Automated Variational Inference for Gaussian Process Models

Edwin V. Bonilla
Senior Lecturer
The University of New South Wales

March 10th, 2015

Bayesian Inference is Darn Hard

Prior p(**f**)

End of last EPL season

- Leicester City rock bottom for a big chunk of the season
- pulled off a 'Great Escape' to stay in Premier League

Likelihood p(**y** | **f**)

This season

Leicester City players 'on fire'

Posterior p(**f** | **y**)

End of current season (?)

$$p(\mathbf{f}|\mathbf{y}) = \frac{p(\mathbf{f})p(\mathbf{y}|\mathbf{f})}{\int p(\mathbf{f})p(\mathbf{y}|\mathbf{f})\ \mathrm{df}}$$

Why is this so hard?

Approximate Bayesian Inference
Markov Chain Monte Carlo

Pseudo-Marginal Bayesian Inference for Gaussian Processes *(TPAMI, 2014)*

Maurizio Filippone and Mark Girolami

Abstract—The main challenges that arise when adopting Gaussian Process priors in probabilistic modeling are how to carry out exact Bayesian inference and how to account for uncertainty on model parameters when making model-based predictions on out-of-sample data. Using probit regression as an illustrative working example, this paper presents a general and effective methodology based on the pseudo-marginal approach to Markov chain Monte Carlo that efficiently addresses both of these issues. The results presented in this paper show improvements over existing sampling methods to simulate from the posterior distribution over the parameters defining the covariance function of the Gaussian Process prior. This is particularly important as it offers a powerful tool to carry out full Bayesian inference of Gaussian Process based hierarchic statistical models in general. The results also demonstrate that Monte Carlo based integration of all model parameters is actually feasible in this class of models providing a superior quantification of uncertainty in predictions. Extensive comparisons with respect to state-of-the-art probabilistic classifiers confirm this assertion.

- Markov Chain Monte Carlo
  - Sampling
  - Exact (?) Bayesian inference
- Efficient
  - Binary classification
  - N = 2835 training points, how long on a decent desktop?
  - Running time ~ 2 weeks

Approximate Bayesian Inference
Variational Inference

Collaborative Multi-output Gaussian Processes *(UAI, 2014)*

Trung V. Nguyen
ANU & NICTA
Canberra, Australia

Edwin V. Bonilla
NICTA & ANU
Sydney, Australia

Abstract

We introduce the collaborative multi-output Gaussian process (GP) model for learning dependent tasks with very large datasets. The model fosters task correlations by mixing sparse processes and sharing multiple sets of inducing points. This facilitates the application of variational inference and the derivation of an evidence lower bound that decomposes across inputs and outputs. We learn all the parameters of the model in a single stochastic optimization framework that scales to a large number of observations per output and a large number of outputs. We demonstrate our approach on a toy problem, two medium-sized datasets and a large dataset. The model achieves superior performance compared to single output learning and previous multi-output GP models, confirming the benefits of correlating sparsity structure of the outputs via the inducing points.

- Variational inference
  - Deterministic approximation
  - Optimization
- Efficient?
  - Multi-output regression
  - N > 46k training points, *how long?*
  - 1.9h in Trung's laptop
  - *Where is the catch?*

Approximate Bayesian Inference
Variational Inference

2 Derivation of the Variational Lower Bound

The variational lower bound of the log marginal (eq. 13 in the main text) is given by:

$$\begin{aligned}\log p(\mathbf{y}) \geq & \int q(\mathbf{u},\mathbf{v}) \log \frac{p(\mathbf{y}|\mathbf{u},\mathbf{v})p(\mathbf{u},\mathbf{v})}{q(\mathbf{u},\mathbf{v})} \mathrm{d}\mathbf{u}\mathrm{d}\mathbf{v} \\ & + \int q(\mathbf{u},\mathbf{v}) \log \frac{p(\mathbf{u},\mathbf{v})}{q(\mathbf{u},\mathbf{v})} \mathrm{d}\mathbf{u}\mathrm{d}\mathbf{v} \\ = & \int q(\mathbf{u},\mathbf{v}) \log p(\mathbf{y}|\mathbf{u},\mathbf{v}) \mathrm{d}\mathbf{u}\mathrm{d}\mathbf{v}\end{aligned}$$

3.1 Derivative of the Noise Hyperparameter

The derivative of the noise hyperparameter β is easily computed as:

$$\frac{\partial \mathcal{L}}{\partial \beta} = \frac{N}{2\beta} - \frac{1}{2}(\mathbf{y}-\mathbf{A}\mathbf{m})^T(\mathbf{y}-\mathbf{A}\mathbf{m}) - \frac{\mathcal{L}_2}{\beta} - \frac{\mathcal{L}_3}{\beta}. \quad (5)$$

3.2 Derivatives of the Covariance Hyperparameters

To simplify the math, we utilize the matrix $\mathbf{A}$ defined above. Firstly, the derivative of $\mathbf{A}$ wrt a covariance hyperparameter t is given by:

$$\frac{\partial \mathbf{A}}{\partial t} = \left(\frac{\partial \mathbf{K}_{NM}}{\partial t} - \mathbf{A}\frac{\partial \mathbf{K}_{MM}}{\partial t}\right)\mathbf{K}_{MM}^{-1}. \quad (6)$$

The derivatives of $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ and $\mathcal{L}_4$ are thus given by:

$$\frac{\partial \mathcal{L}_1}{\partial t} = \beta(\mathbf{y}-\mathbf{A}\mathbf{m})^T\frac{\partial \mathbf{A}}{\partial t}\mathbf{m} \quad (7)$$

$$\frac{\partial \mathcal{L}_2}{\partial t} = \frac{1}{2}\beta \operatorname{tr}\left(\frac{\partial \mathbf{K}_{NN}}{\partial t} - \mathbf{A}\frac{\partial \mathbf{K}_{MN}}{\partial t} - \frac{\partial \mathbf{A}}{\partial t}\mathbf{K}_{MN}\right) \quad (8)$$

$$\begin{aligned}\log\ p(\mathbf{y}|\mathbf{u},\mathbf{v}) &= \log \langle p(\mathbf{y}|\mathbf{g},\mathbf{h})\rangle_{p(\mathbf{g},\mathbf{h}|\mathbf{u},\mathbf{v})} \\ &\geq \langle \log p(\mathbf{y}|\mathbf{g},\mathbf{h})\rangle_{p(\mathbf{g},\mathbf{h}|\mathbf{u},\mathbf{v})} \\ &= \sum_{i=1}^{P}\sum_{n=1}^{N} \langle \log p(y_{in}|\mathbf{g}_n, h_{in})\rangle_{p(\mathbf{g}|\mathbf{u})p(\mathbf{h}_i|\mathbf{v}_i)}\end{aligned}$$

Applying the identity in eq. 1, the expectation of an individual likelihood term with respect to the posterior distribution is given by:

$$\begin{aligned}l_{in} = & \int \log p(y_{in}|\mathbf{g}_n, h_{in}) \prod_{j=1}^{Q} p(g_{jn}|\mathbf{u}_j) p(h_{in}|\mathbf{u}_i) \mathrm{d}\mathbf{g}_n \mathrm{d}h_{in} \\ = & \log \mathcal{N}(y_{in}; \sum_{j=1}^{Q} w_{ij}\mu_{jn} + \mu_{in}^h, \beta_i^{-1}) \\ & - \frac{1}{2}\beta_i \sum_{j=1}^{Q} w_{ij}^2 \tilde{k}_{jnn} - \frac{1}{2}\beta_i \tilde{k}_{inn}^h, \quad (3)\end{aligned}$$

where $\tilde{k}_{jnn} = (\tilde{\mathbf{K}}_j)_{nn}$, $\tilde{k}_{inn}^h = (\tilde{\mathbf{K}}_i^h)_{nn}$, $\mu_{jn} = (\boldsymbol{\mu}_j)_n$, and $\mu_{in}^h = (\boldsymbol{\mu}_i^h)_n$.

lower bound as a function of all parameters is

$$\begin{aligned}\mathcal{L} = & \log \mathcal{N}(\mathbf{y}; \mathbf{K}_{NM}\mathbf{K}_{MM}^{-1}\mathbf{m}, \beta^{-1}\mathbf{I}) \\ & - \frac{1}{2}\beta \operatorname{tr}\tilde{\mathbf{K}} - \frac{1}{2}\beta \operatorname{tr}(\mathbf{S}\mathbf{K}_{MM}^{-1}\mathbf{K}_{MN}\mathbf{K}_{NM}\mathbf{K}_{MM}^{-1}) \\ & - \frac{1}{2}\left(\log|\mathbf{K}_{MM}| + \operatorname{tr}(\mathbf{K}_{MM}^{-1}(\mathbf{m}\mathbf{m}^T + \mathbf{S}))\right) \\ = & \underbrace{\log \mathcal{N}(\mathbf{y}; \mathbf{A}\mathbf{m}, \beta^{-1}\mathbf{I})}_{\mathcal{L}_1} \underbrace{- \frac{1}{2}\beta \operatorname{tr}(\mathbf{K}_{NN} - \mathbf{A}\mathbf{K}_{MN})}_{\mathcal{L}_2} \\ & \underbrace{- \frac{1}{2}\left(\log|\mathbf{K}_{MM}| + \operatorname{tr}(\mathbf{K}_{MM}^{-1}(\mathbf{m}\mathbf{m}^T + \mathbf{S}))\right)}_{\mathcal{L}_4} \\ & \underbrace{- \frac{1}{2}\beta \operatorname{tr}(\mathbf{S}\mathbf{A}^T\mathbf{A})}_{\mathcal{L}_3}, \quad (4)\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{L}_1}{\partial t} = & \beta \operatorname{tr}(\mathbf{y}-\mathbf{A}\mathbf{m})^T\left(\frac{\partial \mathbf{K}_{NM}}{\partial t} - \mathbf{A}\frac{\partial \mathbf{K}_{MM}}{\partial t}\right)\mathbf{K}_{MM}^{-1}\mathbf{m} \\ = & \beta \operatorname{tr}\mathbf{K}_{MM}^{-1}\mathbf{m}(\mathbf{y}-\mathbf{A}\mathbf{m})^T\frac{\partial \mathbf{K}_{NM}}{\partial t} \\ & - \beta \operatorname{tr}\mathbf{K}_{MM}^{-1}\mathbf{m}(\mathbf{y}-\mathbf{A}\mathbf{m})^T\mathbf{A}\frac{\partial \mathbf{K}_{MM}}{\partial t} \quad (11)\end{aligned}$$

$$\frac{\partial \mathcal{L}_2}{\partial t} = -\beta \operatorname{tr}\mathbf{A}^T\frac{\partial \mathbf{K}_{NM}}{\partial t} + \frac{1}{2}\beta \operatorname{tr}\mathbf{A}^T\mathbf{A}\frac{\partial \mathbf{K}_{MM}}{\partial t} \quad (12)$$

$$\frac{\partial \mathcal{L}_3}{\partial t} = \beta \operatorname{tr}\mathbf{K}_{MM}^{-1}\mathbf{S}\mathbf{A}^T\frac{\partial \mathbf{K}_{NM}}{\partial t} - \beta \operatorname{tr}\mathbf{K}_{MM}^{-1}\mathbf{S}\mathbf{A}^T\mathbf{A}\frac{\partial \mathbf{K}_{MM}}{\partial t} \quad (13)$$

$$\begin{aligned}\frac{\partial \mathcal{L}_4}{\partial t} = & \frac{1}{2}\operatorname{tr}\mathbf{K}_{MM}^{-1}\frac{\partial \mathbf{K}_{MM}}{\partial t} \quad (14) \\ & - \frac{1}{2}\operatorname{tr}\mathbf{K}_{MM}^{-1}(\mathbf{m}\mathbf{m}^T + \mathbf{S})\mathbf{K}_{MM}^{-1}\frac{\partial \mathbf{K}_{MM}}{\partial t}\end{aligned}$$

From the above 4 equations we get,

$$\frac{\partial \mathcal{L}}{\partial t} = \operatorname{tr}\mathbf{D}_1\frac{\partial \mathbf{K}_{NM}}{\partial t} + \operatorname{tr}\mathbf{D}_2\frac{\partial \mathbf{K}_{MM}}{\partial t}, \quad (15)$$

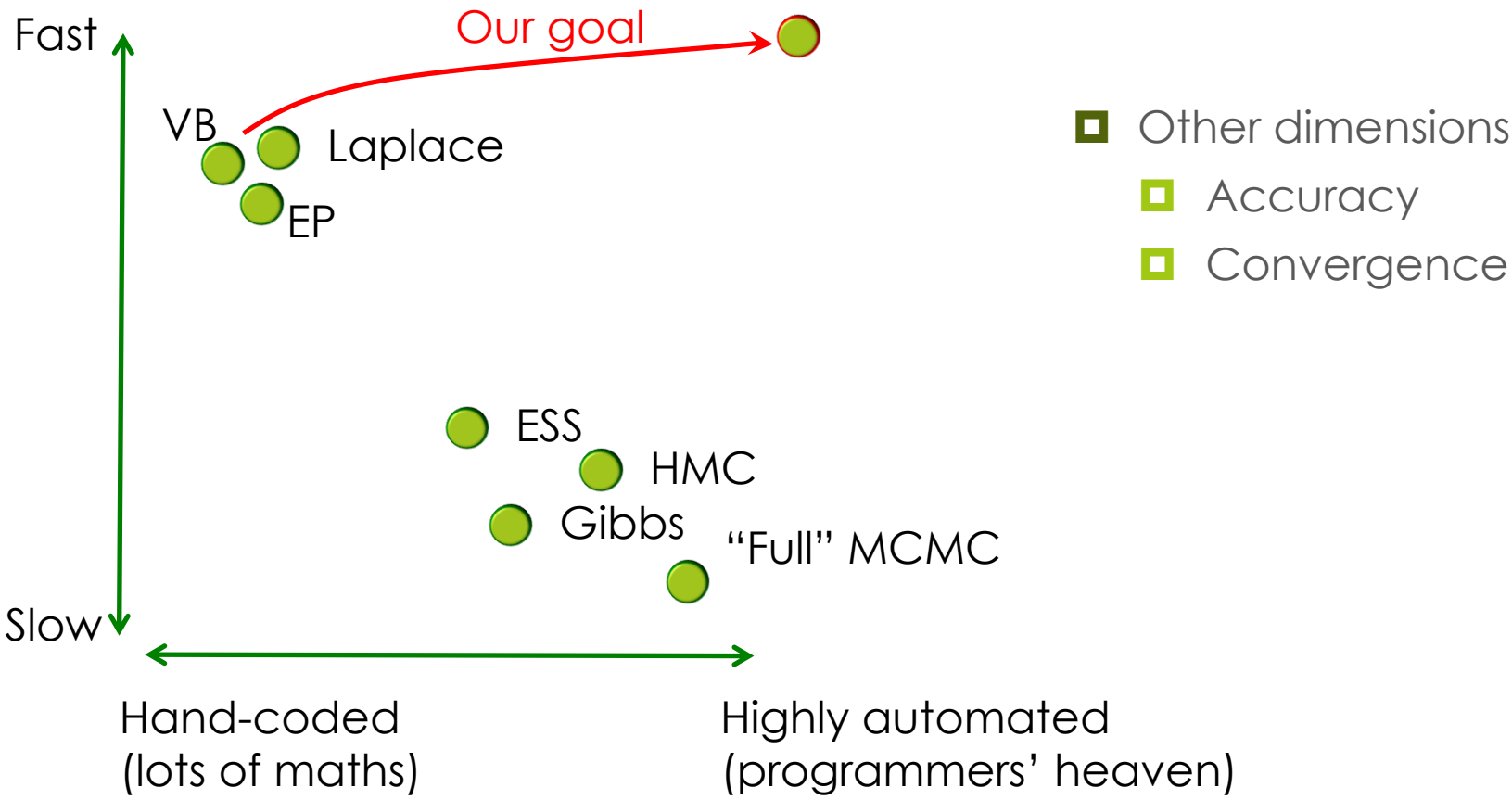
where

$$\mathbf{D}_1 = \beta\mathbf{K}_{MM}^{-1}\mathbf{m}(\mathbf{y}-\mathbf{A}\mathbf{m})^T + \beta\mathbf{A}^T - \beta\mathbf{K}_{MM}^{-1}\mathbf{S}\mathbf{A}^T \quad (16)$$

$$\begin{aligned}\mathbf{D}_2 = & -\beta\operatorname{tr}\mathbf{K}_{MM}^{-1}\mathbf{m}(\mathbf{y}-\mathbf{A}\mathbf{m})^T\mathbf{A} - \frac{1}{2}\beta\mathbf{A}^T\mathbf{A} - \frac{1}{2}\mathbf{K}_{MM}^{-1} \\ & + \beta\mathbf{K}_{MM}^{-1}\mathbf{S}\mathbf{A}^T\mathbf{A} + \frac{1}{2}\mathbf{K}_{MM}^{-1}(\mathbf{m}\mathbf{m}^T + \mathbf{S})\mathbf{K}_{MM}^{-1} \quad (17)\end{aligned}$$

Goal: reduce the # papers published on variational inference ☺

Approximate Bayesian Inference
Automation vs. Efficiency



- Other dimensions
  - Accuracy
  - Convergence

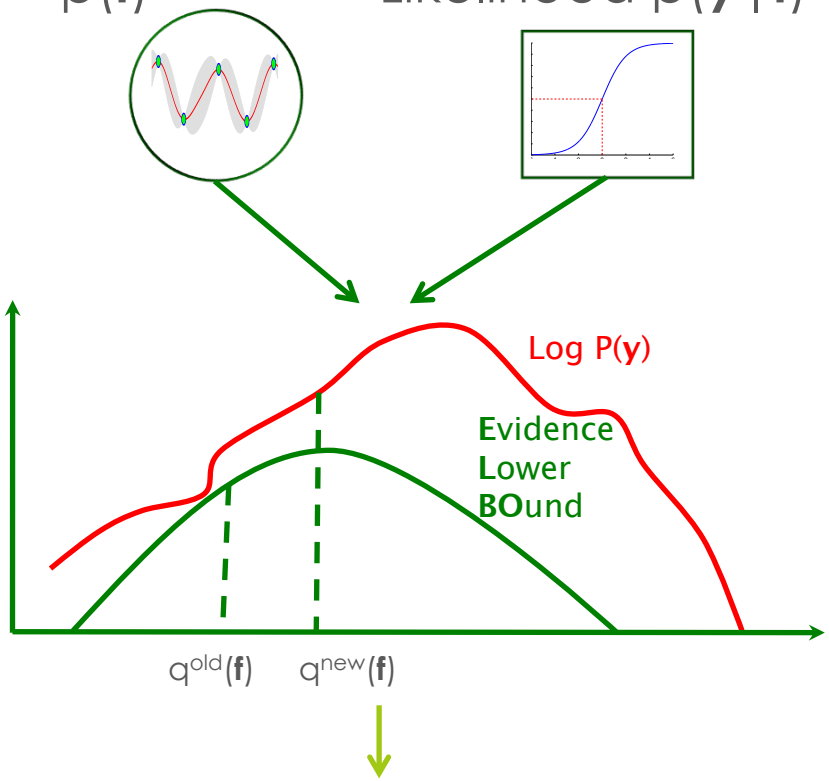
Non-sarcastic goal: We want to build generic yet practical inference tools for practitioners and researchers

This Work at a Glance

Automated Variational Inference (Nguyen & Bonilla, NIPS 2014)

GP prior
p(**f**)

'black-box'
Likelihood p(**y** | **f**)



Approximate posterior:
Mixture of Gaussians q(**f**)

- ELBO = - KL + ELL
  - KL divergence
    - Analytical lower bound
    - Exact gradients
  - Expected log Likelihood (ELL)
    - Expectations over univariate Gaussians
    - No explicit gradients needed
- Practical framework
  - Efficient parameterization
  - As good as hand-coded solutions
  - Orders of magnitude faster than MCMC

A Brief Intro to Gaussian Processes

Gaussian Processes (GPs)

Definition: Gaussian Process

$f(\mathbf{x})$ is a Gaussian process if for any subset of points $\mathbf{x}_1, \ldots, \mathbf{x}_N$, the function values $f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)$ follow a **consistent** Gaussian distribution.

- Consistency: marginalization property
- Notation

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

Mean function

$$\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$$

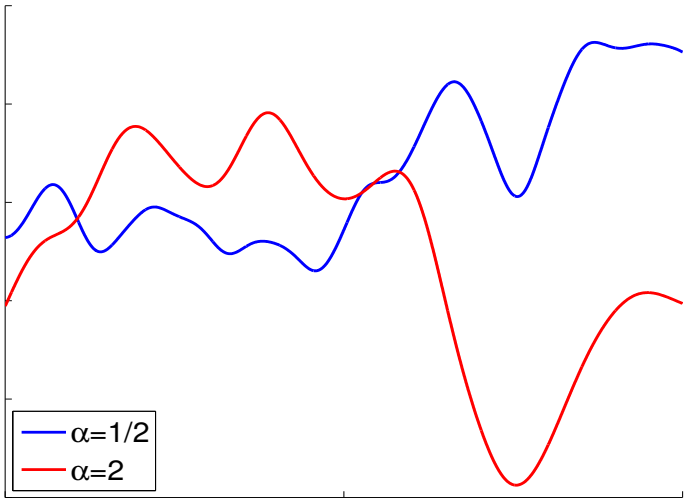
Covariance function

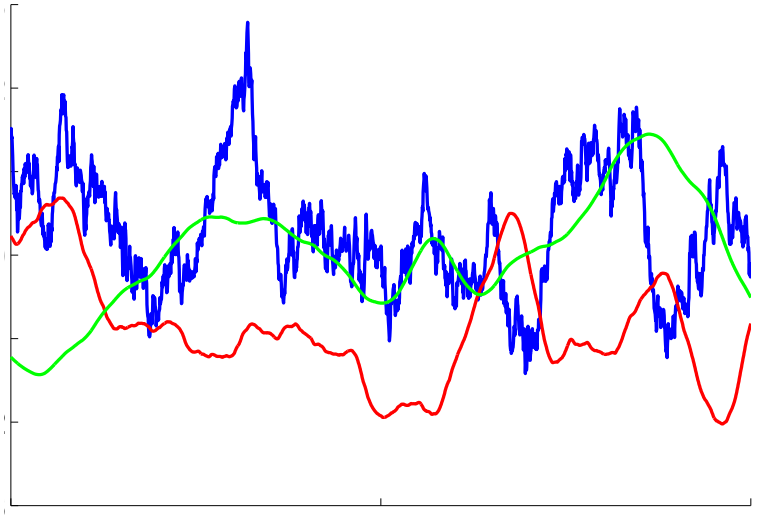
$$\kappa(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))]$$

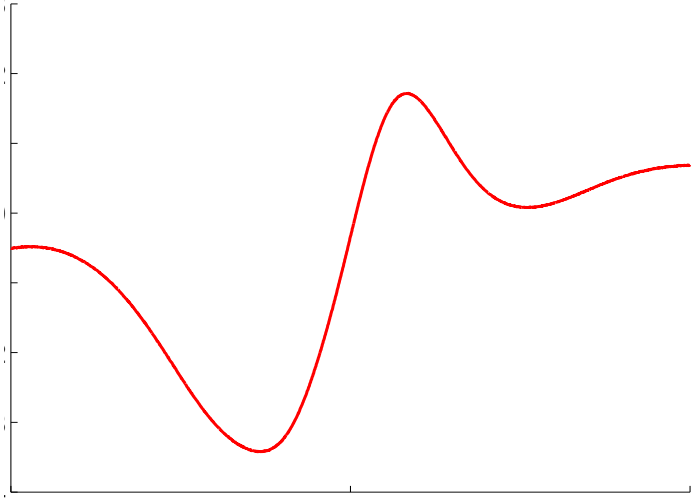
Hyper-parameters

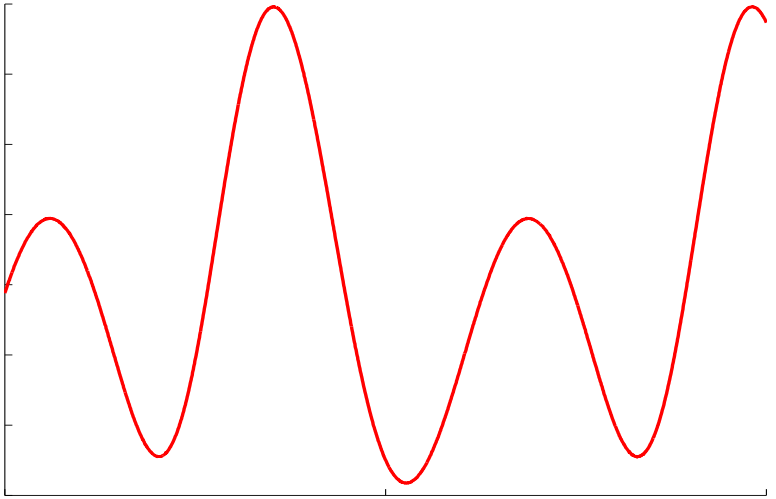
- A GP is a distribution over functions
  - There is not such a thing as the GP method

Samples from a Gaussian Process

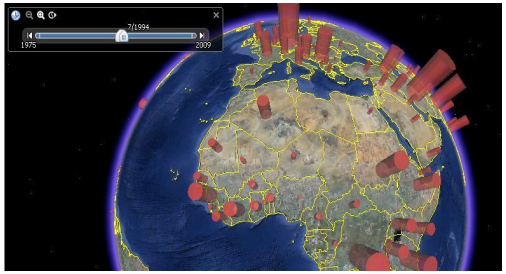








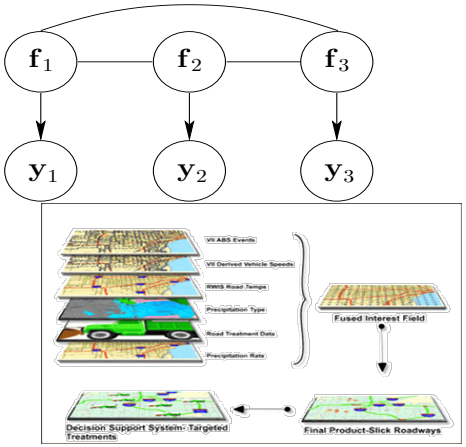
Some Applications of Gaussian Process (GP) Models



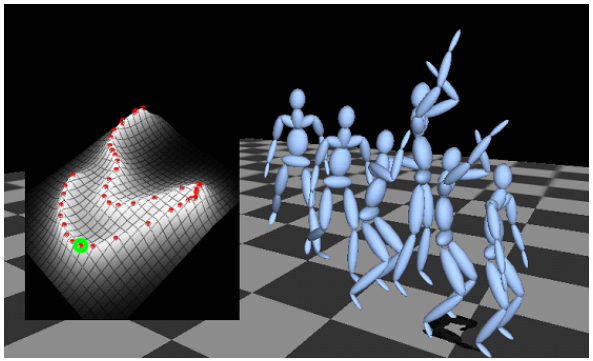
Spatio-temporal modelling

Classification

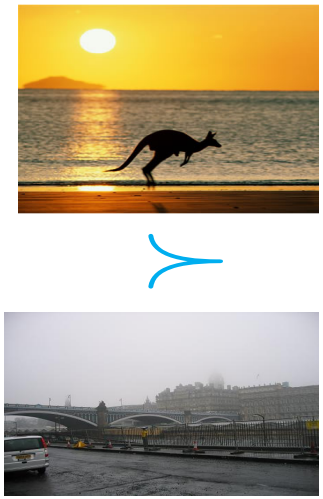
Robot inverse dynamics



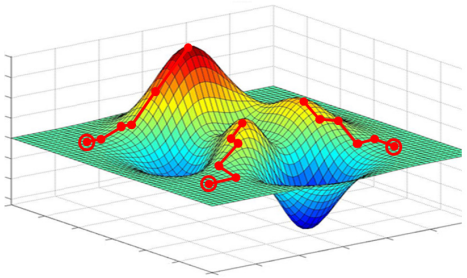
Data fusion / multi-task learning



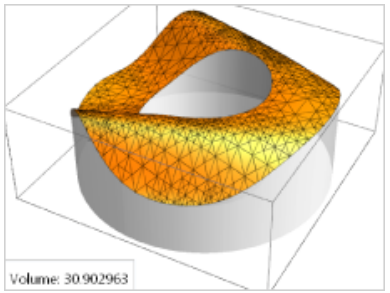
Style-based inverse kinematics



Preference learning



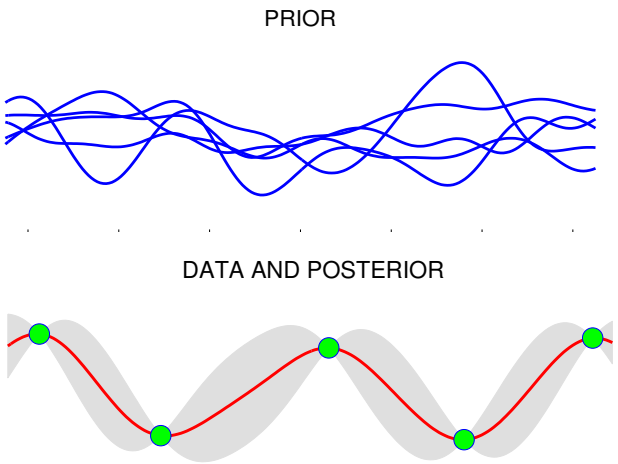
Bayesian optimization



Bayesian quadrature

How Can We 'Solve' All these Problems with the Humble Gaussian Distribution?

- Key components of GP models
  - Non-parametric prior
  - Bayesian
  - Kernels (covariance functions)
- Tasks
  - Prediction (posterior inference)
  - Hyper-parameter learning
- What do we pay?
  - 'Intractability' for non-Gaussian likelihoods
    - E.g. a sigmoid likelihood for classification
  - High Computational cost with # data-points
    - In time and memory



Bayesian non-linear regression

This talk is about approaches for dealing with general likelihood models, i.e. automated inference

A Large Class of GP Models

Automated Variational Inference

A Family of Gaussian Process Models

Latent Gaussian Process Models

- Supervised Learning Problems
  - Inputs: $\mathbf{x} = \{\mathbf{x}_n\}_{n=1}^N$ Labels: $\mathbf{y} = \{\mathbf{y}_n\}_{n=1}^N$

- Factorization of GP prior over Q latent functions

$$f_j \sim \mathcal{GP}(0, \kappa_j(\cdot,\cdot)) \rightarrow p(\mathbf{f}|\boldsymbol{\theta}_0) = \prod_{j=1}^{Q} p(\mathbf{f}_{\bullet j}|\boldsymbol{\theta}_0) = \prod_{j=1}^{Q} \mathcal{N}(\mathbf{f}_{\bullet j}; \mathbf{0}, \mathbf{K}_j)$$

Covariance function of jth GP

All NxQ latent function values

Covariance Hyper-parameters

All N latent values for function j

Covariance matrix induced by κ_j

- Factorization of conditional likelihood

$$p(\mathbf{y}|\mathbf{f}, \boldsymbol{\theta}_1) = \prod_{n=1}^{N} p(\mathbf{y}_n|\mathbf{f}_{n\bullet}, \boldsymbol{\theta}_1)$$

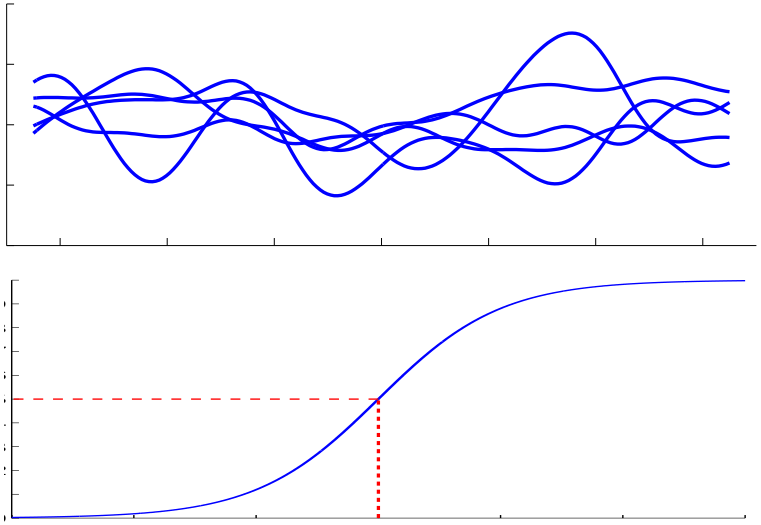
Cond. Likelihood parameters

Observations and latent functions for data-point n

What can we model with this framework?

Latent Gaussian Process Models
Examples

- Multi-class classification
  - Q classes → Q independent GP priors $p(f_j)$, $j = 1, \ldots Q$
    - Each GP can have a different covariance
  - Softmax likelihood
    - $p(y=j) \propto \exp(f_j)$



- Other settings
  - Multi-output regression
  - Warped GPs
  - Log Gaussian Cox process
  - Others
    - Access to 'black-box' likelihood

Automated Variational Inference
The General Framework

- Goal: *Approximate 'intractable' posterior p(**f** | **y**)*
  - Find the closest tractable approximation q(**f**)

$$q(\mathbf{f}|\boldsymbol{\lambda}) = \sum_{k=1}^{K} \pi_k q_k(\mathbf{f}|\boldsymbol{\lambda}_k)$$

  - Minimize KL[q(**f**) || p(**f** | **y**)] → Maximize ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_q[-\log q(\mathbf{f}|\boldsymbol{\lambda})] + \mathbb{E}_q[\log p(\mathbf{f})]}_{-\mathrm{KL}[q(\mathbf{f}|\boldsymbol{\lambda})||p(\mathbf{f})]} + \underbrace{\sum_{k=1}^{K} \pi_k \mathbb{E}_{q_k}[\log p(\mathbf{y}|\mathbf{f})]}_{\mathrm{ELL}}$$

- Irrespective of the likelihood models (black-box):
  - KL can be lower bounded using Jensen's inequality
    - Exact gradients of the GP hyper-parameters can be obtained
  - ELL and its gradients can be approximated *efficiently*

Automated Variational Inference
Expected Log Likelihood (ELL) Term

Theorem 1

The ELL and its gradients can be estimated using expectations over **univariate** Gaussian distributions.

$$q_{k(n)} = q_{k(n)}(\mathbf{f}_{n\bullet}|\boldsymbol{\lambda}_{k(n)})$$

$$\mathbb{E}_{q_k}[\log p(\mathbf{y}|\mathbf{f})] = \sum_{n=1}^{N} \mathbb{E}_{q_{k(n)}}[\log p(\mathbf{y}_n|\mathbf{f}_{n\bullet})]$$

$$\nabla_{\boldsymbol{\lambda}_{k(n)}} \mathbb{E}_{q_{k(n)}}[\log p(\mathbf{y}_n|\mathbf{f}_{n\bullet})] = \mathbb{E}_{q_{k(n)}} \nabla_{\boldsymbol{\lambda}_{k(n)}} \log q_{k(n)}(\mathbf{f}_{n\bullet}|\boldsymbol{\lambda}_{k(n)}) \log p(\mathbf{y}_n|\mathbf{f}_{n\bullet})$$

- Practical consequences
  - We can use Monte Carlo estimates
  - Gradients of the likelihood are not required
    - Only likelihood evaluations are needed
  - Also holds for Q > 1

Automated Variational Inference
Practical Variational Distributions

- Two distribution classes of interest
  - **FG:** Full Gaussian, i.e. K=1, full covariance matrix
  - **MoDG:** Mixture of diagonal Gaussians

Theorem 2

The covariance matrices can be parameterized **linearly** in the number of observations

- Optimization is made easier (less parameters and correlations)

Theorem 3

Gradients estimates of MoDG have **lower variance** than FG's

- Optimization with MoDG converges faster

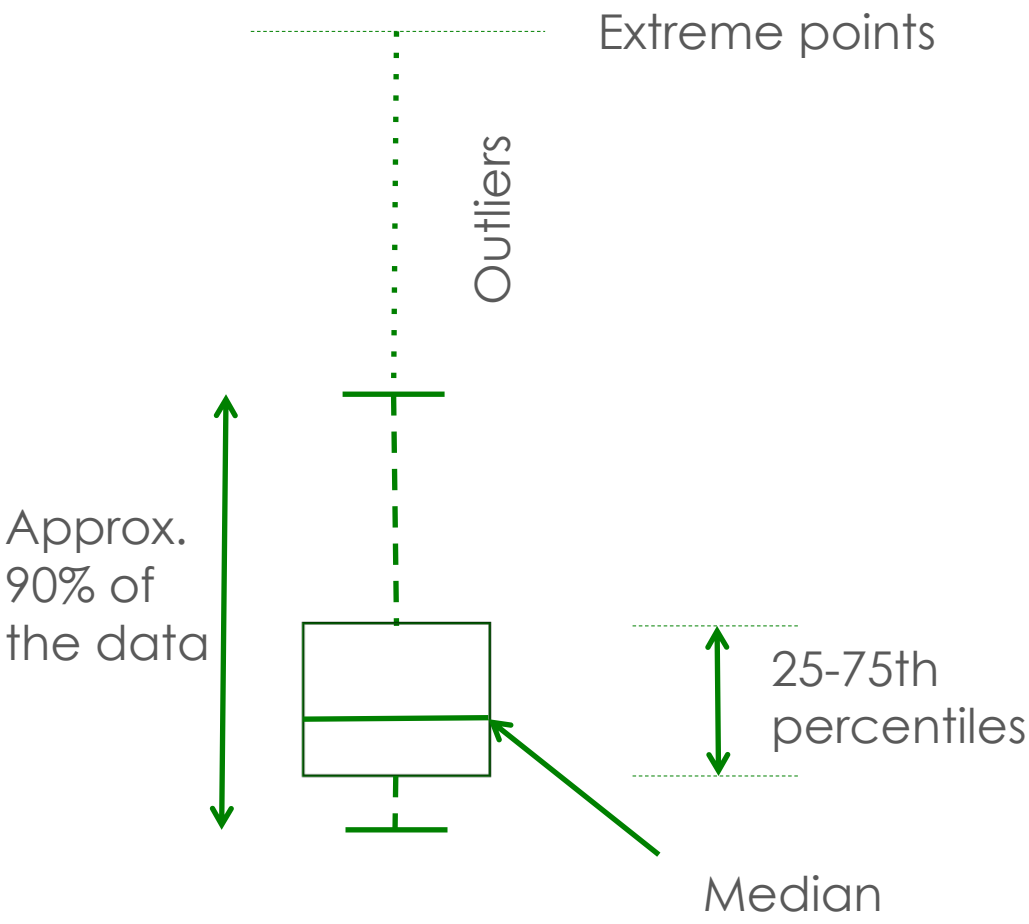
EXPERIMENTS

EXPERIMENTS
PREAMBLE

Performance measures

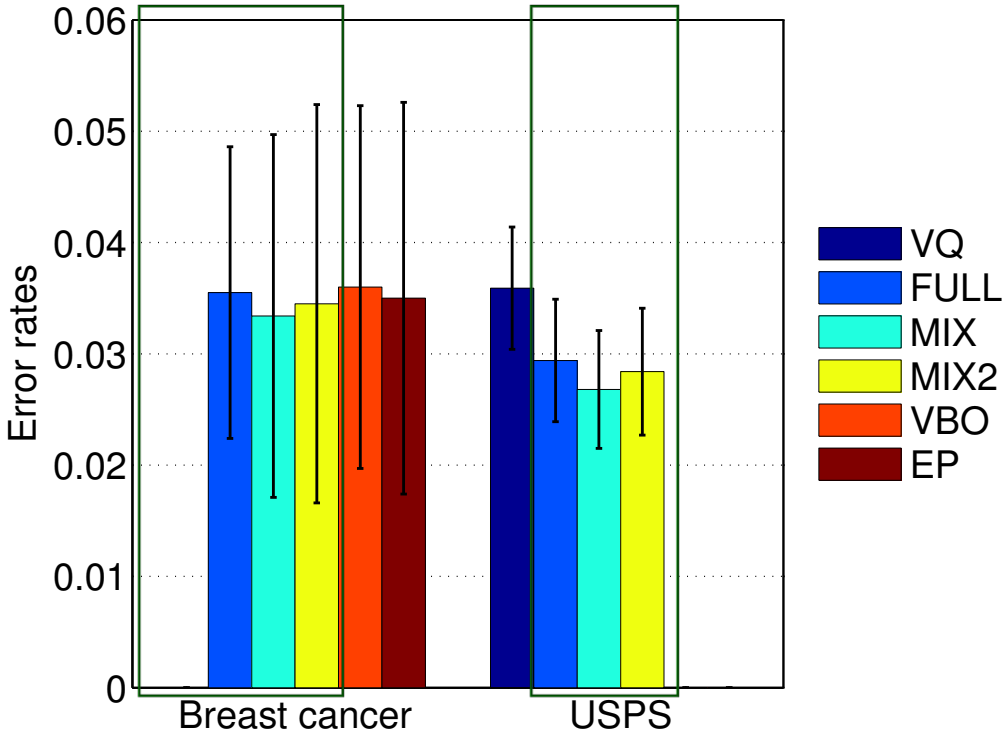
- SSE
  - Standardised square error
- NLPD
  - Negative log predictive density

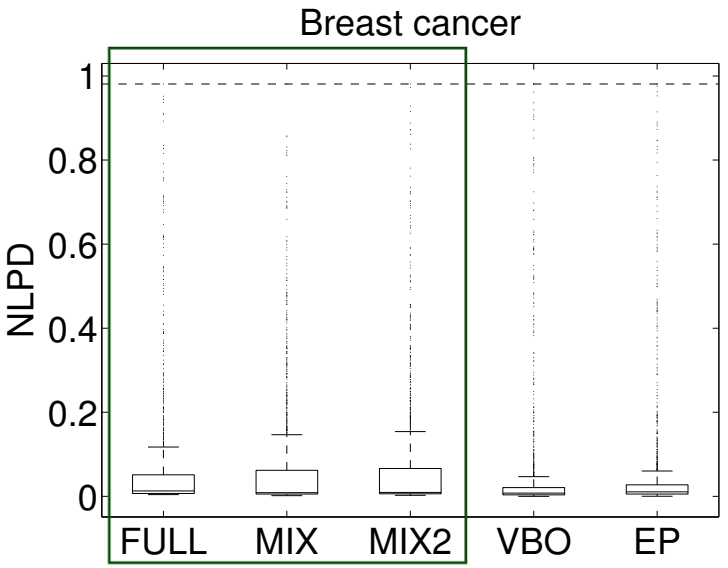
Box-and-whisker plots

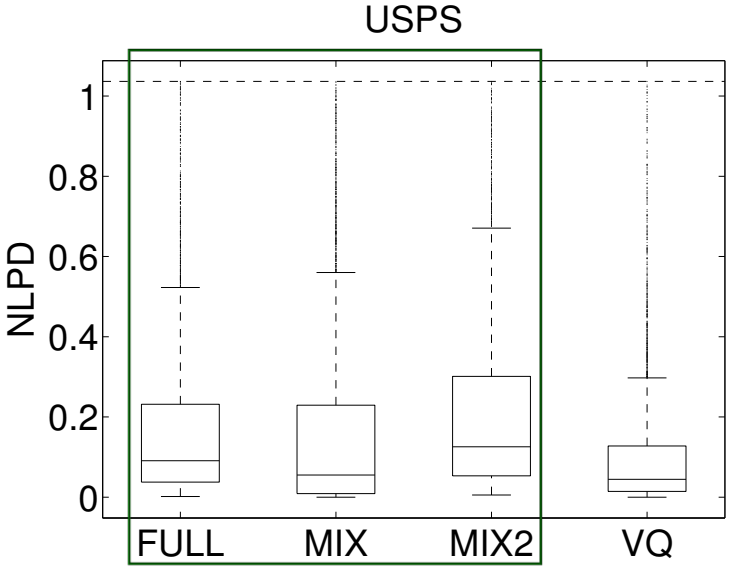


EXPERIMENTS
BINARY AND MULTI-CLASS CLASSIFICATION

Sigmoid and softmax likelihoods





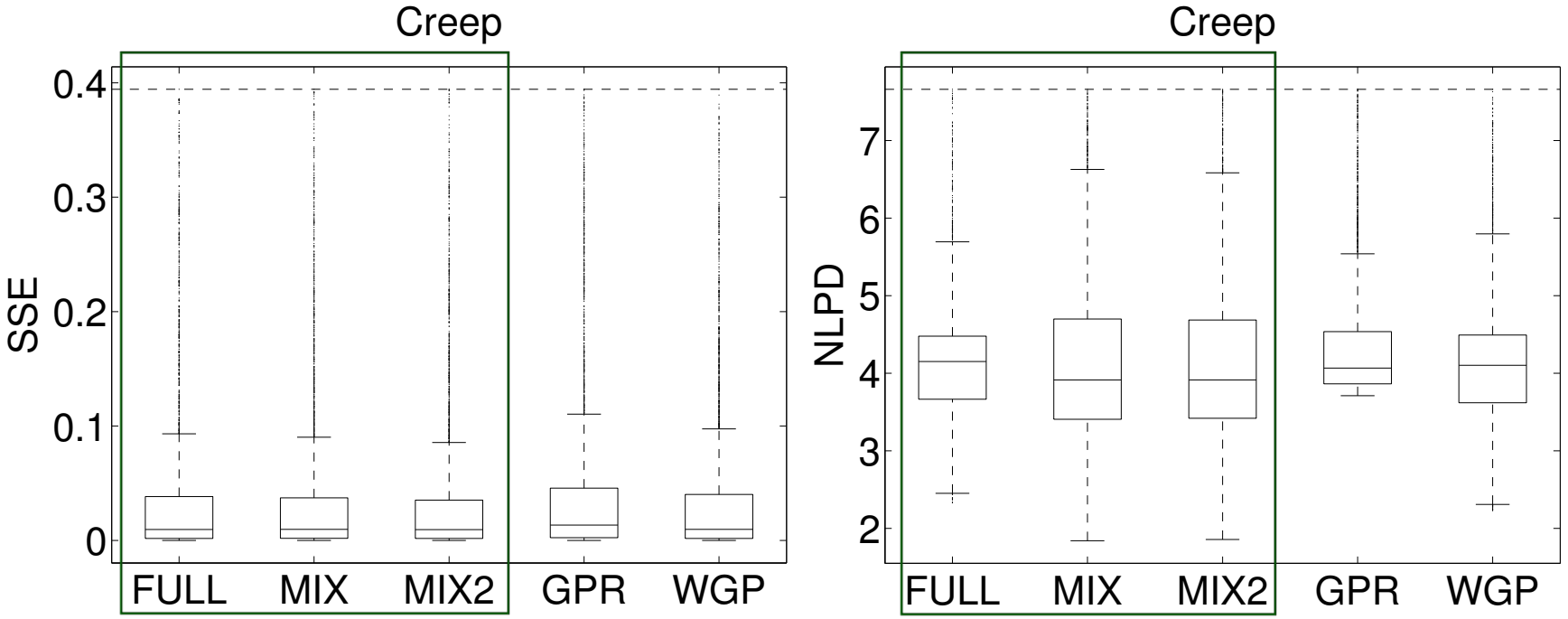


Comparable performance to hard-coded methods

EXPERIMENTS
WARPED GAUSSIAN PROCESSES

Likelihood: $p(y|f) = \nabla_y t(y) \mathcal{N}(t(y); f, \sigma^2)$

t(y): Non-linear monotonic transformation



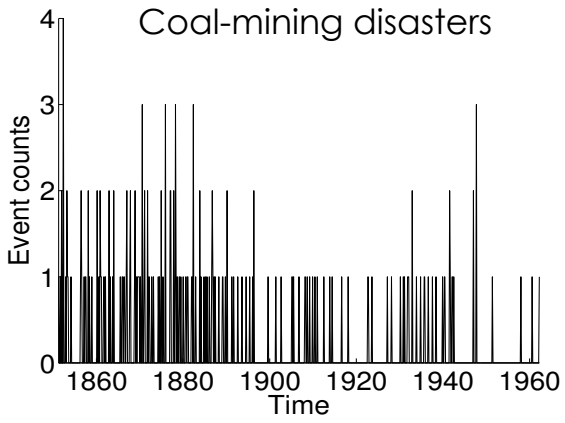
- Comparable performance to exact method WGP
  - GPR has narrower ranges of predictive variances

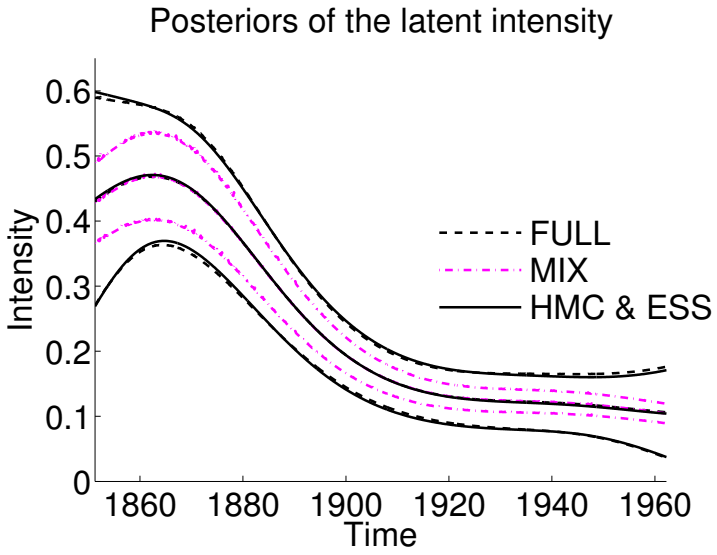
EXPERIMENTS
LOG GAUSSIAN COX PROCESS

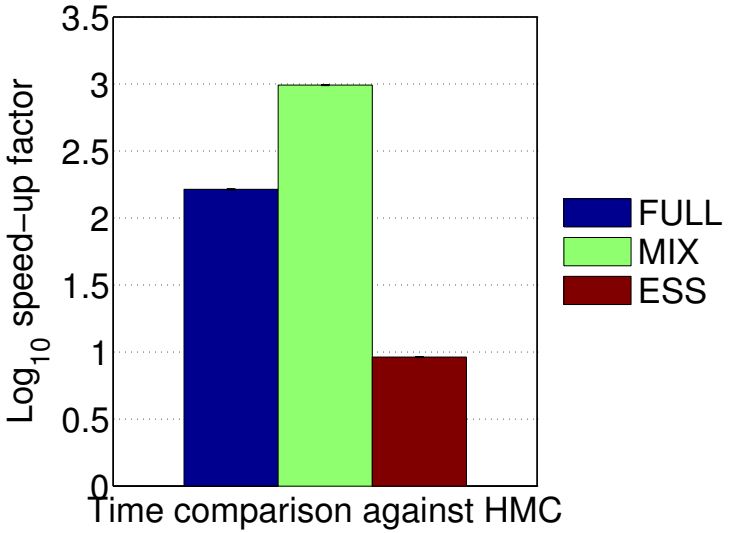
Likelihood:

$$p(y_n|f_n) = \frac{\lambda_n^{y_n} \exp(-\lambda_n)}{y_n!}$$

where $\lambda_n = \exp(f_n + m)$







Same performance as sampling, orders of magnitude faster

Conclusions

Conclusions and Future Work

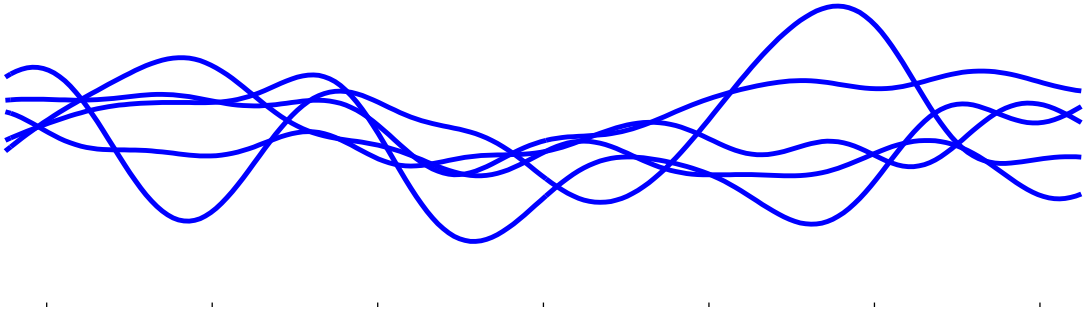
- Automated Variational Inference for GPs
  - Valuable tool for GP practitioners and researchers to investigate new models
    - Minimal effort in deriving and coding model-specific inference algorithms
    - Only requires calls to a 'black-box' likelihood function
- Scaling up to very large datasets
  - See Dezfouli and Bonilla (NIPS 2015)

"The world is a graphical model, sometimes well-represented by a GP"

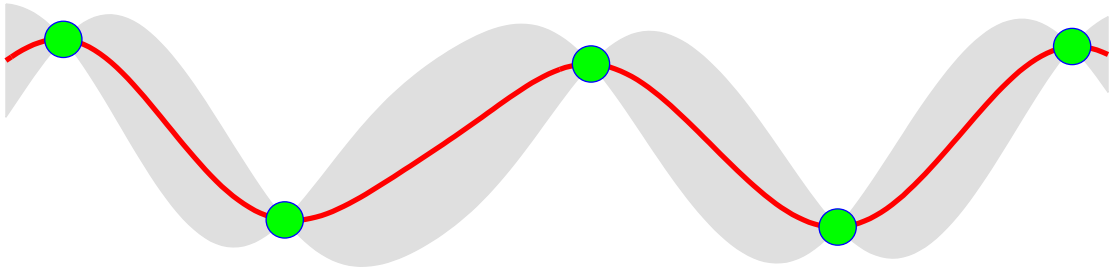
EVB

DEMO

PRIOR



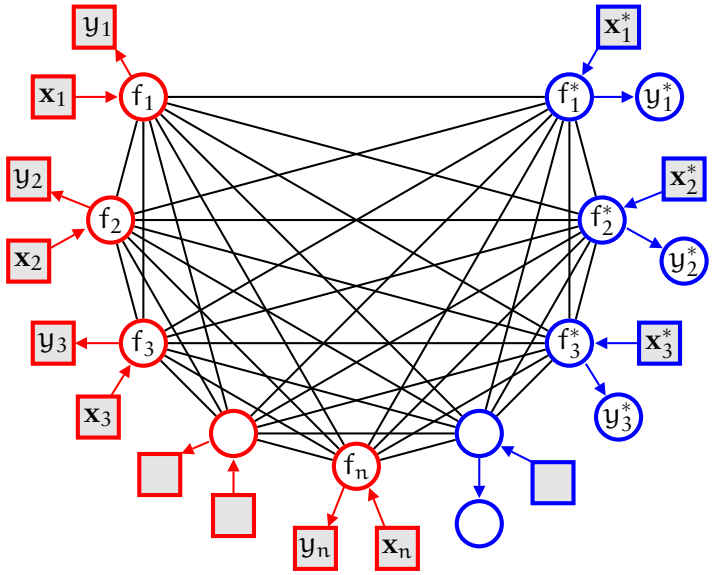
DATA AND POSTERIOR



demo_gp_prior.m

The Standard GP Regression Setting

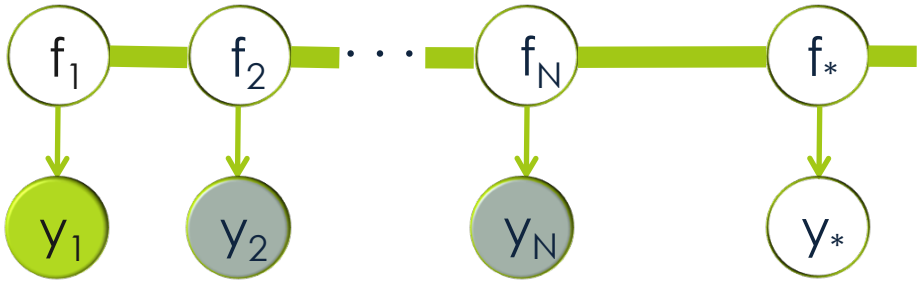
- Data: $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}; \quad \mathbf{x} \in \mathbb{R}^D, \; y \in \mathbb{R}$
- Input: $(\mathbf{X})_{D \times N}$ Targets: $(\mathbf{y})_{N \times 1}$



- Model
  - Prior: $f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \kappa(\mathbf{x}, \mathbf{x}'))$
  - Likelihood: $y_i = f(\mathbf{x}_i) + \epsilon_i,$ $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$
- Tasks:
  - Prediction: $p(\mathbf{f}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*)$
  - Hyper-parameter learning: $\boldsymbol{\theta}$ and σ^2
- Graphical model for GPs?

Inference in Standard GP Regression

- GP prior: $\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_{\mathbf{f},\mathbf{f}} & \mathbf{K}_{\mathbf{f},*} \\ \mathbf{K}_{*,\mathbf{f}} & \mathbf{K}_{*,*} \end{bmatrix}\right)$
- Likelihood: $\mathbf{y}|\mathbf{f} \sim \mathcal{N}\left(\mathbf{f}, \sigma^2\mathbf{I}\right)$



- Posterior predictive: $p(\mathbf{f}_*|\mathbf{y}) = \frac{1}{p(\mathbf{y})}\int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}_*,\mathbf{f})d\mathbf{f} = \mathcal{N}\left(\mathbf{f}_*; \mathbf{K}_{*,\mathbf{f}}(\tilde{\mathbf{K}}_{\mathbf{f},\mathbf{f}})^{-1}\mathbf{y}, \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{f}}(\tilde{\mathbf{K}}_{\mathbf{f},\mathbf{f}})^{-1}\mathbf{K}_{\mathbf{f},*}\right)$

$$\tilde{\mathbf{K}}_{\mathbf{f},\mathbf{f}} = \mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I}$$

  - Computational cost: $O(N^3)$ in time and $O(N^2)$ in memory
  - Similarly for hyper-parameter learning
    - Via maximization of the marginal likelihood

A Historical Note

- How old are Gaussian processes (GPs)?
  a) 1970s
  b) 1950s
  c) 1940s
  d) 1880s

Thorvald Nicolai Thiele

[T. N. Thiele, 1880] *"Om Anvendelse af mindste Kvadraters Methode i nogle Tilfælde, hvor en Komplikation af visse Slags uensartede tilfælde Fejlkilder giver Fejlene en 'systematisk' Karakter",* Vidensk. Selsk. Skr. 5. rk, naturvid. og mat. Afd., 12, 5, 381– 40.

- First mathematical theory of Brownian motion
- EM algorithm (Dempster et al, 1977)?

Approximate Bayesian Inference
Variational Inference

Latent Dirichlet Allocation (Blei et al, 2003)

Finally, we expand Eq. (14) in terms of the model parameters (α, β) and the varia
(γ, ϕ). Each of the five lines below expands one of the five terms in the bound:

$$L(\gamma,\phi;\alpha,\beta) = \log\Gamma\left(\sum_{j=1}^{k}\alpha_j\right) - \sum_{i=1}^{k}\log\Gamma(\alpha_i) + \sum_{i=1}^{k}(\alpha_i-1)\left(\Psi(\gamma_i) - \Psi\left(\right.\right.$$
$$+ \sum_{n=1}^{N}\sum_{i=1}^{k}\phi_{ni}\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right)\right)$$
$$+ \sum_{n=1}^{N}\sum_{i=1}^{k}\sum_{j=1}^{V}\phi_{ni}w_n^j\log\beta_{ij}$$

$$L_{[\gamma]} = \sum_{i=1}^{k}(\alpha_i-1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right)\right) + \sum_{n=1}^{N}\phi_{ni}\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right)\right)$$
$$-\log\Gamma\left(\sum_{j=1}^{k}\gamma_j\right) + \log\Gamma(\gamma_i) - \sum_{i=1}^{k}(\gamma_i-1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right)\right).$$

This simplifies to:

$$L_{[\gamma]} = \sum_{i=1}^{k}\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right)\right)\left(\alpha_i + \sum_{n=1}^{N}\phi_{ni} - \gamma_i\right) - \log\Gamma\left(\sum_{j=1}^{k}\gamma_j\right) + \log\Gamma(\gamma_i).$$

We take the derivative with respect to γ_i:

$$\frac{\partial L}{\partial\gamma_i} = \Psi'(\gamma_i)\left(\alpha_i + \sum_{n=1}^{N}\phi_{ni} - \gamma_i\right) - \Psi'\left(\sum_{j=1}^{k}\gamma_j\right)\sum_{j=1}^{k}\left(\alpha_j + \sum_{n=1}^{N}\phi_{nj} - \gamma_j\right).$$

Setting this equation to zero yields a maximum at:

$$\gamma_i = \alpha_i + \sum_{n=1}^{N}\phi_{ni}. \qquad (17)$$

Since Eq. (17) depends on the variational multinomial ϕ, full variational inference requires alternating between Eqs. (16) and (17) until the bound converges.

A.4.1 Conditional multinomials

To maximize with respect to β, we isolate terms and add Lagrange multipliers:

$$L_{[\beta]} = \sum_{d=1}^{M}\sum_{n=1}^{N_d}\sum_{i=1}^{k}\sum_{j=1}^{V}\phi_{dni}w_{dn}^j\log\beta_{ij} + \sum_{i=1}^{k}\lambda_i\left(\sum_{j=1}^{V}\beta_{ij} - 1\right).$$

We take the derivative with respect to β_{ij}, set it to zero, and find:

$$\beta_{ij} \propto \sum_{d=1}^{M}\sum_{n=1}^{N_d}\phi_{dni}w_{dn}^j.$$

A.4.2 Dirichlet

The terms which contain α are:

$$L_{[\alpha]} = \sum_{d=1}^{M}\left(\log\Gamma\left(\sum_{j=1}^{k}\alpha_j\right) - \sum_{i=1}^{k}\log\Gamma(\alpha_i) + \sum_{i=1}^{k}\left((\alpha_i-1)\left(\Psi(\gamma_{di}) - \Psi\left(\sum_{j=1}^{k}\gamma_{dj}\right)\right)\right)\right)$$

Taking the derivative with respect to α_i gives:

$$\frac{\partial L}{\partial\alpha_i} = M\left(\Psi\left(\sum_{j=1}^{k}\alpha_j\right) - \Psi(\alpha_i)\right) + \sum_{d=1}^{M}\left(\Psi(\gamma_{di}) - \Psi\left(\sum_{j=1}^{k}\gamma_{dj}\right)\right)$$

This derivative depends on α_j, where $j \neq i$, and we therefore must use an iterative method to find the maximal α. In particular, the Hessian is in the form found in Eq. (10):

$$\frac{\partial L}{\partial\alpha_i\alpha_j} = \delta(i,j)M\Psi'(\alpha_i) - \Psi'\left(\sum_{j=1}^{k}\alpha_j\right),$$
$$\frac{\partial L}{\partial\phi_{ni}} = \Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right) + \log\beta_{iv} - \log\phi_{ni} - 1 + \lambda.$$

o zero yields the maximizing value of the variational parameter ϕ_{ni} (cf. Eq. 6):

$$\phi_{ni} \propto \beta_{iv}\exp\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^{k}\gamma_j\right)\right). \qquad (16)$$

Automated Variational Inference
Potential Impact

- What makes the framework practical?
  - Exact gradients of the GP hyper-parameters
  - Approximation using samples of univariate Gaussians
    - Variance reduction techniques (e.g. control variates)
  - Linear parameterization of the covariance matrices
  - *Standard off-the-shelf optimisation tools can be used*

- Other methods under-utilise the information available to GP models:
  - Ranganath et al (AISTATS, 2014)
    - Variation reduction techniques and stochastic optimisation
  - Sampling methods:
    - Pure black-box, ESS, HMC