{\alpha + \sum_j c_j} & \text{new } k \end{cases}$$

    - if object in a new component, create one.
- The prior we are sampling from is a *Dirichlet Process*

Sampling from the $K = \infty$ prior

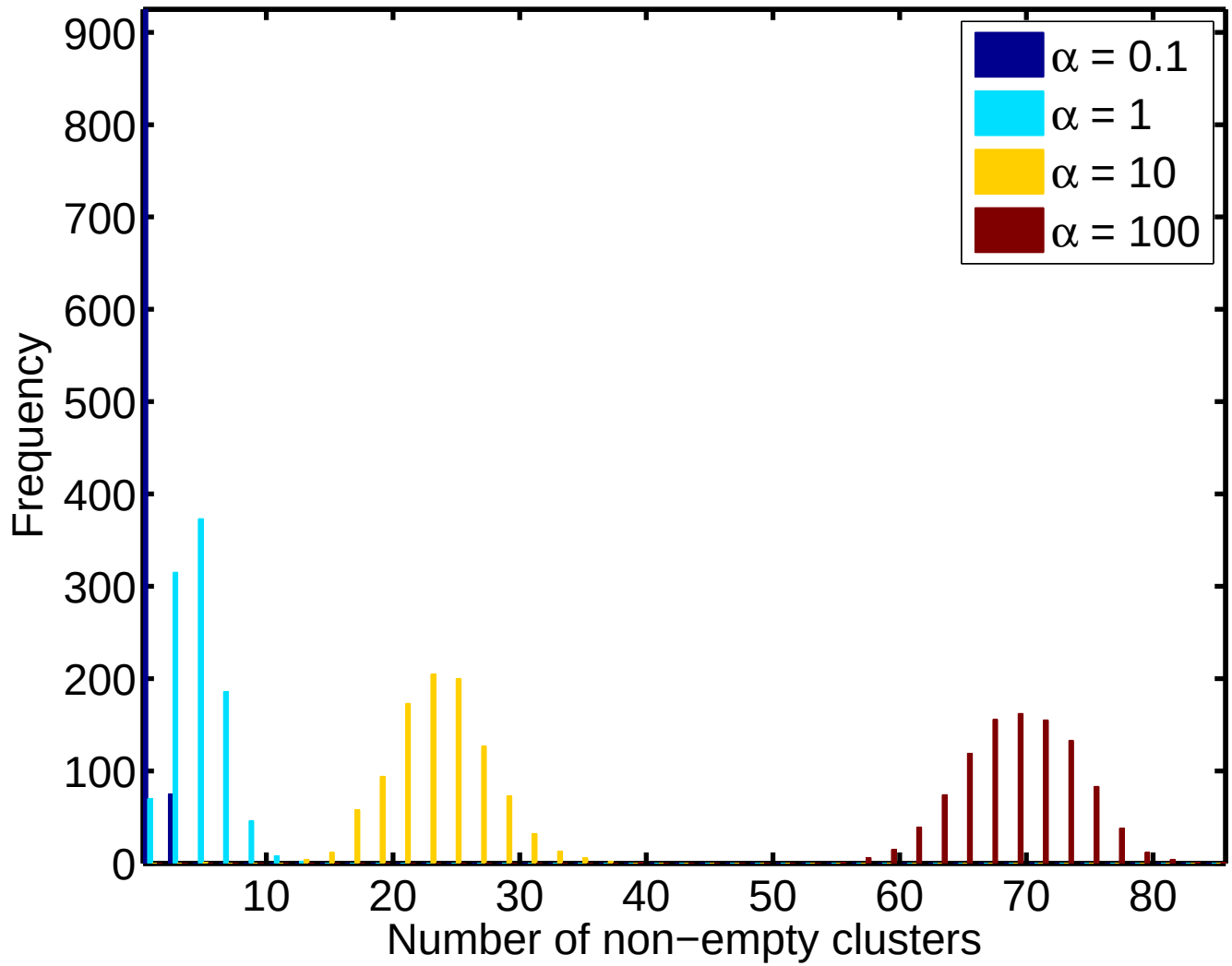


Figure 26 : Number of non empty clusters as α is varied. 1000 samples for $N = 100$ objects.

Including data

- In general, we want to fit the model to objects $\mathbf{x}_n$.
- Assume each component has parameters θ_k.
- Assume a likelihood $p(\mathbf{x}_n|\theta_k)$ and a prior $p(\theta_k)$

Including data

- In general, we want to fit the model to objects $\mathbf{x}_n$.
- Assume each component has parameters θ_k.
- Assume a likelihood $p(\mathbf{x}_n|\theta_k)$ and a prior $p(\theta_k)$
- Gibbs sampling:
  - Given a current clustering $\mathbf{Z}$ and parameters $\theta_1, \ldots, \theta_k$
  - For each object $\mathbf{x}_n$ in each iteration:
    - Remove $\mathbf{x}_n$ from the model (might require deleting a component).
    - Re-assign with probabilities:

$$P(z_{nk} = 1|\mathbf{x}_n, \ldots) \propto \begin{cases} c_k p(\mathbf{x}_n|\theta_k) & k \text{ currently occupied} \\ \alpha \int p(\mathbf{x}_n|\theta) p(\theta)\, d\theta & \text{new } k \end{cases}$$

    - Sample θ_k for each component:

$$p(\theta_k| \ldots) \propto p(\theta_k) \prod_n p(\mathbf{x}_n|\theta_k)^{z_{nk}}$$

Including data

- In general, we want to fit the model to objects $\mathbf{x}_n$.
- Assume each component has parameters θ_k.
- Assume a likelihood $p(\mathbf{x}_n|\theta_k)$ and a prior $p(\theta_k)$
- Gibbs sampling:
  - Given a current clustering $\mathbf{Z}$ and parameters $\theta_1, \ldots, \theta_k$
  - For each object $\mathbf{x}_n$ in each iteration:
    - Remove $\mathbf{x}_n$ from the model (might require deleting a component).
    - Re-assign with probabilities:

$$P(z_{nk} = 1|\mathbf{x}_n, \ldots) \propto \begin{cases} c_k p(\mathbf{x}_n|\theta_k) & k \text{ currently occupied} \\ \alpha \int p(\mathbf{x}_n|\theta) p(\theta)\, d\theta & \text{new } k \end{cases}$$

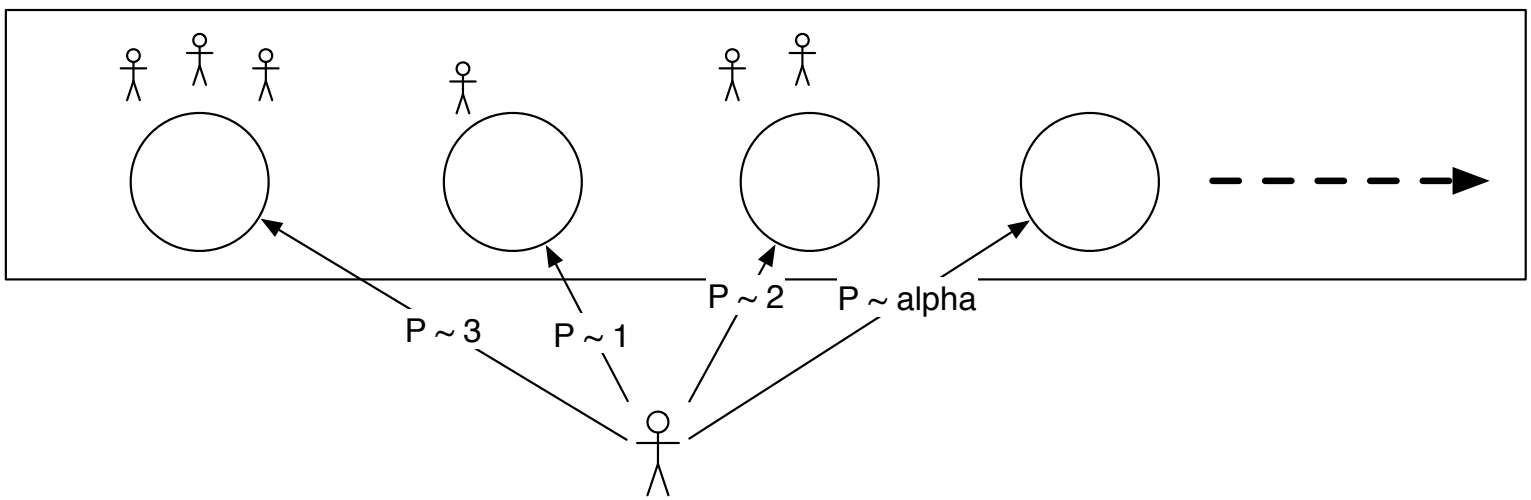
    - Sample θ_k for each component:

$$p(\theta_k| \ldots) \propto p(\theta_k) \prod_n p(\mathbf{x}_n|\theta_k)^{z_{nk}}$$

- If everything conjugate, can integrate θ_k out completely (better convergence).

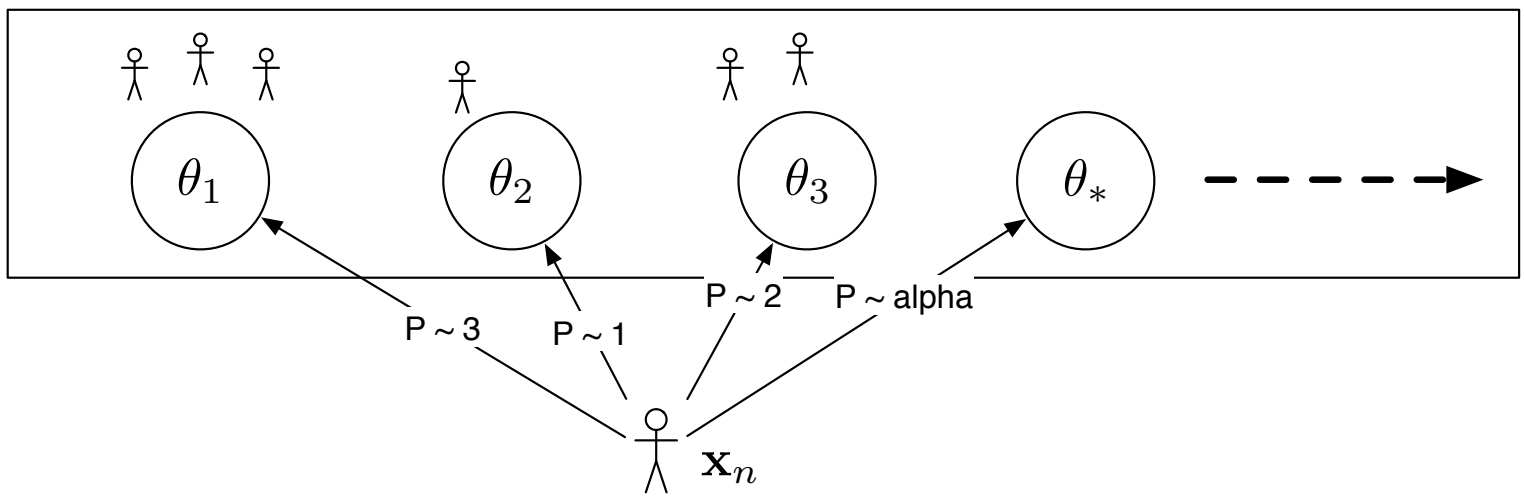
Chinese Restaurant Process (CRP)

- A popular way of thinking about DPs is via the 'Chinese Restaurant Process'
- Prior:
  - N people enter a Chinese Restaurant with infinite tables.
  - The first person sits at the first table.
  - The nth person sits at an occupied table with probability proportional to the number of people already at the table, or a new table with probability proportional to α.



CRP

- With data:
  - Each table has one dish. Table choice also depends on dish preference.
  - Table dishes updated according to preference of people at table (here the analogy starts(!) to get a bit tenuous)



Example

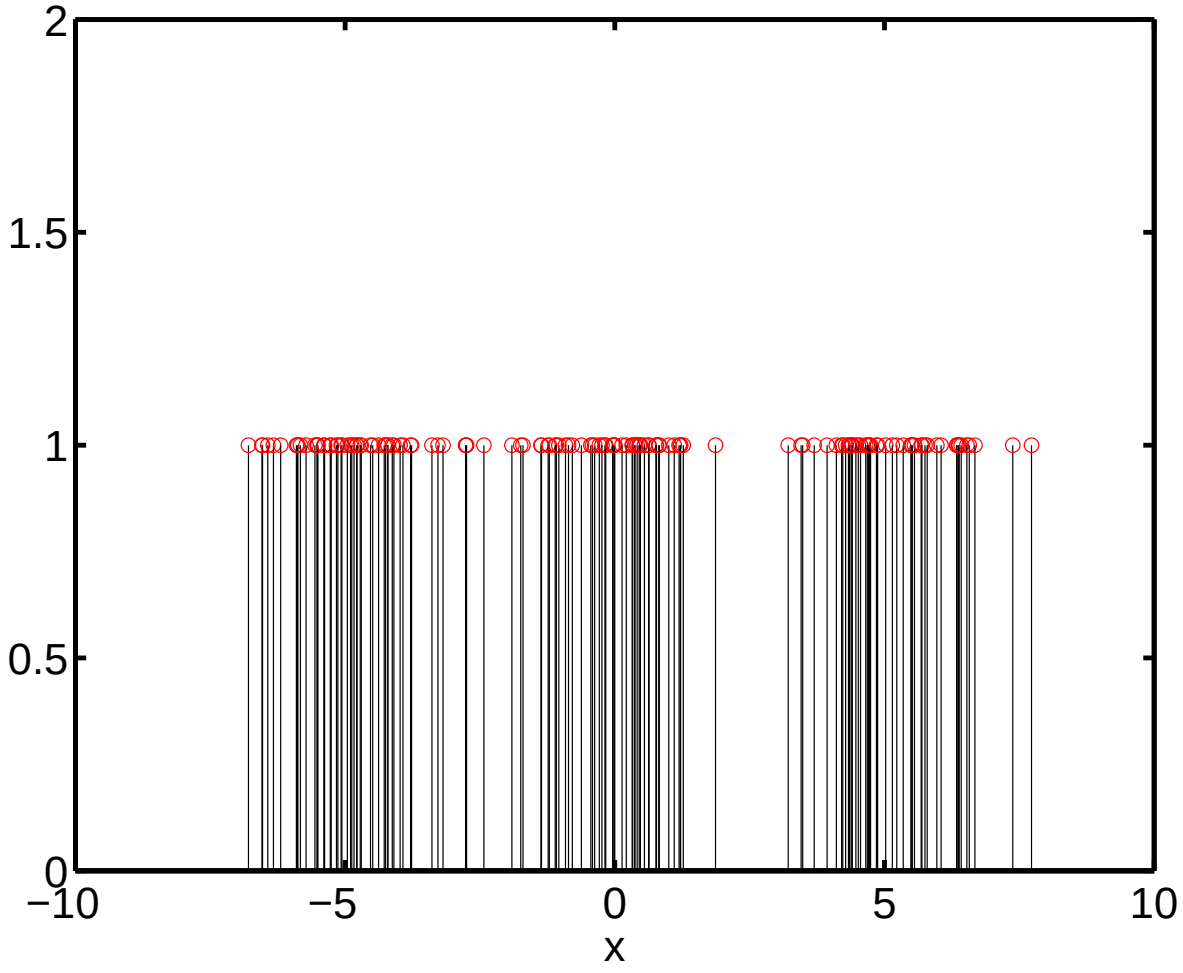


Figure 27 : Example output from CRP with Gibbs sampling. The data has three clusters. Visualising the θ_k is impossible, but we can look at the probablility that two objects are in the same cluster, and the number of clusters. (Data are sorted)

Example

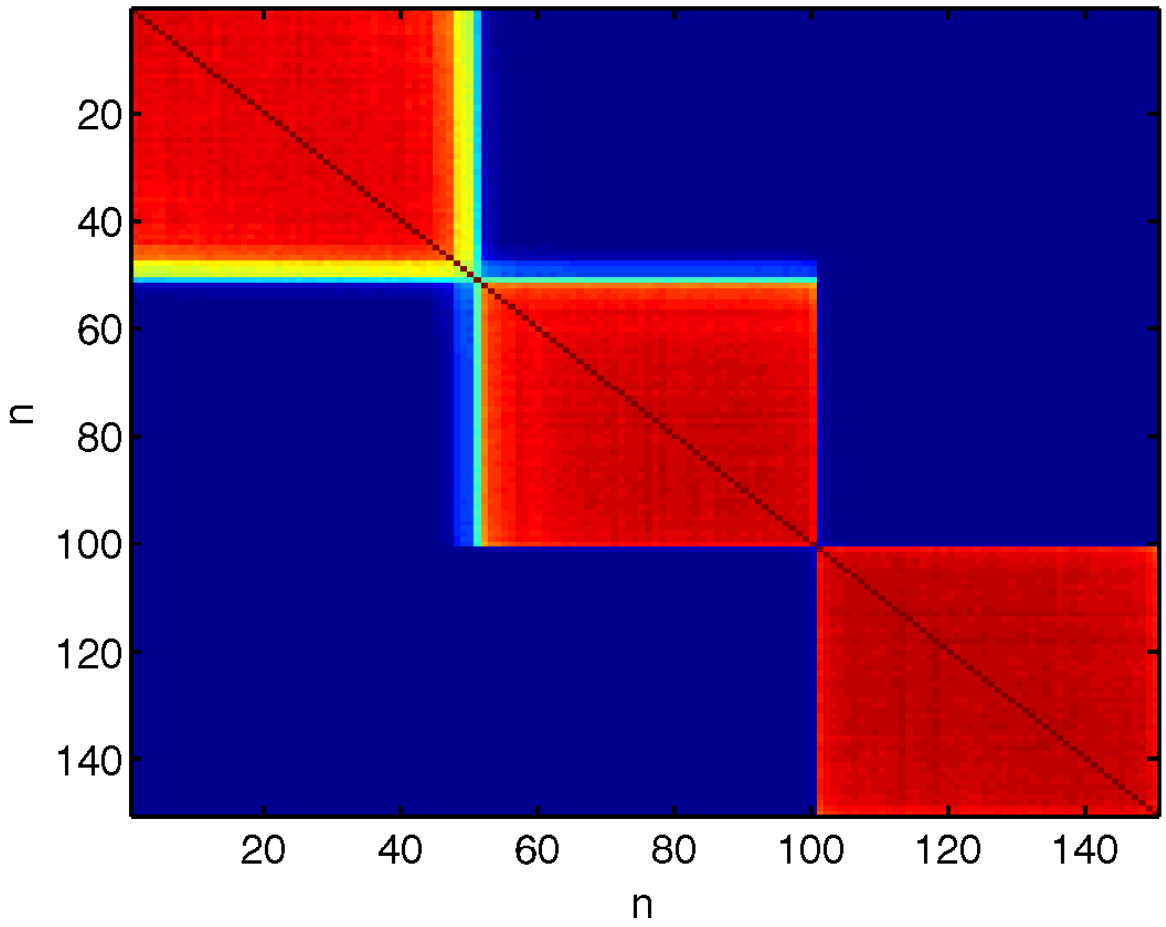


Figure 27 : Example output from CRP with Gibbs sampling. The data has three clusters. Visualising the θ_k is impossible, but we can look at the probablility that two objects are in the same cluster, and the number of clusters. (Data are sorted)

Example

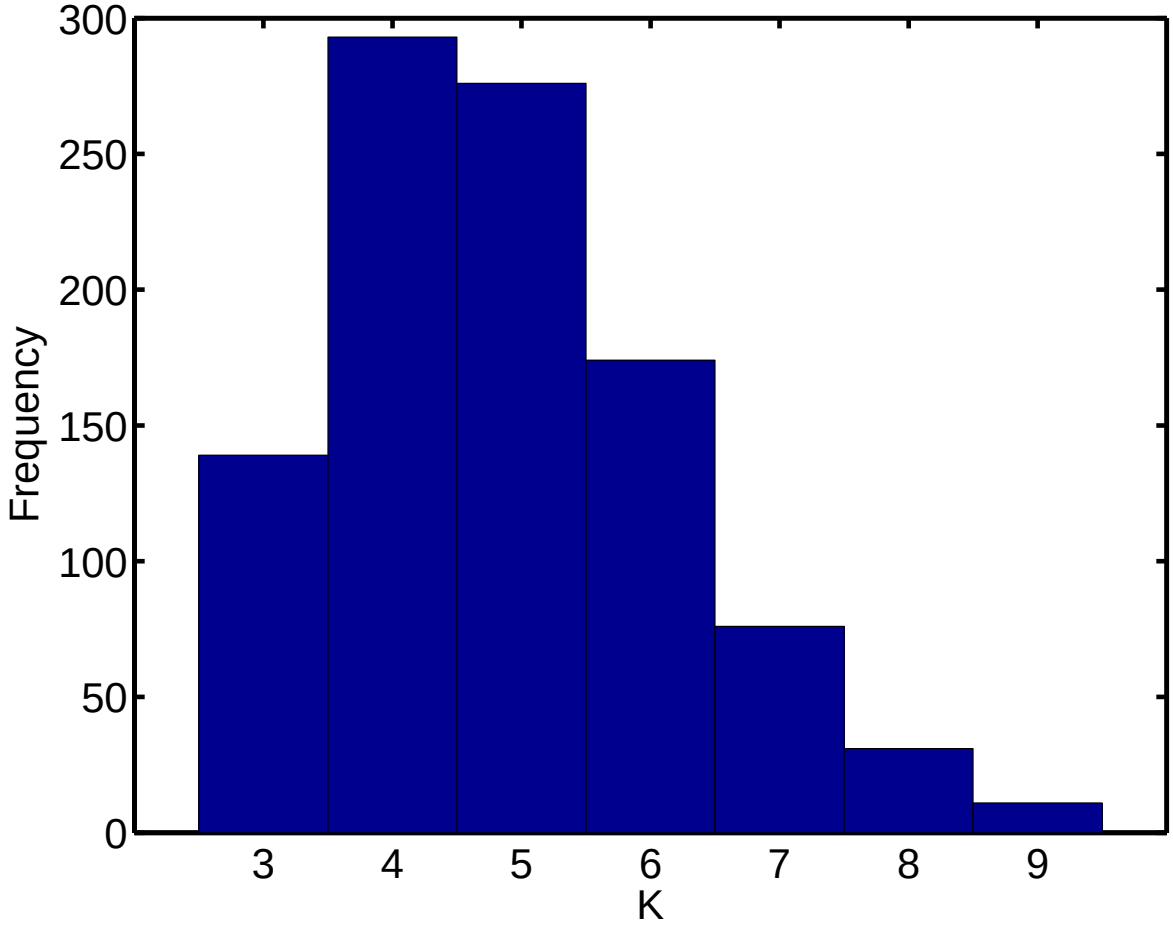


Figure 27 : Example output from CRP with Gibbs sampling. The data has three clusters. Visualising the θ_k is impossible, but we can look at the probablility that two objects are in the same cluster, and the number of clusters. (Data are sorted)

CRP Exercise

TASK [4]

- `dp_script.m` allows you to sample data from a DP mixture and then do Gibbs sampling
- Try varying the concentration parameter α and the priors
- Note: in Octave, you might want to switch off the constant visualisation
- You'll also need `lognormpdf.m` and `clusterlike.m`

Some practical tips

- This works much better with conjugate models.
- Making it work on real problems is hard.
- It's not magic.
  - Avoid specifying how many clusters there are...
  - ...but have to quite precisely speciy what a cluster looks like.
- How to interpret the output?
  - Great for things where the clustering is the *input* to something else

Lecture 7: A mixture model for metabolite peak identification

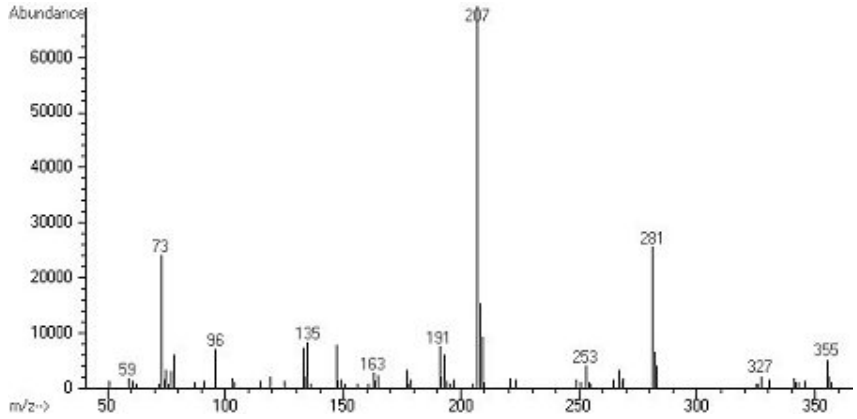
Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Metabolomics

- Metabolome: the set of small molecule metabolites found within an organism.
  - Hormones, sugars, etc
- Gives a reliable picture of the phenotype (Fu et al 2009)
- But metabolites are hard to measure.
- Dominant paradigm is Liquid Chromatography (LC) Mass Spectrometry (MS)

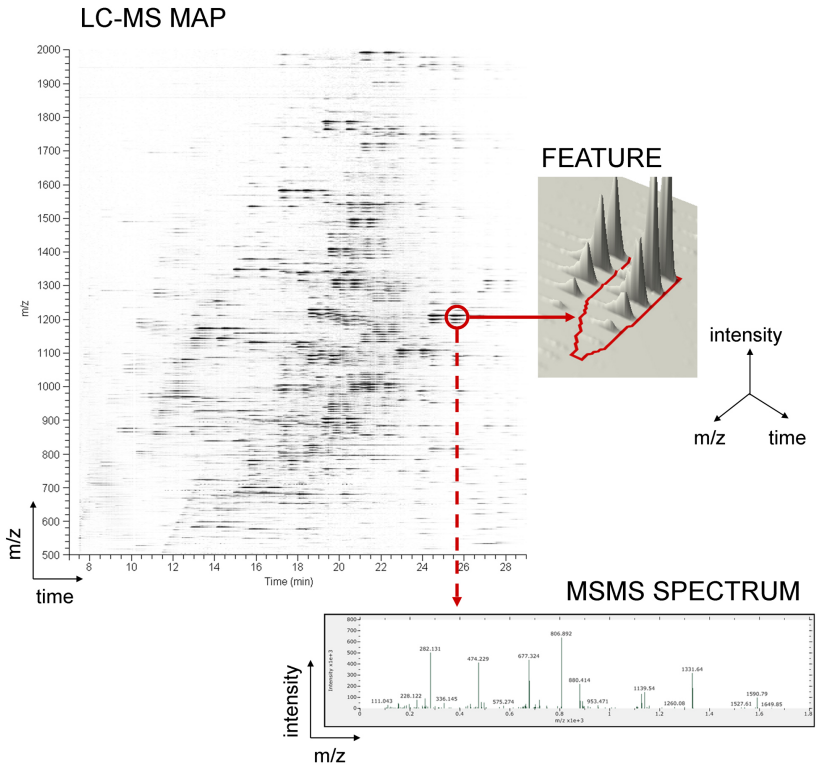
MS



- Output of MS is a set of mass-intensity pairs (peaks).
- Each peak corresponds to one ion.
- Each metabolite can result in many different ions:
  - Different ions (i.e. H^+, K^+)
  - Isotopes
- All have predictable theoretical mass (for particular metabolite)

LC/MS

- Most samples are too complex for a single MS analysis
- First *separate* the sample via LC
- Perform many MS analysis at different Retention Time (RT)



(image from Peltoniemi et. al)

What are the peaks?

- How do we identify things in this 2D image?
- Doing each peak separately (traditional approach; searching mass against a database) leads to many false positives
- Can we use the fact the relationships between all peaks from the same metabolite to improve identification?

Relationships between peaks

- Retention time:
  - All peaks derived from the same metabolite will elute at roughly the same time

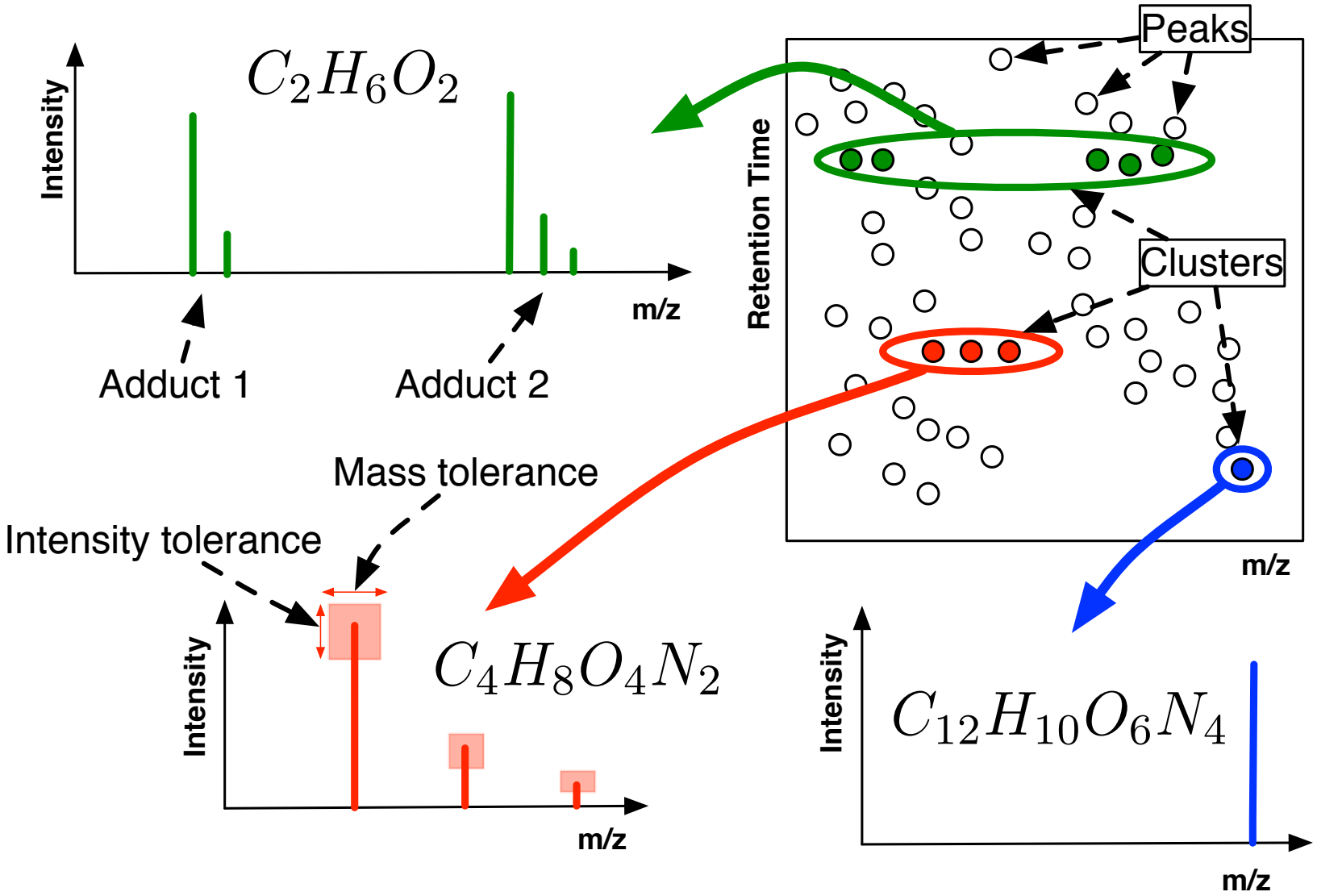
Relationships between peaks

- Retention time:
  - All peaks derived from the same metabolite will elute at roughly the same time
- Mass:
  - For a particular chemical formula we can work out what all the adduct peaks (e.g. M+H) will be
  - No known relationship between different adducts [might be worth investigating]

Relationships between peaks

- Retention time:
  - All peaks derived from the same metabolite will elute at roughly the same time
- Mass:
  - For a particular chemical formula we can work out what all the adduct peaks (e.g. M+H) will be
  - No known relationship between different adducts [might be worth investigating]
- Intensity:
  - The isotope peaks for a particular adduct have well defined relative intensities

MetAssign


$C_2H_6O_2$
Intensity
m/z
Adduct 1
Adduct 2
Peaks
Clusters
Retention Time
m/z
Mass tolerance
Intensity tolerance
$C_4H_8O_4N_2$
Intensity
m/z
$C_{12}H_{10}O_6N_4$
Intensity
m/z

MetAssign

- Use a DP mixture model
- Each cluster is linked to a particular chemical formula (from database)
- Likelihood has terms for:
  - RT: all peaks in cluster must have similar RT
  - Mass: Must be from one of the possible masses for this formula
  - Intensity: peaks within a particular adduct must have correct intensity relationship

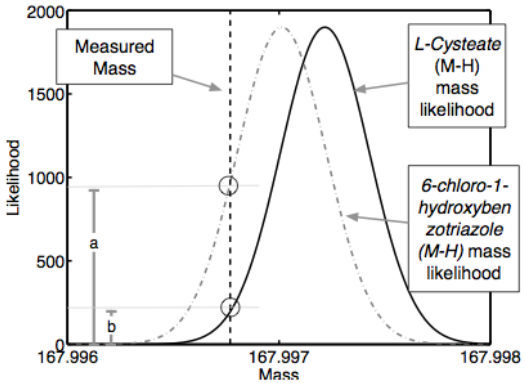
MetAssign

- Gibbs sampling
- At each Gibbs step, each peak is assigned to a cluster and the cluster to a formula
- Average peak to formula assignments over all samples
- Averaging over posterior clusterings

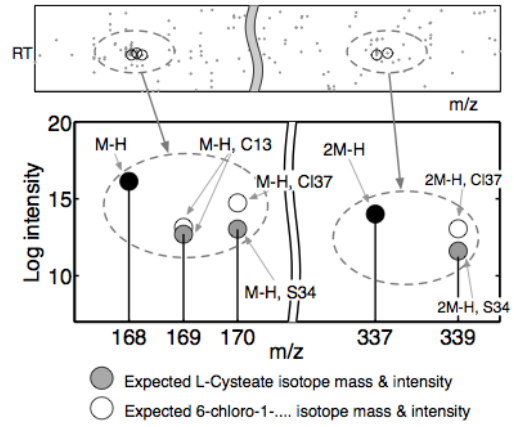
MetAssign

- Gibbs sampling
- At each Gibbs step, each peak is assigned to a cluster and the cluster to a formula
- Average peak to formula assignments over all samples
- Averaging over posterior clusterings
- Filtering:
  - Access to posterior samples allows us to filter out obviously bad assignments (more later)

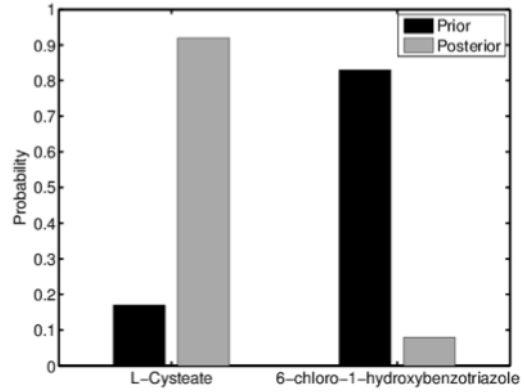
MetAssign



(a) Relative mass likelihoods for two different formulas



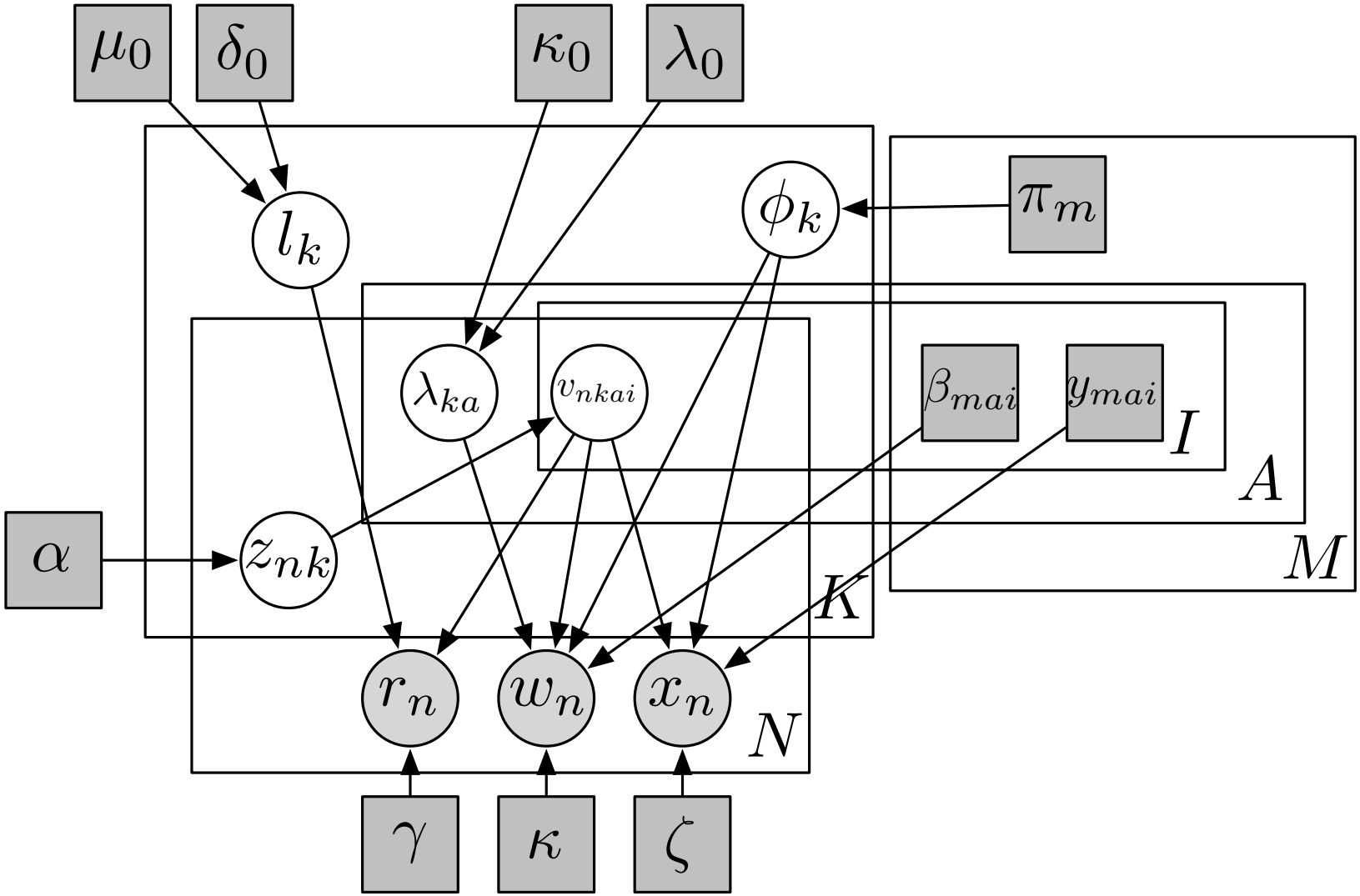
(b) Peaks with similar rentention time included in the cluster with the peak at m/z=168. The circles show the expected intensity values of isotope peaks



(c) The change in probability from the prior to the posterior

- Database matches for each peak can be thought of as prior annotations
- After clustering we have posterior matches
  - Note that by averaging over all clusterings we get posterior assignments for each peak.
  - i.e. we are not interested in one clustering.

MetAssign


μ_0
δ_0
κ_0
λ_0
l_k
ϕ_k
π_m
λ_{ka}
v_{nkai}
β_{mai}
y_{mai}
I
A
M
α
z_{nk}
K
r_n
w_n
x_n
N
γ
κ
ζ

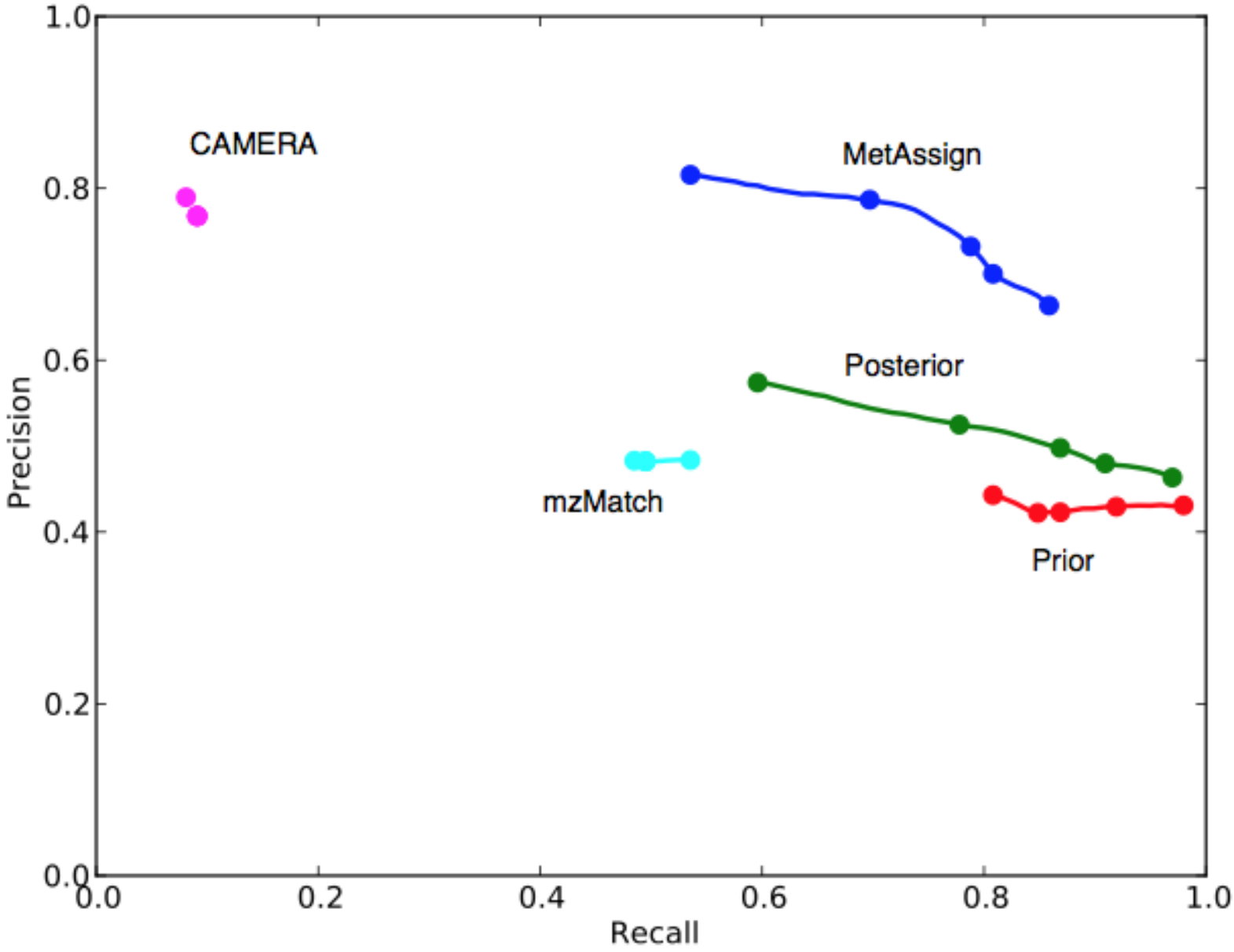
MetAssign

- Model consists of K clusters
- Each cluster is linked to a metabolite from the database (ϕ_k)
- The metabolite links lets us work out what masses (y_{mai}) and intensity relationships (β_{mai}) (for isotopes) we ought to see
- Each cluster has a retention time l_k
- Each adduct (ionisation type) has an intensity (λ_{ka})
- z_{nk} defines cluster membership for peak n
- v_{nkai} defines membership within the cluster

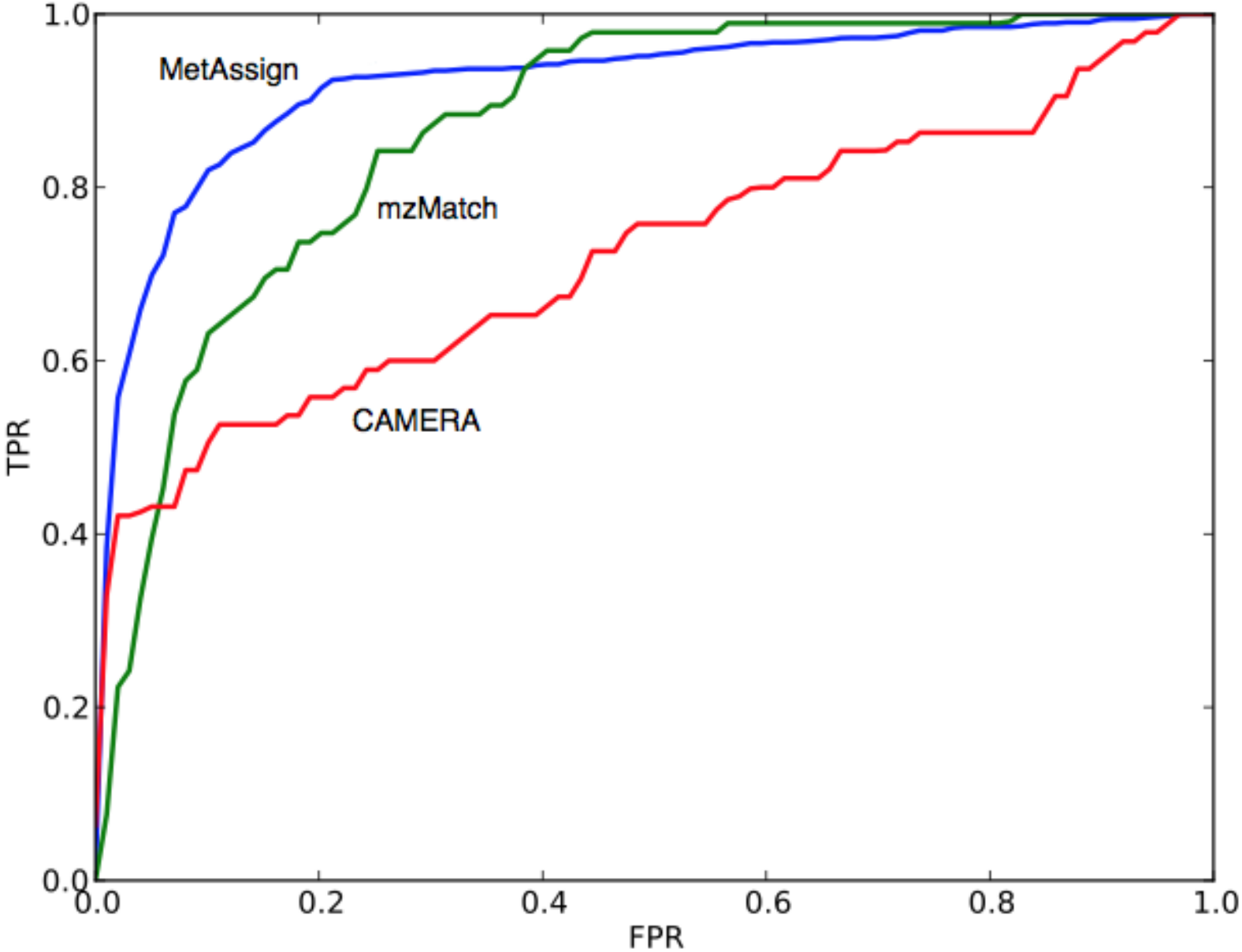
MetAssign: inference

- Gibbs sampling updates are all fairly straightforward (assuming Gaussian noise everywhere)
- Can also include Metropolis-Hastings steps for changing which metabolite a cluster is assigned to
- Proportion of times a peak is assigned to a metabolite (via a cluster) gives posterior probability
- Access to posterior samples lets us do useful things
  - e.g. only consider assignments of peaks if all bigger isotope peaks are present

MetAssign: results



MetAssign: results



Conclusions

- Excellent performance (better than state of the art)
- Dirichlet Process (DP) prior allows us to not fix number of metabolites *a-priori*
- Probabilities are obtained by averaging over the clusterings
- Probabilistic assignments are useful for the experimenters
- Gibbs sampling is easy to implement (although a pain to make efficient)

Problematic isomers

- Unfortunately, some metabolites share chemical formulas but differ in structure
- Identifying them is very hard
  - If they have different RT, current model will put them into different clusters linked to the same formula
  - Can possibly identify them if we include additional information
  - Even slight changes in probabilities would be useful

Additional information 1 - predicted RT

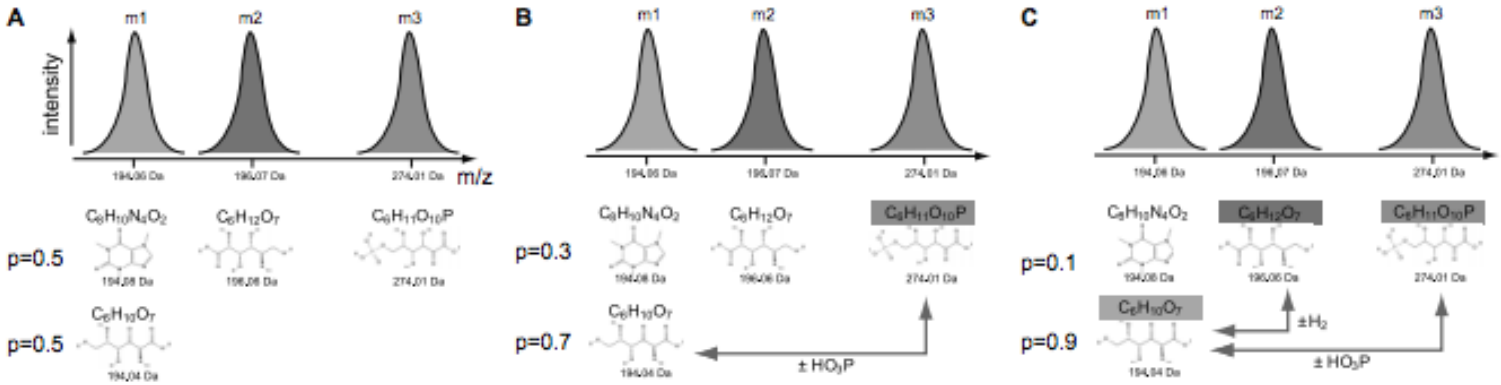
- RT prior is currently fixed across all metabolites
- Easy to extend to be metabolite specific
- RTs can (kind of) be predicted with multi-variate regression
  - Use predicted RTs as prior means for metabolites
  - Tune the prior variance depending on how good the RT prediction is

Additional information 2 - metabolic networks

- Metabolites do not exist in isolation
  - They are the products of reactions involving other metabolites

Additional information 2 - metabolic networks

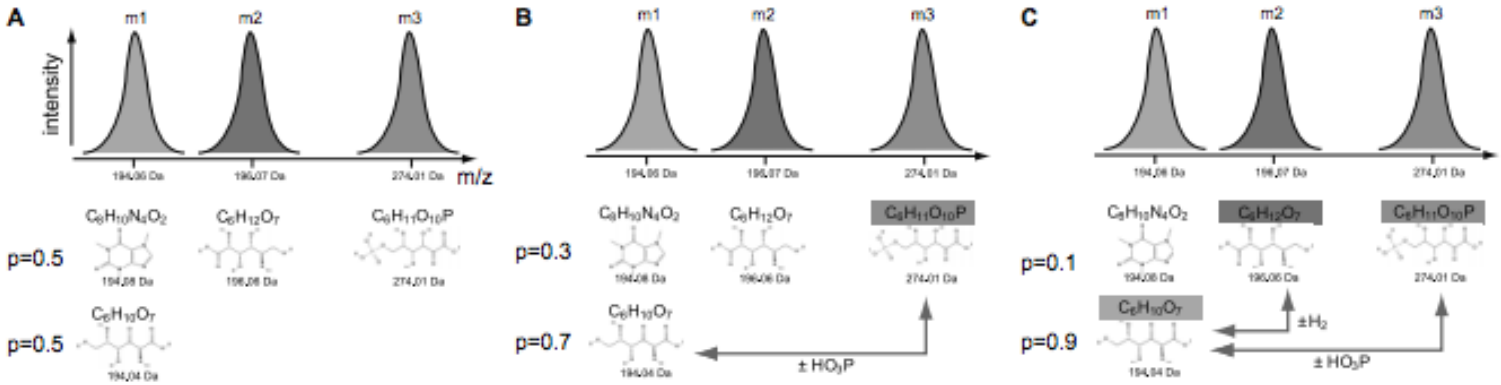
- Metabolites do not exist in isolation
  - They are the products of reactions involving other metabolites
- This structure is useful for identification



(Rogers et. al 2009)

Additional information 2 - metabolic networks

- Metabolites do not exist in isolation
  - They are the products of reactions involving other metabolites
- This structure is useful for identification



(Rogers et. al 2009)

- Can incorporate this as a prior over metabolites to make more connected metabolites mode likely

Future work

- Incorporate predicted RT

Future work

- Incorporate predicted RT
- Incorporate connectivity

Future work

- Incorporate predicted RT
- Incorporate connectivity
- Note: MetAssign currently under revision. When (if) published, search for ‘Rogers Daly Breitling metassign bioinformatics’

Lecture 8: The Hierarchical Dirichlet Process

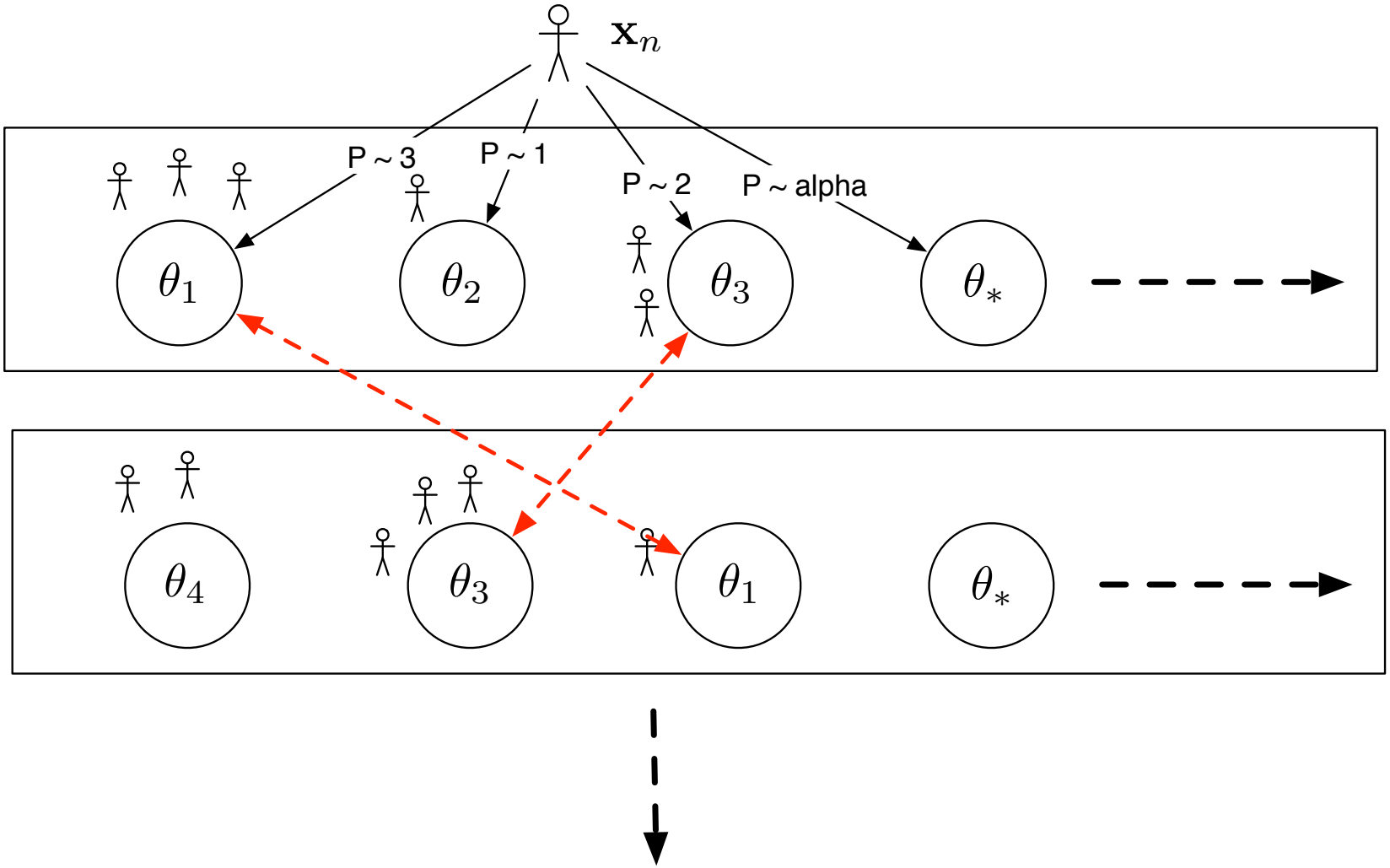
Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

The Hierarchical DP

- Imagine we have > 1 related dataset to cluster, generated by the same process
- Fitting separate mixtures to each results in a loss of information
- The Hierarchical Dirichlet Process (HDP) allows them to be analysed together with shared parameters
  - e.g. datasets clustered individually but cluster parameters (e.g. means) can be *shared*
  - Analogy: The Chinese Restaurant Franchise
  - Described in Teh et. al

The Chinese Restaurant Franchise



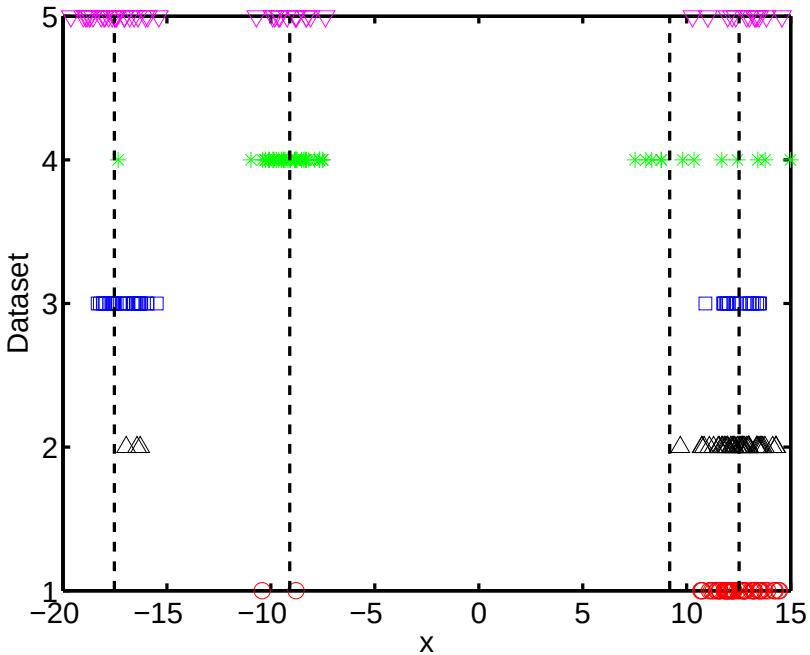
The Chinese Restaurant Franchise

- Gibbs sampling is very similar to the single restaurant case
- Each object is re-assigned to a table with probabilities (i indexes datasets):

$$P(z_{ink} = 1 | \ldots) \propto \begin{cases} n_{ik} p(\mathbf{x}_{in} | \theta_k) & \text{for current table} \\ \alpha p(\mathbf{x}_{in}) & \text{for new } k \end{cases}$$

- $p(\mathbf{x}_{in})$ is computed by marginalising all possible values for the parameters at the new tables.
  - These include values used for current tables (with prior proportional to the number of tables they're used at) and a completely new value (with prior proportional to, say, γ).
  - i.e. There are DPs for assignment of objects to tables and assignment tables to parameters.

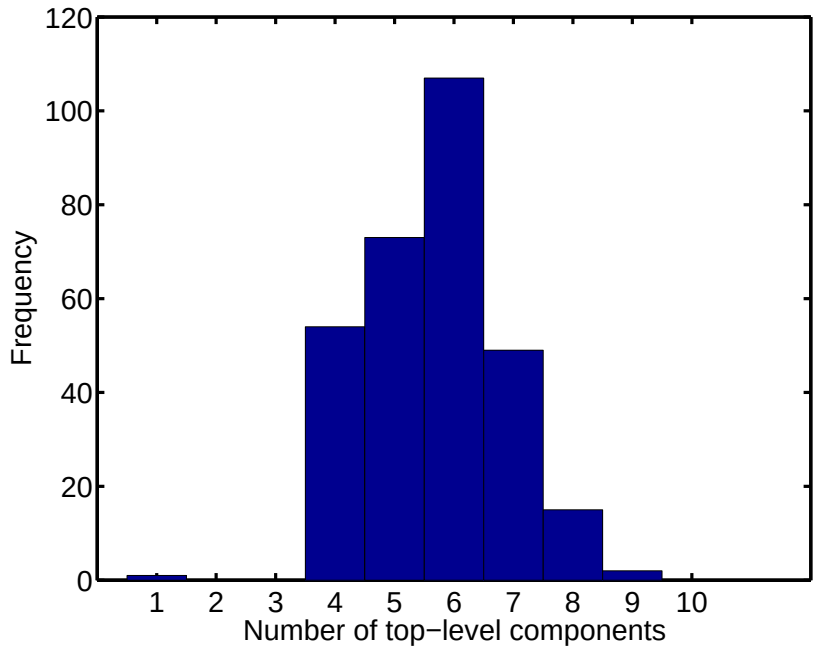
Data sampled from a HDP



- 5 datasets
- 4 top level components (dashed line)
- Each data has data from a subset of components

HDP inference example

- 300 posterior samples.
- Does pretty good job.
- At last sample, posterior means over top-level components were: 12.5215, -17.0304, -9.0535, 8.7857, -17.5636, -16.12
- True: [12.5185,-9.0909,-17.5295,9.1839]



- Note that mixing would be improved by also re-sampling assignment of clusters to top components.

HDP Exercise

TASK [5]

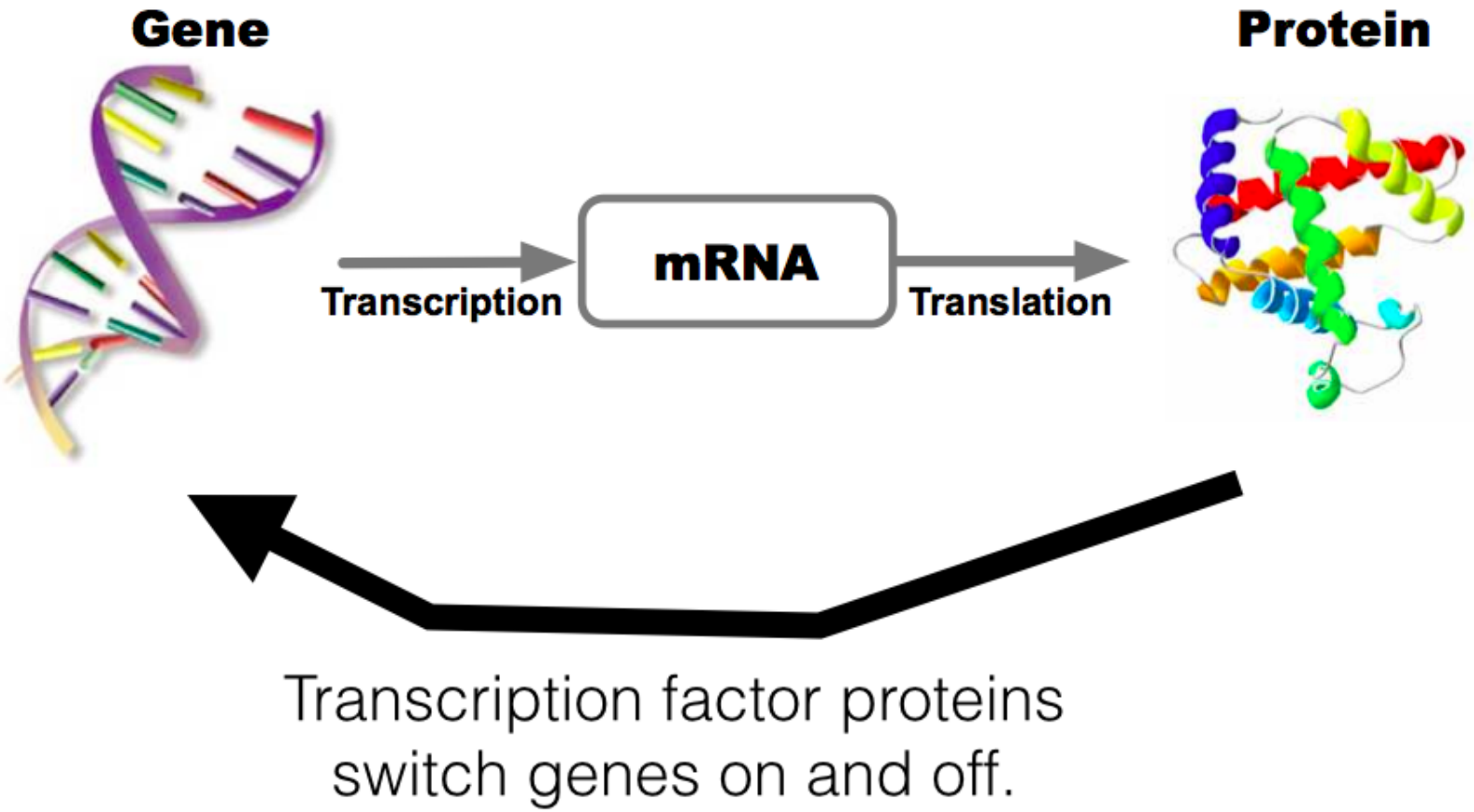
- `HDP_task.m` allows you to sample data from a HDP and then perform inference
- Try varying α and γ to see the effect, and the number of datasets
- Switch `resampletables` on and off to see the effect on the number of components
- You'll also need `lognormpdf.m`, `clusterlike.m`, `hdp_sample.m`

Lecture 9: Infinte factorisaion of multiple non-parametric views

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

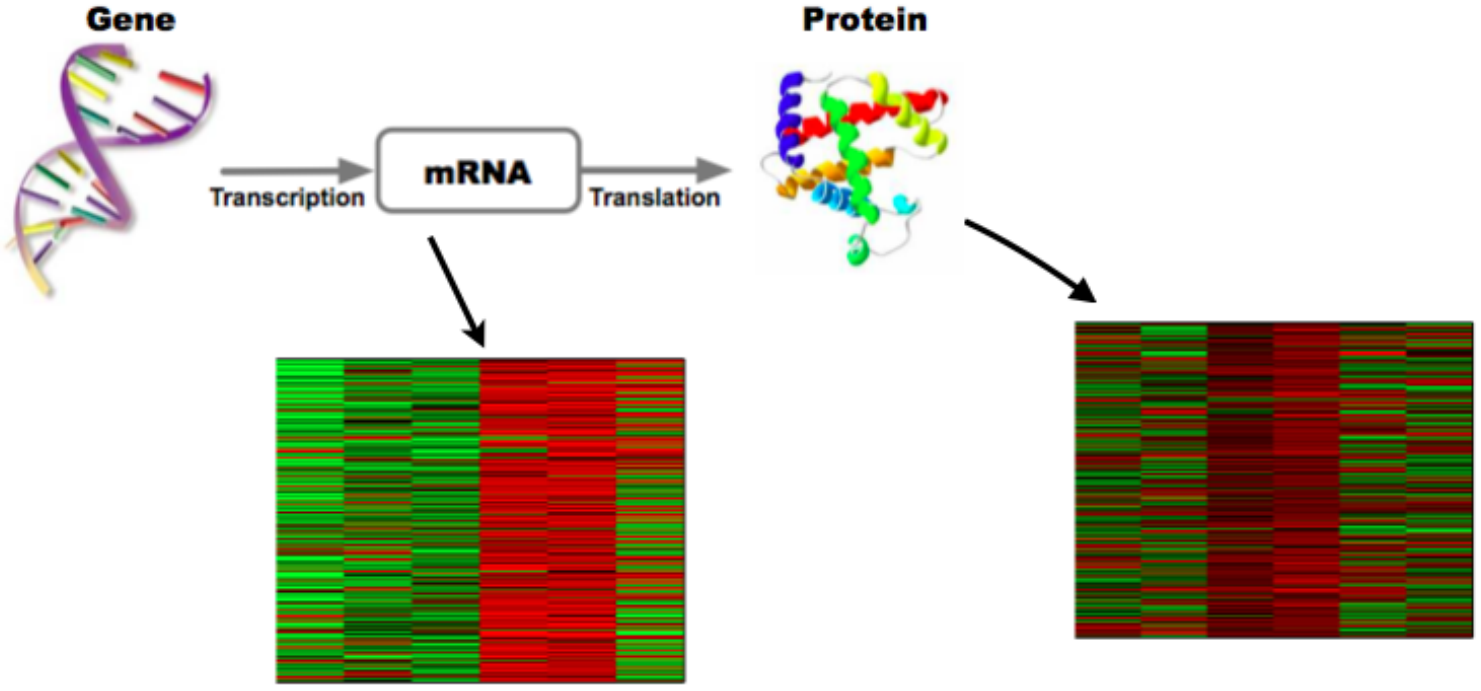
Introduction



- How related are mRNA and protein?
- **Where is the external control?**

Data

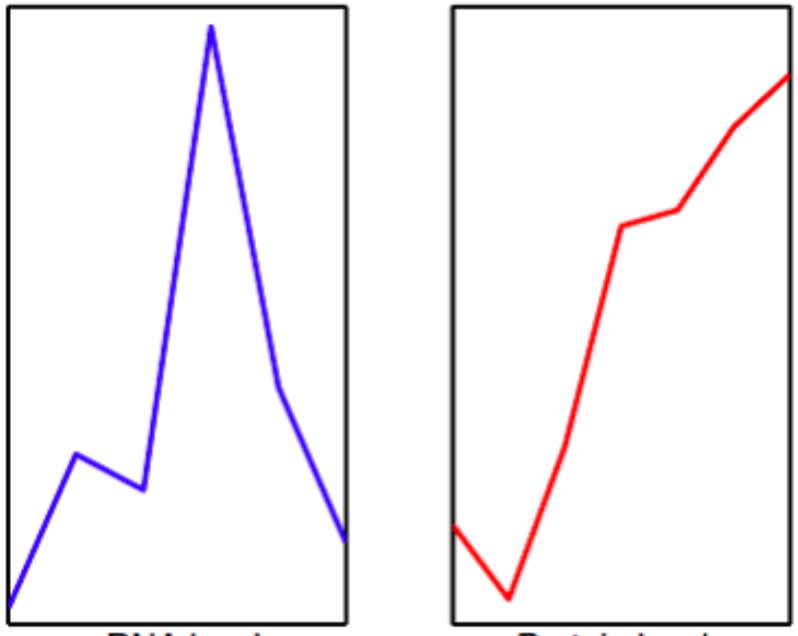
- mRNA and protein time series for $\sim$ 500 genes



mRNA & protein for ~500 genes

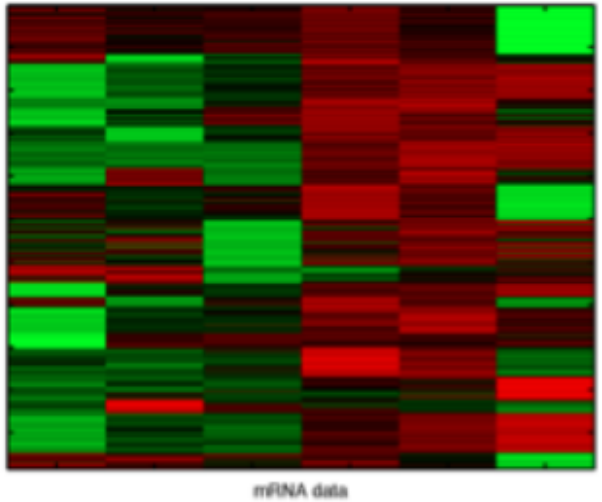
(rows in matrix correspond to one another)

Most don't look correlated

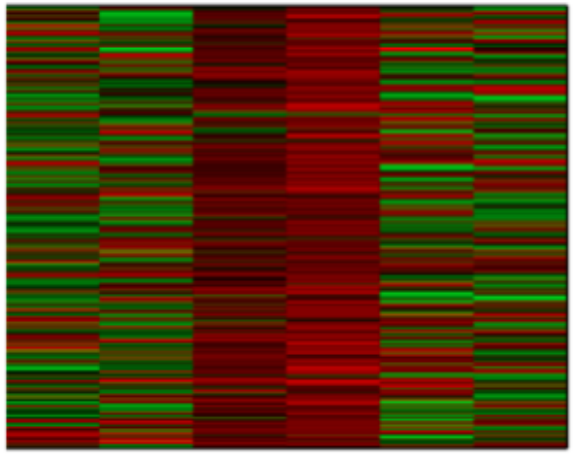


- Most don't look correlated.
  - Time delays? Saturation? Decay rates? Post-transcriptional control?

Non-parametric relationships



Cluster genes by mRNA



Cluster genes by protein

- Use clustering to define similarity.
- If genes A,B,C and D cluster together on both sides, then profiles are similar
  - They are controlled by similar processes

Generative coupled mixture model

- In Rogers et. al 2008 we developed a generative linked cluster model
- Prior membership of gene n in proteomic cluster j was dependent on assignment of mRNA to cluster k:

$$P(z_{nj} = 1, z_{nk} = 1) = P(z_{nj} = 1 | z_{nk} = 1) P(z_{nk} = 1)$$

Generative coupled mixture model

- In Rogers et. al 2008 we developed a generative linked cluster model
- Prior membership of gene n in proteomic cluster j was dependent on assignment of mRNA to cluster k:

$$P(z_{nj} = 1, z_{nk} = 1) = P(z_{nj} = 1 | z_{nk} = 1) P(z_{nk} = 1)$$

- Note that another obvious way to factorise this joint it to assume independence:

$$P(z_{nj} = 1, z_{nk} = 1) = P(z_{nj} = 1) P(z_{nk} = 1)$$

- I.e. cluster them separately

Generative coupled mixture model

- In Rogers et. al 2008 we developed a generative linked cluster model
- Prior membership of gene n in proteomic cluster j was dependent on assignment of mRNA to cluster k:

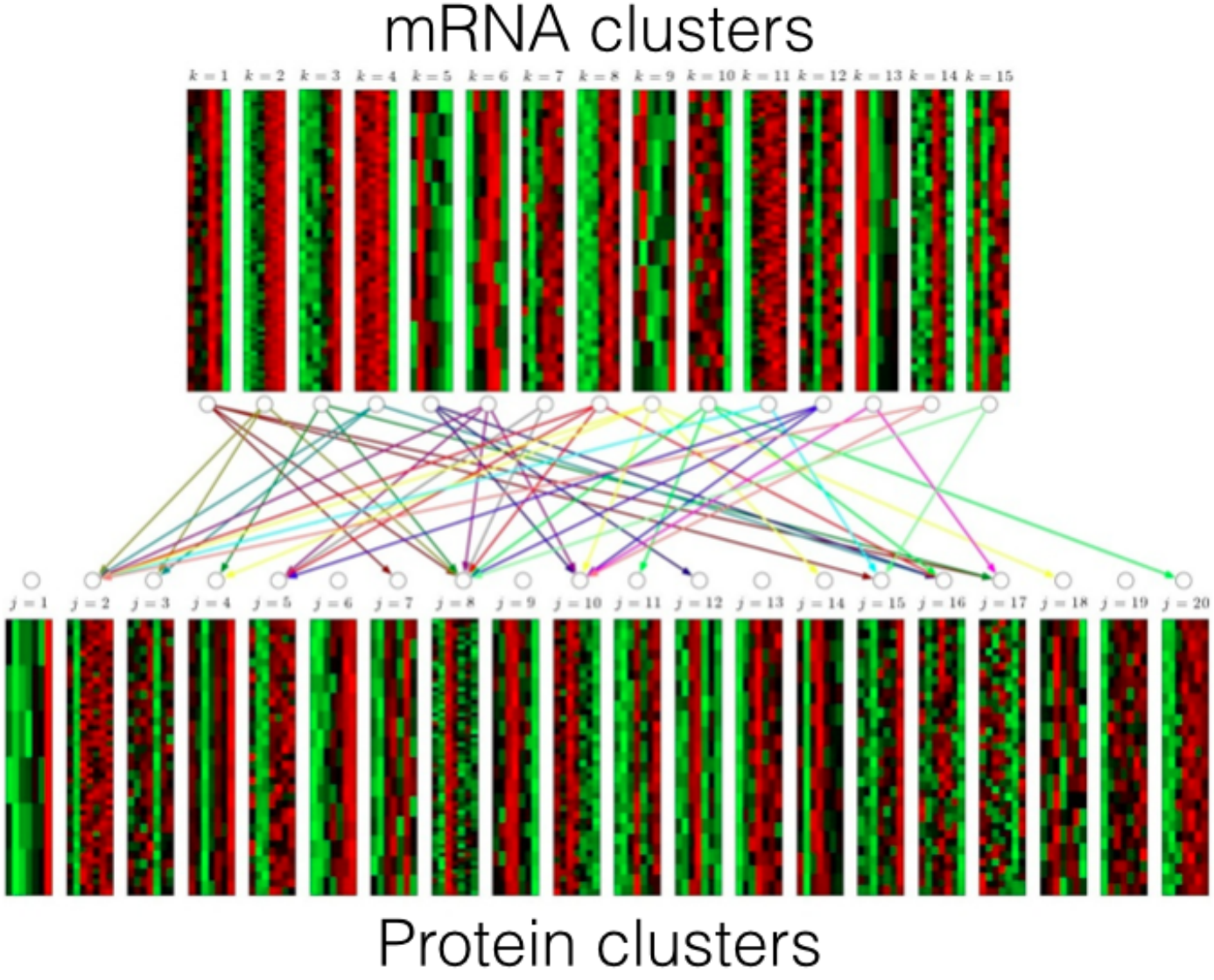
$$P(z_{nj} = 1, z_{nk} = 1) = P(z_{nj} = 1 | z_{nk} = 1) P(z_{nk} = 1)$$

- Note that another obvious way to factorise this joint it to assume independence:

$$P(z_{nj} = 1, z_{nk} = 1) = P(z_{nj} = 1) P(z_{nk} = 1)$$

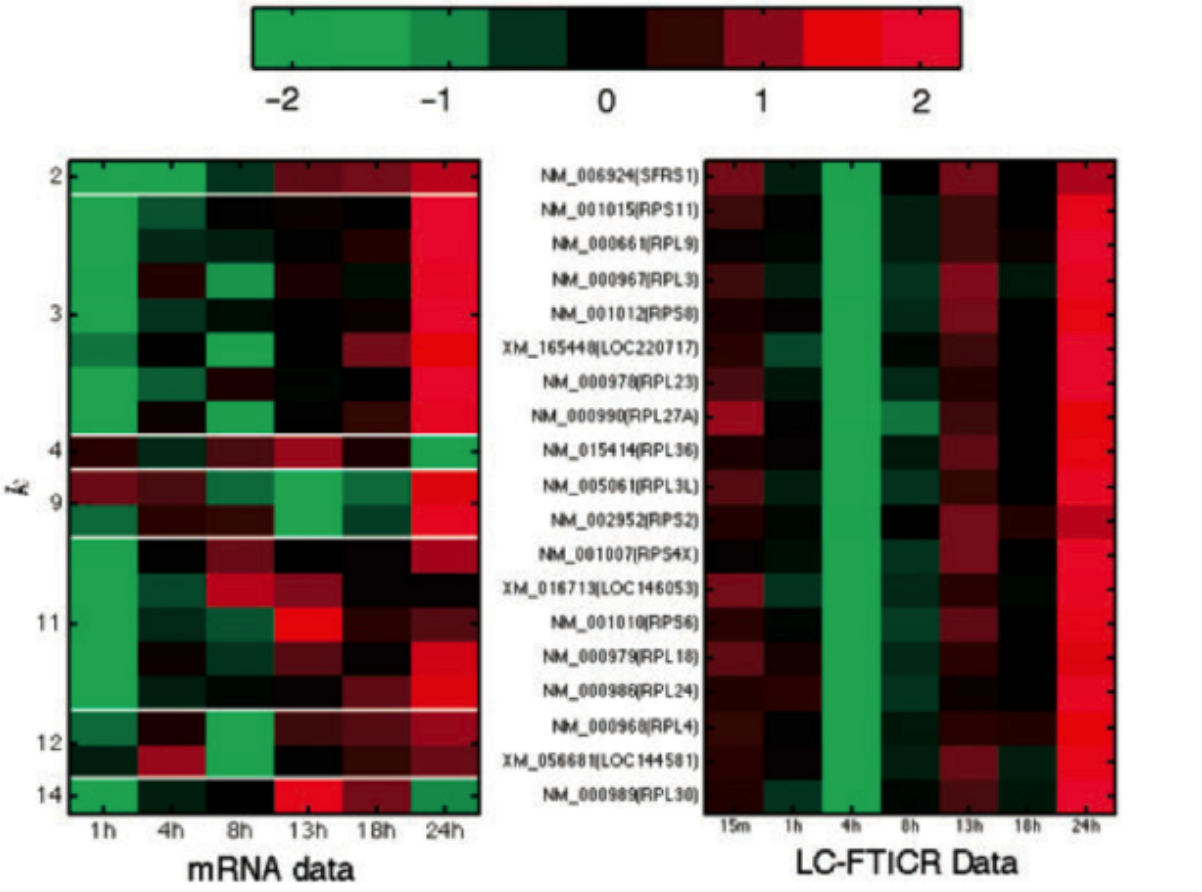
- I.e. cluster them separately
- When we do inference, we can find the links between clusters

Lots of links



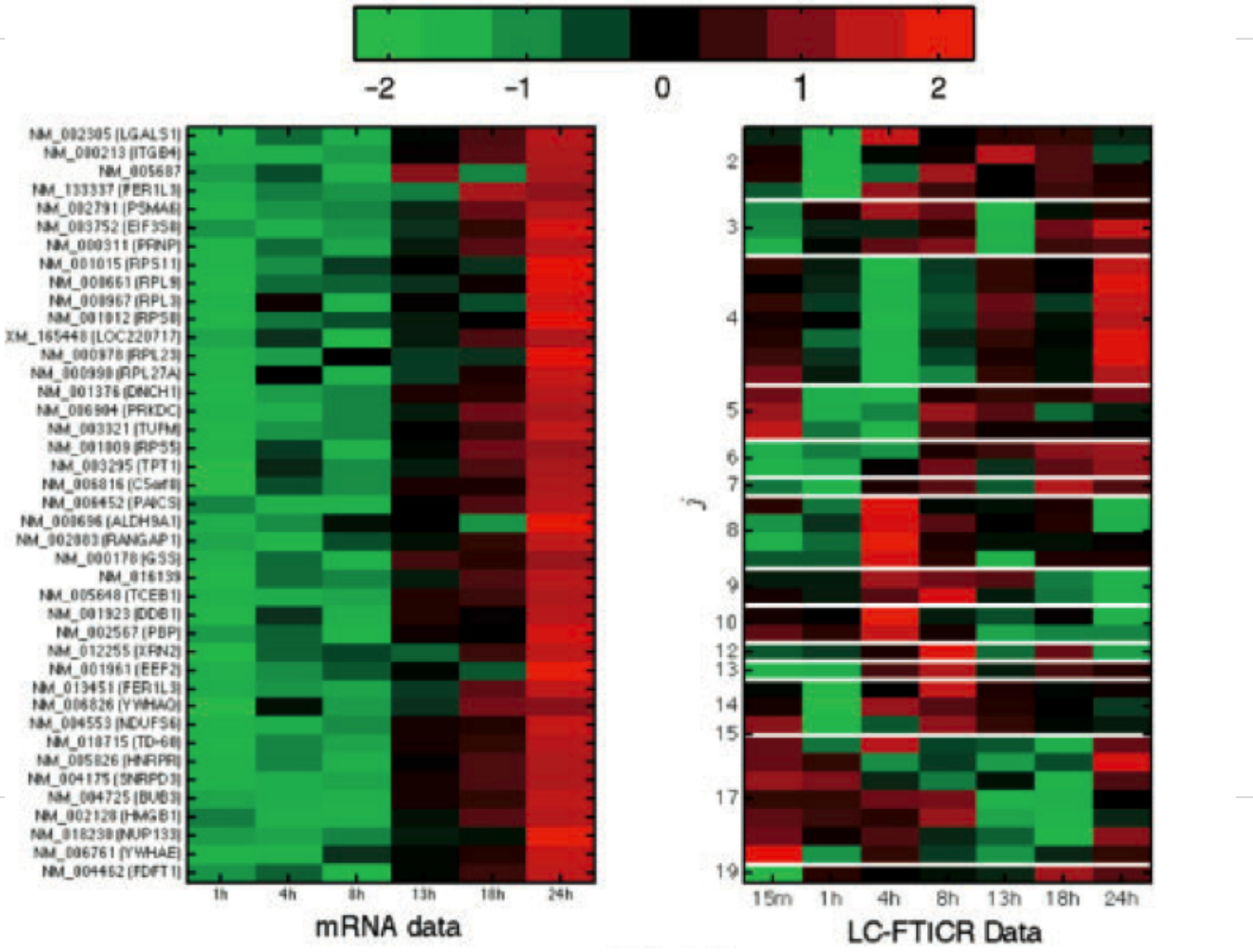
- If all control before transcription, we would expect sparse links.
- That doesn't happen!

Ribosomes



- Some strong links
- These are all ribosomal proteins
- The ribosome is where proteins are constructed
- Makes sense for them to be tightly transcriptionally controlled

Crazy genes



- In some cases, profiles were all over the place
- Here, highly conserved mRNA profiles, diverse protein profiles

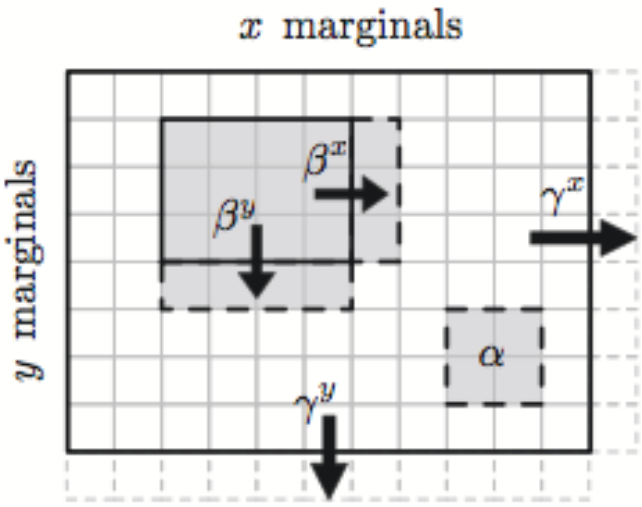
More flexible models

- At this point, we thought about more flexible models (!)
- In particular, how about decomposing the joint density as:

$$P(z_{nk} = 1, z_{nj} = 1) = \sum_i P(z_{nk} = 1|i)P(z_{nj} = 1|i)P(i)$$

- Each latent factor (i) defines a distribution over mRNA and protein clusters
- Use DP priors on i and the clusters in the two *views*

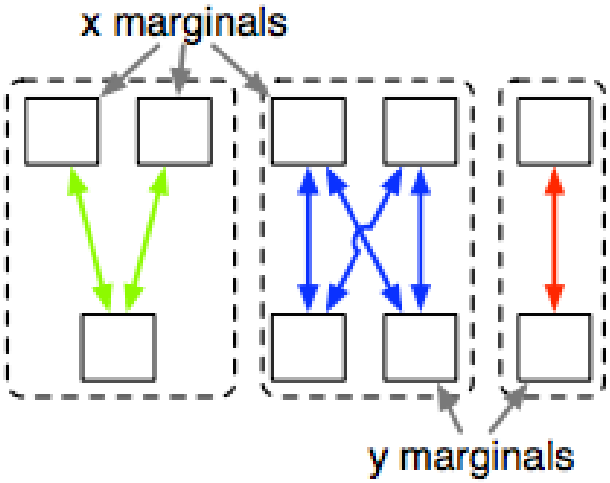
Contingency tables



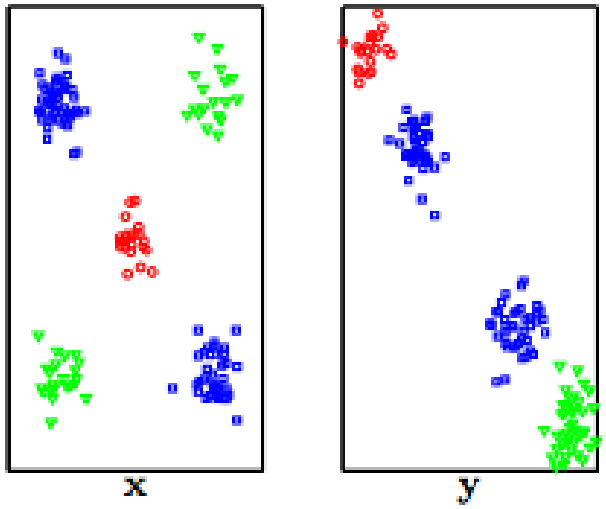
- Can visualise $P(j, k)$ as a table
- Each i is a block (if the clusters are ordered nicely)
- Numbers of rows, columns and blocks can all vary
- Restaurant analogies are possible but unhelpful

- Inference can be done with Gibbs sampling
- Details in Rogers et. al 2009

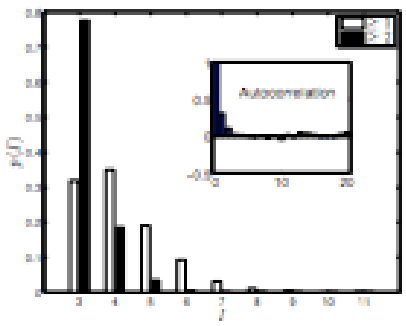
Synthetic example



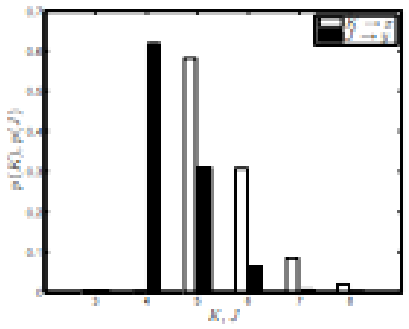
(a) Structure of Gaussian synthetic example. Top boxes represent x marginal components, bottom y. Arrows and dashed lines represent the block structure.



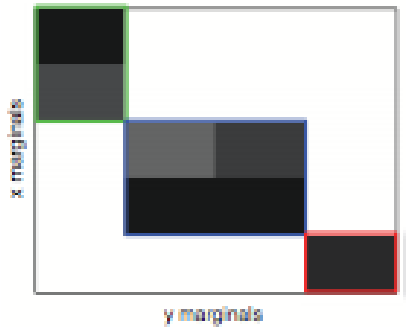
(b) Synthetic dataset for $\mathbf{x}$ (left) and $\mathbf{y}$ (right). Symbols/colors represent top-level clustering.



(c) Marginal posterior distribution over the number of top-level components, I. White bars show the histogram over all clusters, whereas black bars ignore singleton clusters. (Inset: autocorrelation)



(d) Posterior distribution over the number of marginal components. For both margins the mode corresponds with the true solution.



(e) Example contingency table from the sampler. Gray shades denote the count of samples in each cell, and the block structure corresponds exactly to that shown in subfigure (a).

Interpretation

- Interpretation is hard
- Ultimately we're interested in biological processes present in the data

Interpretation

- Interpretation is hard
- Ultimately we're interested in biological processes present in the data
- For each gene GO annotations are available
  - These tell us what the gene is known to be involved in

Interpretation

- Interpretation is hard
- Ultimately we're interested in biological processes present in the data
- For each gene GO annotations are available
  - These tell us what the gene is known to be involved in
- We can compute whether or not a GO term is *enriched* in a cluster

Interpretation

- Interpretation is hard
- Ultimately we're interested in biological processes present in the data
- For each gene GO annotations are available
  - These tell us what the gene is known to be involved in
- We can compute whether or not a GO term is *enriched* in a cluster
- Can do this at each posterior sample for all three types of cluster

Interpretation

- Interpretation is hard
- Ultimately we're interested in biological processes present in the data
- For each gene GO annotations are available
  - These tell us what the gene is known to be involved in
- We can compute whether or not a GO term is *enriched* in a cluster
- Can do this at each posterior sample for all three types of cluster
- Average the p-values across samples to obtain average values for each gene in the three cluster types.

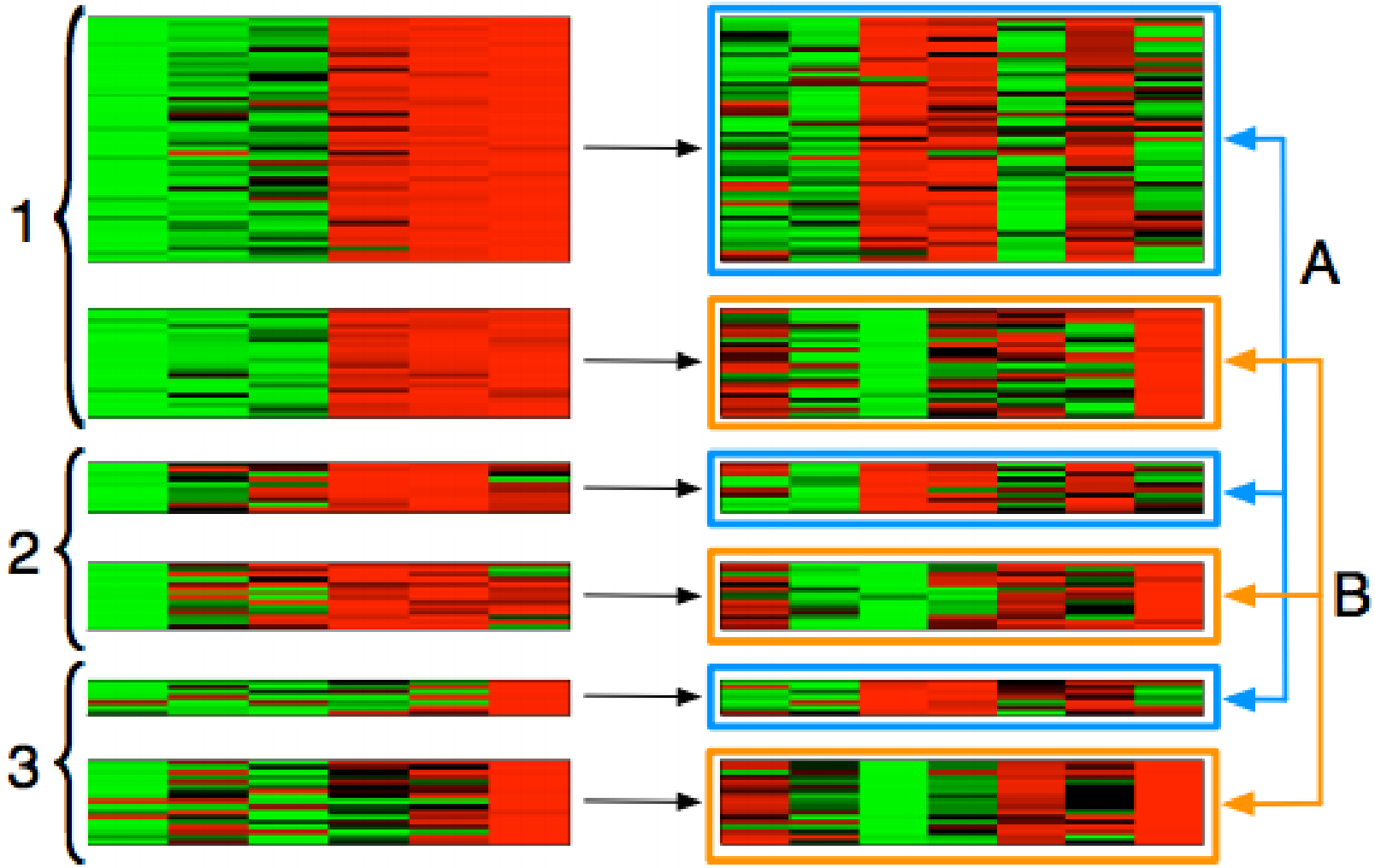
Interpretation

- Interpretation is hard
- Ultimately we're interested in biological processes present in the data
- For each gene GO annotations are available
  - These tell us what the gene is known to be involved in
- We can compute whether or not a GO term is *enriched* in a cluster
- Can do this at each posterior sample for all three types of cluster
- Average the p-values across samples to obtain average values for each gene in the three cluster types.
- Tells us if that gene is involved in that process according to mRNA, Protein, or both

Interpretation

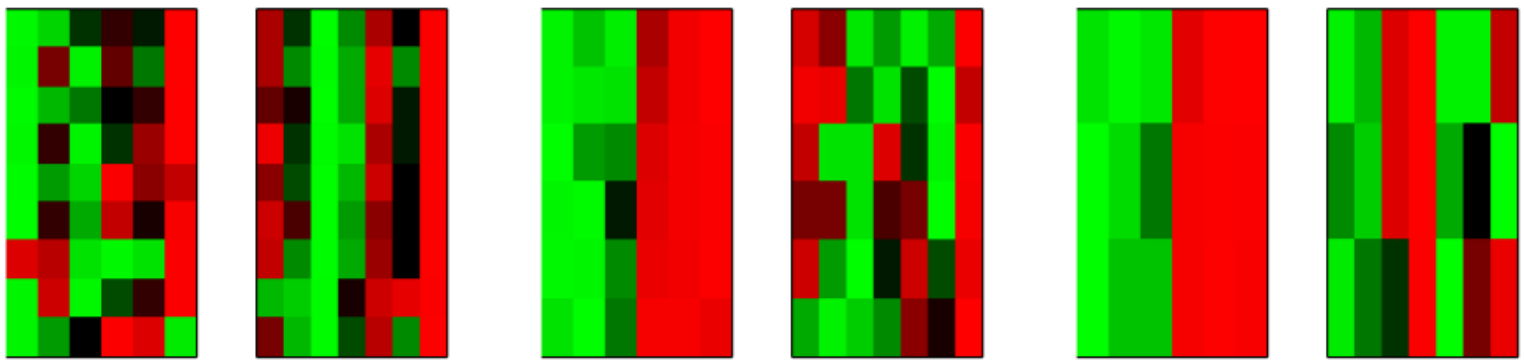
- Interpretation is hard
- Ultimately we're interested in biological processes present in the data
- For each gene GO annotations are available
  - These tell us what the gene is known to be involved in
- We can compute whether or not a GO term is *enriched* in a cluster
- Can do this at each posterior sample for all three types of cluster
- Average the p-values across samples to obtain average values for each gene in the three cluster types.
- Tells us if that gene is involved in that process according to mRNA, Protein, or both
- As in previous example, the clustering is not our final goal!

Results 1: what kind of blocks are present



- Highly inter-connected. Clusters on left shown by numbers, on right by letters.

Results 2: what's enriched?

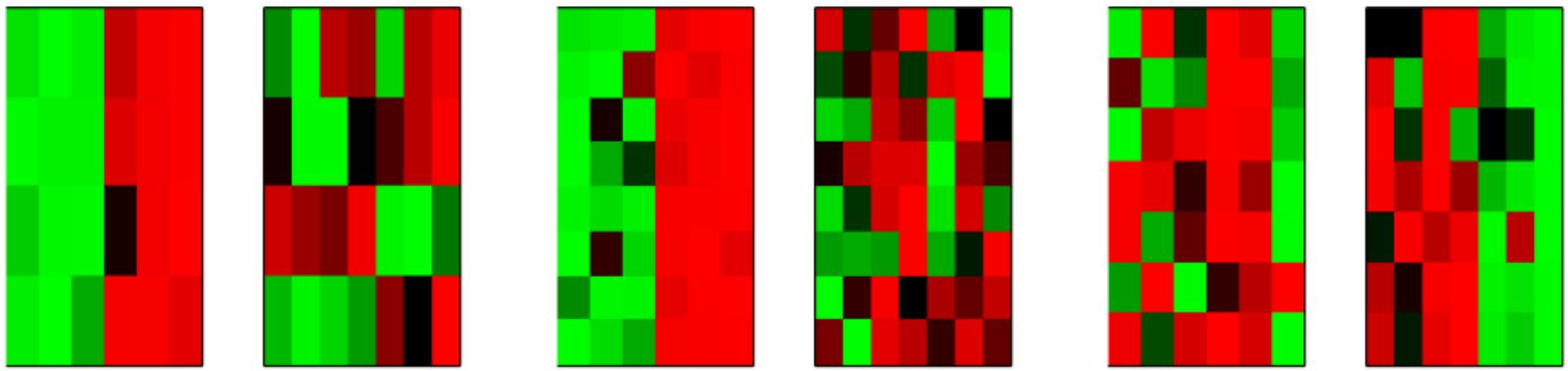


(a) GO:0003735 (b) GO:0008380 (c) GO:0000502

Fig. 8 3 examples of gene ontology terms significantly enriched in top level components. In all cases, left heat map is mRNA data, right is protein data.

- Some terms (and genes) enriched in the top components (i.e. for mRNA and protein)

Results 2: what's enriched?



(a) GO:0006281 (b) GO:0006457 (c) GO:0007155

- Some terms (and genes) enriched in one component and not the other (a,b: enriched in mRNC; c: enriched in protein)

Conclusions

- Flexible model can pull our interesting biology
- Actually equivalent to Rich Savage's model from Savage et. al 2010 for integrating mRNA and Transcription Factor (TF) binding data
- But, it's hard to use and interpret

Lecture 10: Summary

Dr. Simon Rogers
School of Computing Science
University of Glasgow
simon.rogers@glasgow.ac.uk
@sdrogers

May 13, 2014

Summary: GPs

- Flexible method for regression
- Auxiliary variable trick allows:
  - Binary classification
  - Multi-class classification
  - Semi-supervised classification
  - Ordinal regression
- Applications:
  - Modelling uncertainty in text entry
  - Investigating the disagreement between A&E clinicians

Summary: DPs

- Infinite prior for mixture models
- Infer the number of clusters
- No magic: The cost is that we have to define what a cluster is quite precisely
- Applications:
  - Identifying metabolites
  - Finding patterns across different datasets

Summary: all

- GPs and DPs allow computationally tractable Bayesian inference
- In all applications, inference of the curve, or clustering was only an intermediate step:
  - *Typing*: propogate uncertainty in key predictions
  - *Clinicians*: average over curve to infer characteristics of clinicians
  - *Metabolomics*: average over clustering to annotate peaks
  - *Multiview*: average over clusterings to extract biological information
- Thinking about what the results of regression / clustering will be used for is useful.
- GP and DP provide access to the posterior and therefore the ability to average over it