Sparse GP inference

Gaussian Process Inference using sparse approximations

Lehel Csató

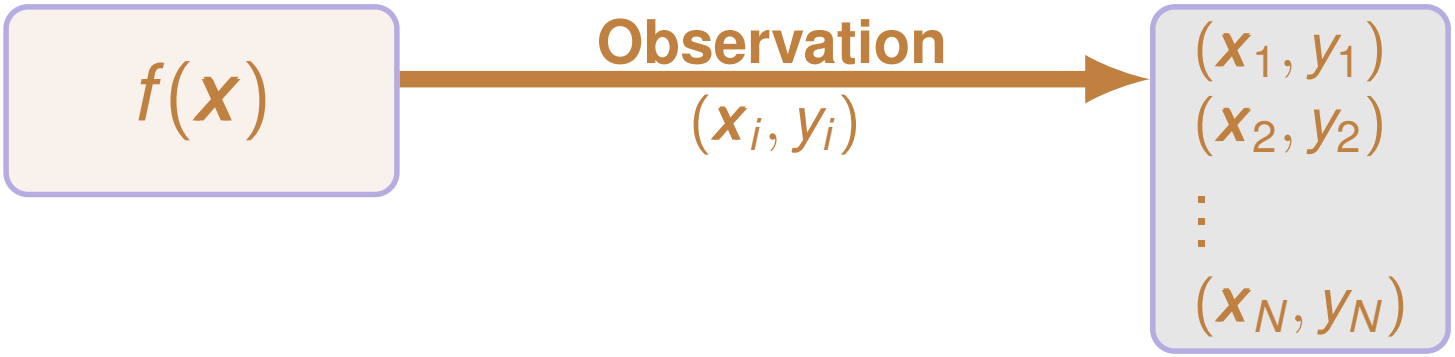
Faculty of Mathematics and Informatics
Babeş–Bolyai University, Cluj-Napoca,

9 October 2018

Table of Contents

Contents

Modelling Data

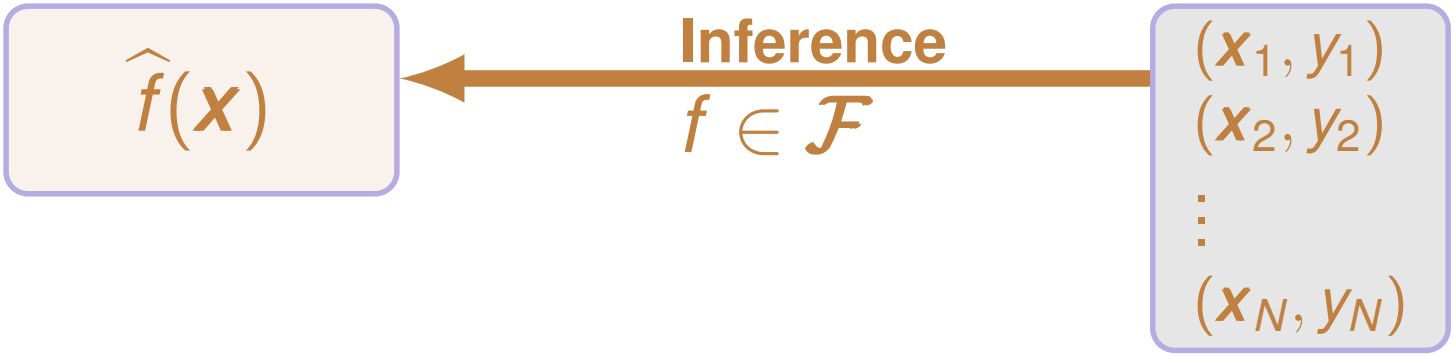


- Real world: there "is" a function $y = f(x)$
- Observation process: a corrupted datum is collected for a sample x_n:

$$\begin{aligned} t_n &= y_n + \epsilon && \text{additive noise} \\ t_n &= h(y_n, \epsilon) && h \text{ distortion function} \end{aligned}$$

- **Problem: find function** $y = f(x)$

Latent variable models



- **Data set** – collected.
- We **have to** assume:
  - a function class (polynomial, Fourier, Wavelet);
  - Observation process that encodes the noise;
- **Goal:** Find the optimal function from the class.

Latent variable models II

- We have the **data set** $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$.
- We assume a function class:
  1. Linear or generalised linear models,
  2. Fourier expansions up to a given order.
- Assume an observation (or generation) process, eg:

$$\begin{aligned} y_n &= f(\boldsymbol{x}_n|\theta) + \epsilon \quad \text{with} \quad \epsilon \sim N(0, \sigma^2) \\ y_n &= \operatorname{sign}(f(\boldsymbol{x}|\theta) + \epsilon) \end{aligned}$$

Latent variables

Since $f(\boldsymbol{x}|\theta)$ is never seen, they are latent functions.

Contents

- $\mathcal{GP}$ models
  $\equiv$ functional latent variable models.
  $\equiv$ "simple" random functions.

- Appear in the likelihood:

$$P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}}) = \prod_i P(y_i|\boldsymbol{x}_i, \boldsymbol{f}) = \prod_i P(y_i|\boldsymbol{x}_i, f_{\boldsymbol{x}_i})$$

Local dependencies only: $\boldsymbol{f} \longrightarrow f_{\boldsymbol{x}}$

Gaussian processes I

Gaussian process: a "generalised" Gaussian distribution.

- $\boldsymbol{f}$ Gaussian random function.

$$\boldsymbol{f} = [f_{\boldsymbol{x}_1}, f_{\boldsymbol{x}_2}, \ldots, f_{\boldsymbol{x}_N}, \ldots]^T, \quad \boldsymbol{x}_n \in \text{domain}.$$

- $\mathcal{GP}$ prior $p_0(\boldsymbol{f})$ characterised with
  - mean function $\langle f_{\boldsymbol{x}} \rangle_0$,
  - covariance kernel $K_0(\boldsymbol{x}, \boldsymbol{x}')$.

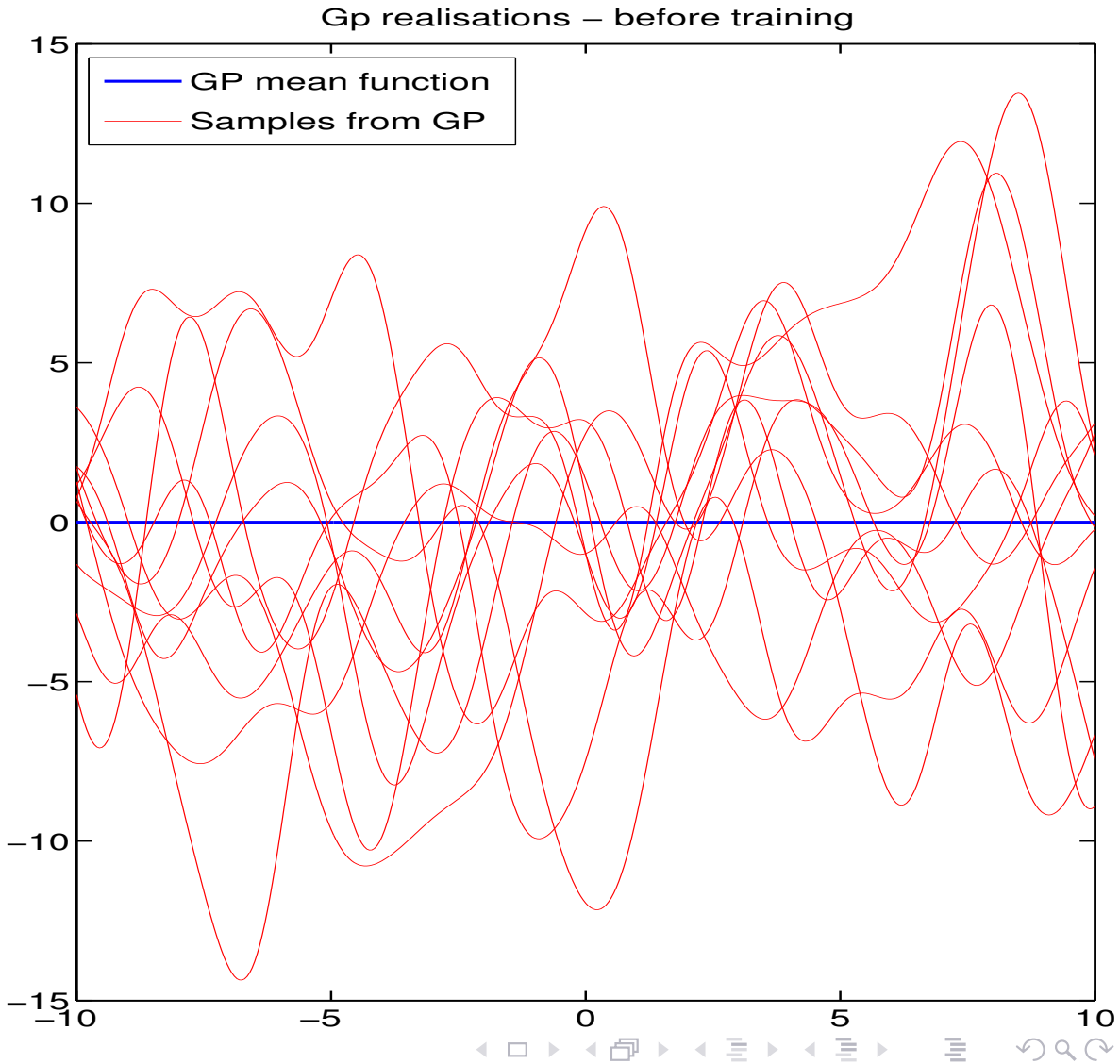
Property - for any sample set $\mathcal{D}$, a joint Gaussian r.v.:

$$\boldsymbol{f}_{\mathcal{D}} = [f_{\boldsymbol{x}_1}, \ldots, f_{\boldsymbol{x}_N}] \sim \mathcal{N}(\boldsymbol{f}_{\mathcal{D}} | \langle f_{\mathcal{D}} \rangle_0, \boldsymbol{K}_0)$$

Gaussian process: random function

"Parameters"

- mean function
- covariance kernel



Gaussian process parameters:

- mean function – usually is 0.
- the kernel function, usually containing other parameters,
  - Example:
    $K(\boldsymbol{x}, \boldsymbol{x}'|\boldsymbol{\theta}) = \exp\left[\theta_0 - \sum_{i=1}^{d} \theta_i \left(x_i - x_i'\right)^2 / 2\right]$
  - Always generate valid covariance matrices.
    $\forall\ \mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$,
    $\boldsymbol{K}_{\mathcal{X}\mathcal{X}} = [K(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1}^{T}$
    is a positive definite matrix.

Gaussian Process Inference I

- $\mathcal{GP}$ inference is the application of Bayes' rule.

$$p_{\text{post}}(\boldsymbol{f}, \boldsymbol{f}_{\mathcal{D}}) \propto P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}})\; p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f})$$

- For any set $\mathcal{X}$ of inputs, the posterior distribution is:

$$p_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}) = \frac{1}{P(\mathcal{D})} \int d\boldsymbol{f}_{\mathcal{D}}\; P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}})\; p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f}_{\mathcal{X}})$$

where

$$P(\mathcal{D}) = \int d\boldsymbol{f}_{\mathcal{D}}\; P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}})\; p_0(\boldsymbol{f}_{\mathcal{D}})$$

is the data probability.

Problem: the posterior process cannot be analytically represented – not a $\mathcal{GP}$.

Motivating the approximations to the posterior

- **If likelihood non–Gaussian** $\Rightarrow$ posterior is not analytic.
  (No “summarising” statistics)
- Non–Gaussian posterior process: we have to build approximations to them.
- Representation – How to represent the posterior?
  Must be finite;

A Lemma for posterior processes

Property of Gaussian averages:

$$\langle f_{\boldsymbol{x}} \rangle_{\text{post}} = \langle f_{\boldsymbol{x}} \rangle_0 + \sum_i K_0(\boldsymbol{x}, \boldsymbol{x}_i)\alpha(i)$$

where $\alpha(i) = \partial_{\langle f_i \rangle_0} \ln P(\mathcal{D})$

Parametrisation lemma

We have a parametrisation similar to the Kimeldorf-Wahba parametrisation lemma, augmented with the kernel:

A Lemma for posterior processes

Property of Gaussian averages:

$$\langle f_{\boldsymbol{x}} \rangle_{\text{post}} = \langle f_{\boldsymbol{x}} \rangle_0 + \sum_i K_0(\boldsymbol{x}, \boldsymbol{x}_i)\alpha(i)$$

where $\alpha(i) = \partial_{\langle f_i \rangle_0} \ln P(\mathcal{D})$

Parametrisation lemma

We have a parametrisation similar to the Kimeldorf-Wahba parametrisation lemma, augmented with the kernel:

$$K_{\text{post}}(\boldsymbol{x}, \boldsymbol{x}') = K_0(\boldsymbol{x}, \boldsymbol{x}') + \sum_{i,j} K_0(\boldsymbol{x}, \boldsymbol{x}_i) C(ij) K_0(\boldsymbol{x}_j, \boldsymbol{x}')$$
$$\text{with} \quad C(ij) = \partial_{\langle f_i \rangle_0, \langle f_j \rangle_0} \ln P(\mathcal{D})$$

Methods to approximate the posterior

Obtaining the posterior approximation:

- Sampling;
- Using modes – Laplace approximation;
- KL-based approximation – Expectation-propagation.

▸ skip others

Sampling from the posterior I

Sampling

$$p_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}) = \frac{1}{Z_{\mathcal{D}}} \int d\boldsymbol{f}_{\mathcal{D}}\; P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}}) p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f}_{\mathcal{X}})$$

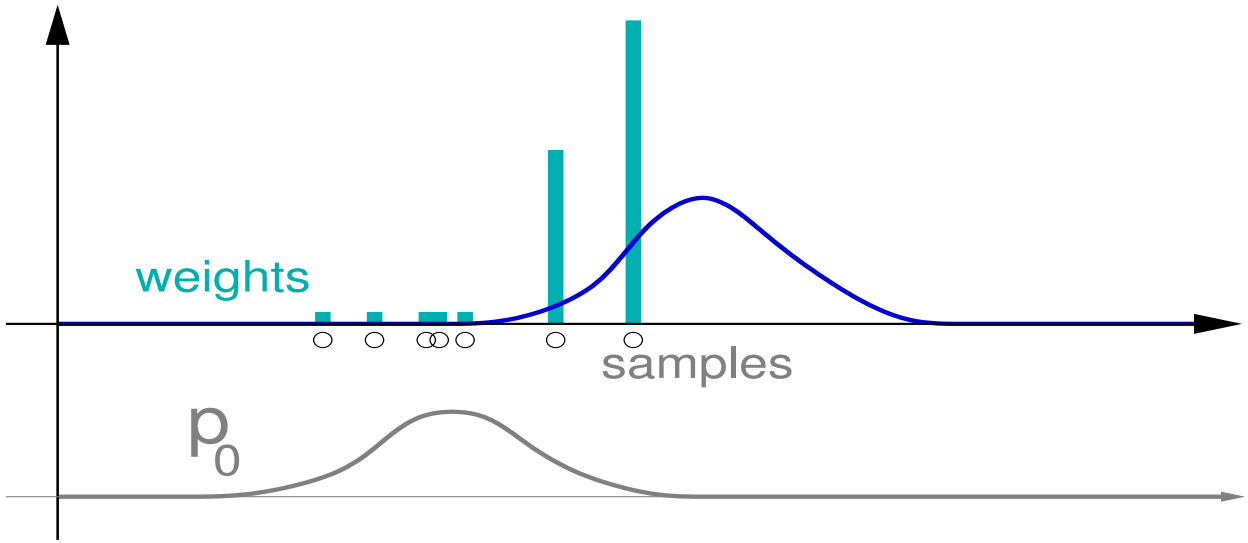
In practise: – joint sampling from $p_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}, \boldsymbol{f}_{\mathcal{D}})$,
– keeping only $\boldsymbol{f}_{\mathcal{X}}$.

Implementation: sampling from $p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f}_{\mathcal{X}})$ + weighting:

$$p_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}) \approx \frac{1}{C_T} \sum_{t=1}^{T} P(\boldsymbol{y}_N|\boldsymbol{f}_{\mathcal{D}}^{(t)})\; \delta(\boldsymbol{f}_{\mathcal{X}} - \boldsymbol{f}_{\mathcal{X}}^{(t)})$$

with $C_T = \sum_{t=1}^{T} P(\boldsymbol{y}_N|\boldsymbol{f}_{\mathcal{D}}^{(t)})$

Sampling from the posterior II



Sampling methods:

- powerful *i.e.* allow flexibility in modelling
- Hard to assess convergence
- Sampling algorithms suited for different models.
- Can be incredibly slow (tempering, MCMC)

Laplace Approximation – I

Log–Posterior:

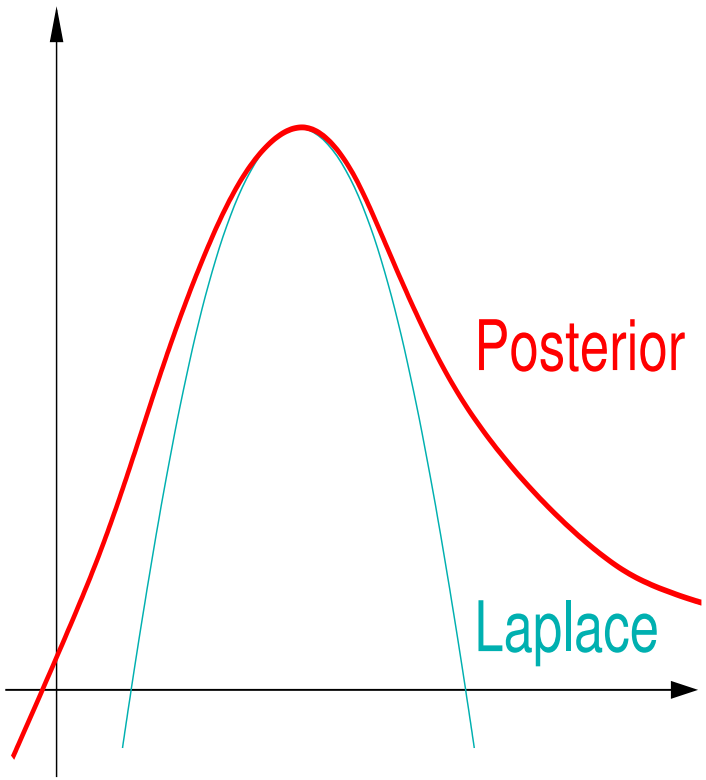
$$\log p_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}, \boldsymbol{f}_{\mathcal{D}}) = K + \log P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}}) + \log p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f}_{\mathcal{X}})$$

Finding maximum of:

$$\hat{\boldsymbol{f}}_{\mathcal{D}} = \arg\max g(\boldsymbol{f}_{\mathcal{D}})$$
$$g(\boldsymbol{f}_{\mathcal{D}}) = \log P(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}}) + \log p_0(\boldsymbol{f}_{\mathcal{D}})$$

Taylor expansion around $\hat{\boldsymbol{f}}_{\mathcal{D}}$:

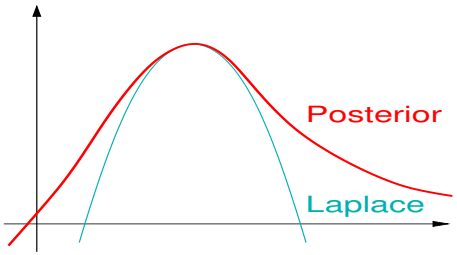
$$\hat{p}_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}, \boldsymbol{f}_{\mathcal{D}}) \propto$$
$$\left(\hat{\boldsymbol{f}}_{\mathcal{D}} - \boldsymbol{f}_{\mathcal{D}}\right)^T \left[H_g(\hat{\boldsymbol{f}}_{\mathcal{D}})\right] \left(\hat{\boldsymbol{f}}_{\mathcal{D}} - \boldsymbol{f}_{\mathcal{D}}\right)$$
$$+ \log\log p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f}_{\mathcal{X}}) - \log p_0(\boldsymbol{f}_{\mathcal{D}})$$



$\Rightarrow$ **Gaussian**

Laplace approximation – II

Log-quadratic approximation:



$$\hat{p}_{\text{post}}(\boldsymbol{f}_{\mathcal{X}}, \boldsymbol{f}_{\mathcal{D}}) \propto \left(\hat{\boldsymbol{f}}_{\mathcal{D}} - \boldsymbol{f}_{\mathcal{D}}\right)^{T} \left[H_g(\hat{\boldsymbol{f}}_{\mathcal{D}})\right] \left(\hat{\boldsymbol{f}}_{\mathcal{D}} - \boldsymbol{f}_{\mathcal{D}}\right) + \log p_0(\boldsymbol{f}_{\mathcal{D}}, \boldsymbol{f}_{\mathcal{X}}) - \log p_0(\boldsymbol{f}_{\mathcal{D}})$$

Defines an approximation to the **likelihood**:

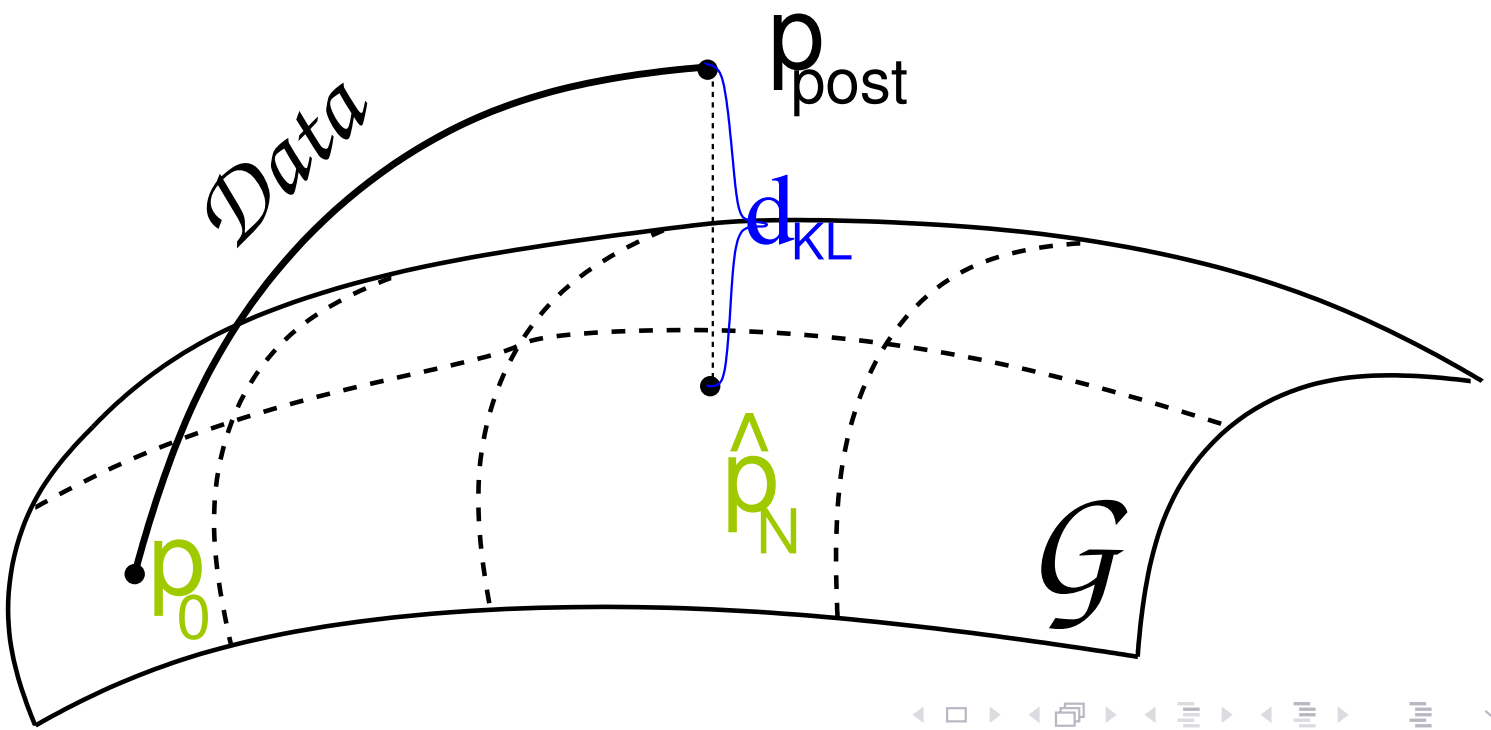
$$\hat{P}(\mathcal{D}|\boldsymbol{f}_{\mathcal{D}}) \propto \left(\hat{\boldsymbol{f}}_{\mathcal{D}} - \boldsymbol{f}_{\mathcal{D}}\right)^{T} \left[H_g(\hat{\boldsymbol{f}}_{\mathcal{D}})\right] \left(\hat{\boldsymbol{f}}_{\mathcal{D}} - \boldsymbol{f}_{\mathcal{D}}\right) - \log p_0(\boldsymbol{f}_{\mathcal{D}})$$

The approximation is **Gaussian**.

Expectation propagation – I

Aim: approximate the posterior distribution – or the posterior process.

$\mathcal{GP}$ **prior** $\longrightarrow$ $\mathcal{GP}$ **approximation to posterior.**
projection – closest $\mathcal{GP}$.



Expectation propagation – II

Choice of projection: Kullback-Leibler divergence

$$\mathrm{KL}(\mathcal{GP}_{\text{post}} \| \mathcal{GP}) = \int d\mathcal{GP}_{\text{post}}(\boldsymbol{f}) \; \log \frac{d\mathcal{GP}_{\text{post}}(\boldsymbol{f})}{d\mathcal{GP}(\boldsymbol{f})}$$

$$\mathcal{GP}^* = \arg\min_{\mathcal{GP}} \mathrm{KL}(\mathcal{GP}_{\text{post}} \| \mathcal{GP})$$

The minimiser:

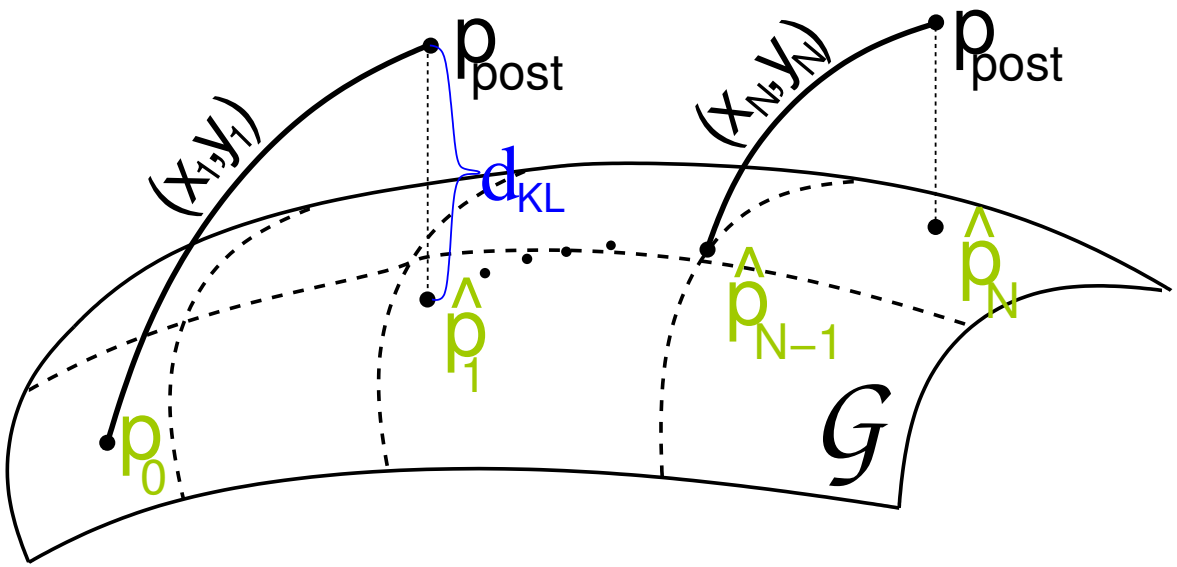
$$\begin{aligned} \langle f_x \rangle_{\mathcal{GP}^*} &\stackrel{\text{def}}{=} \langle f_x \rangle_{\text{post}} \\ K_{\mathcal{GP}^*}(\boldsymbol{x}, \boldsymbol{x}') &\stackrel{\text{def}}{=} K_{\text{post}}(\boldsymbol{x}, \boldsymbol{x}') \end{aligned}$$

Implies that the KL-approximation the $\mathcal{GP} \Leftrightarrow (\boldsymbol{\alpha}_{\mathcal{D}}, \boldsymbol{C}_{\mathcal{D}})$.

Bayesian Online Learning (recursive)

- Instead of $|\mathcal{D}| = N$ uses $\mathcal{D} = (\boldsymbol{x}_{t+1}, y_{t+1})$ and;
- For prior process $\langle f_{\boldsymbol{x}} \rangle_t$, $K_t(\boldsymbol{x}, \boldsymbol{x}')$.

$$\mathrm{KL}(\mathcal{GP}^{t+1}_{\text{post}} \| \mathcal{GP}^*) \quad \text{smaller steps}$$



Expectation propagation – IV

Inference is propagating the mean and the kernel:

$$\begin{aligned}\langle f_{\boldsymbol{x}} \rangle_{t+1} &= \langle f_{\boldsymbol{x}} \rangle_t \quad + q\, K_t(\boldsymbol{x}, \boldsymbol{x}_{t+1}) \\ K_{t+1}(\boldsymbol{x}, \boldsymbol{x}') &= K_t(\boldsymbol{x}, \boldsymbol{x}') + r\, K_t(\boldsymbol{x}, \boldsymbol{x}_{t+1}) K_t(\boldsymbol{x}_{t+1}, \boldsymbol{x}')\end{aligned}$$

with q (and r also) function of the **single** likelihood:
$q = \partial_{\langle f_{t+1} \rangle_t} \ln \langle P(y_{t+1} | f_{t+1}) \rangle_t$

Expectation propagation:

Instead of $\mathcal{GP}_{t+1}$, at each step, an "actual" $\mathcal{GP}_{\setminus \ell}$, with $\ell = t + 1$, is computed.
$\mathcal{GP}_{\setminus \ell} \stackrel{\text{def}}{=} \mathcal{GP}_{\mathcal{D}} / t_\ell(q_\ell, r_\ell)$
It is used as an actual prior in computing the updates.

Consequence: potential to several iterations.

Expectation propagation – V

Expectation propagation:

- Keeps the update parameters q_ℓ and r_ℓ for each data – called "sites".
- Updates the "site parameters" using the steps:
  1. Removing the actual site from the posterior,
  2. Computing the new approximation and inferring the updated site parameters.
- The algorithm runs until convergence.
- It is stable for log-concave likelihood functions – proof?

Unification Perspective

Prediction with Gaussian processes

Given: $\boldsymbol{x}^*$ - for which we require answer y^*.

$$
\begin{aligned}
p(y^*|\boldsymbol{x}^*, \mathcal{D}) &= \int df^* \int d\boldsymbol{f}_{\mathcal{D}}\, p(y^*, \boldsymbol{f}_{\mathcal{D}}, f^*|\boldsymbol{x}^*, \mathcal{D}) \\
&= \int df^*\; P(y^*|\boldsymbol{x}^*, f^*) \int d\boldsymbol{f}_{\mathcal{D}}\; p_{\text{post}}(\boldsymbol{f}_{\mathcal{D}}, f^*|\mathcal{D}) \\
&= \int df^*\; P(y^*|\boldsymbol{x}^*, f^*) p_{\text{post}}(f^*|\mathcal{D})
\end{aligned}
$$

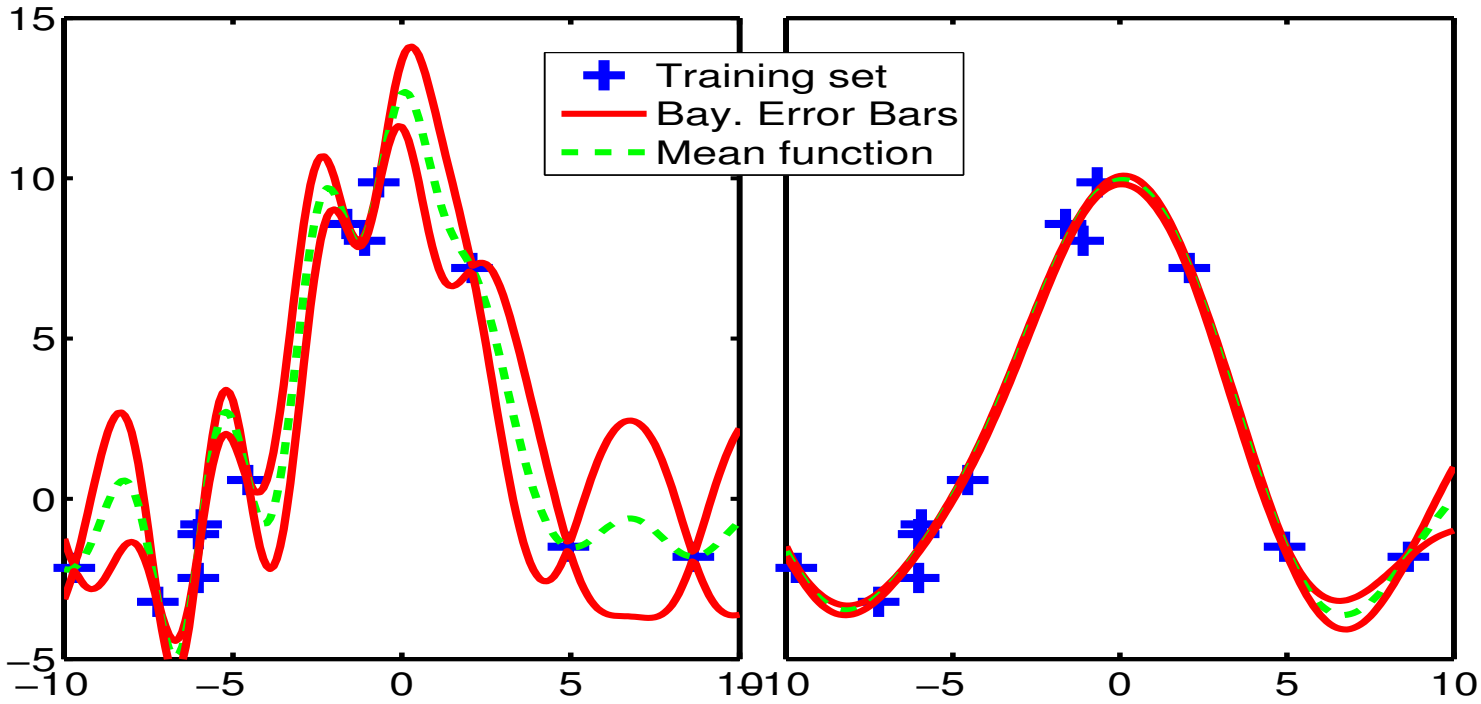
where $f^* = f_{\boldsymbol{x}^*}$ – random variable "at" $\boldsymbol{x}^*$.

Optimising hyper-parameters I

$\mathcal{GP}$ **kernel** parameters $\Leftrightarrow$ model choice.

Exemplu:

RBF kernel: $K(\boldsymbol{x}, \boldsymbol{x}') = A \exp\left[-\sum (x - x')^2 \beta\right]$



$\beta = 0.1$ $\beta = 1.$

Optimising hyper-parameters II

Model evidence:

$$Z_{\mathcal{D}}(\boldsymbol{\theta}) = P(\mathcal{D}|\boldsymbol{\theta}) = \int d\mathbf{f}_{\mathcal{D}}\, P(\mathcal{D}|\mathbf{f}_{\mathcal{D}})\, p_0(\mathbf{f}_{\mathcal{D}}|\theta)$$

Maximum Likelihood II inference

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\Omega} P(\boldsymbol{\theta}|\mathcal{D}) = \arg\min_{\boldsymbol{\theta}\in\Omega} \frac{P(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{M})}{p(\mathcal{D}|\mathcal{M})}$$

if $p(\boldsymbol{\theta}|\mathcal{M})$ "flat"

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\Omega} Z_{\mathcal{D}}(\boldsymbol{\theta})$$

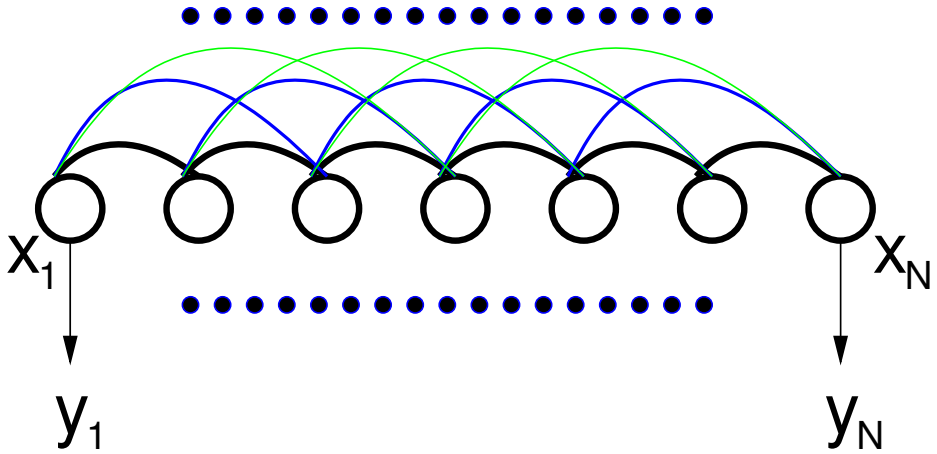
Evidence maximisation.

Gradient/conj.grad. methods are used.

Contents

Sparse representations – Motivation

Gaussian Processes are fully connected graphical models.



$\Longrightarrow$ Computing estimates is difficult. E.g for the posterior mean:

$$\langle f_{\boldsymbol{x}} \rangle_{\text{post}} = \boldsymbol{y}^T \left(\boldsymbol{K}_N + \sigma_o^2 \boldsymbol{I}_N \right)^{-1} \boldsymbol{k}_{\boldsymbol{x}}$$

inversion requires $\mathcal{O}(N^3)$ time.

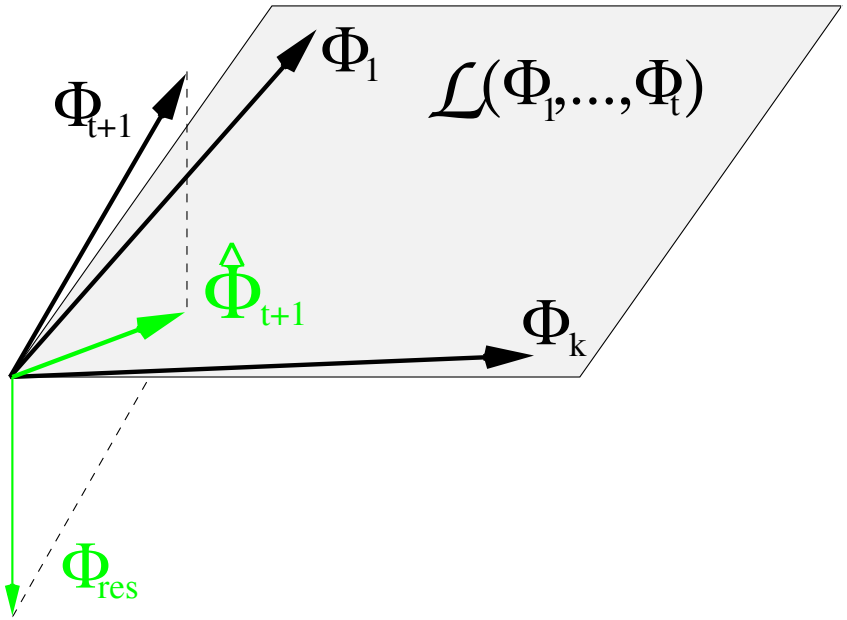
$K_0(\boldsymbol{x}, \boldsymbol{x}')$ defines a *feature space* with $\phi_{\boldsymbol{x}}, \phi_{\boldsymbol{x}'} \in \mathcal{F}$ and $K_0(\boldsymbol{x}, \boldsymbol{x}') = \phi_{\boldsymbol{x}}^T \phi_{\boldsymbol{x}'}$.

Using $\mathcal{F}$ and the scalar product:

$$\langle f_{\boldsymbol{x}} \rangle_{\text{post}} = \phi_{\boldsymbol{x}}^T \sum_{i=1}^{N} \alpha(i) \phi_i \stackrel{\text{def}}{=} \phi_{\boldsymbol{x}}^T \boldsymbol{\mu}_{\text{post}}$$

$$K_{\text{post}}(\boldsymbol{x}, \boldsymbol{x}') = \phi_{\boldsymbol{x}}^T (\boldsymbol{I}_{\mathcal{F}} + \sum_{i,j=1}^{N} \phi_i C(ij) \phi_j^T) \phi_{\boldsymbol{x}'} \stackrel{\text{def}}{=} \phi_{\boldsymbol{x}}^T \boldsymbol{\Sigma}_{\text{post}} \phi_{\boldsymbol{x}'}$$

By identification, this means that $\mathcal{GP}$ inference is the estimation of a Gaussian distribution in $\mathcal{F}$.



Projection

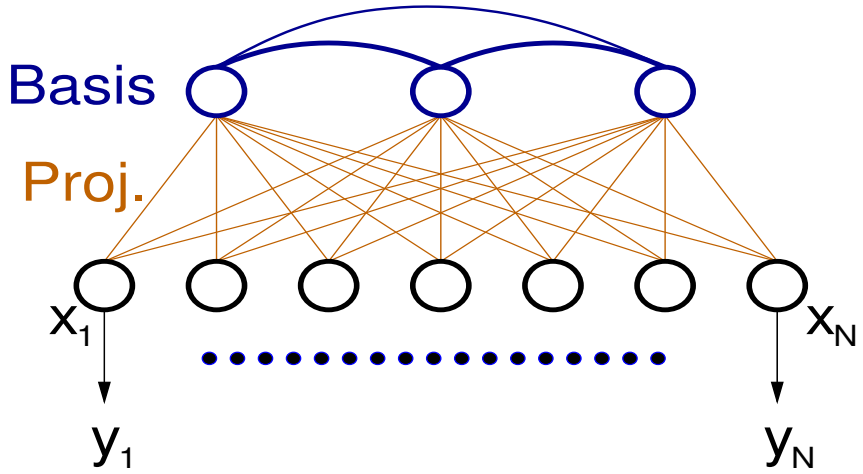
- is on the space of **inducing variables**;
- loss is measured using the KL-divergence between two posteriors:

$$\mathrm{KL}(\mathcal{GP}(\boldsymbol{f}_{\mathcal{X}})\|\mathcal{GP}(\boldsymbol{f}_{\mathcal{X}}|\boldsymbol{f}_{\mathcal{BV}})) \quad (1)$$

Sparse representations – a solution

Condition all training locations on a set of basis locations or inducing variables[1].

$$\boldsymbol{f}_{\mathcal{BV}} \sim \mathrm{N}\left(\mu_{\mathcal{BV}}, \Sigma_{\mathcal{BV}}\right)$$



The pseudo-latents $\boldsymbol{f}_{\mathcal{X}}$ are conditioned on $\boldsymbol{f}_{\mathcal{BV}}$:

$$\boldsymbol{f}_{\mathcal{X}} | \boldsymbol{f}_{\mathcal{BV}} \sim \mathrm{N}\left(\boldsymbol{P}\, \mu_{\mathcal{BV}}, \boldsymbol{P}\, \Sigma_{\mathcal{BV}}\, \boldsymbol{P}^T\right)$$

where $\boldsymbol{P}$ is the projection matrix:

$$\boldsymbol{P} = \boldsymbol{P}_{\boldsymbol{X},\mathcal{BV}} = \boldsymbol{K}_{\boldsymbol{X},\mathcal{BV}}\, \boldsymbol{K}_{\mathcal{BV}}^{-1}$$

[1] Term used by Rasmussen, Candela, Snelson, etc.

Sparse algorithm

The similarity with Gaussian distributions in an Euclidean space allowed to:

- Optimally remove any single component from the basis of expansion;
- Compute a “loss” associated to removing a component;
- Doing it whist performing the other sparsification steps.

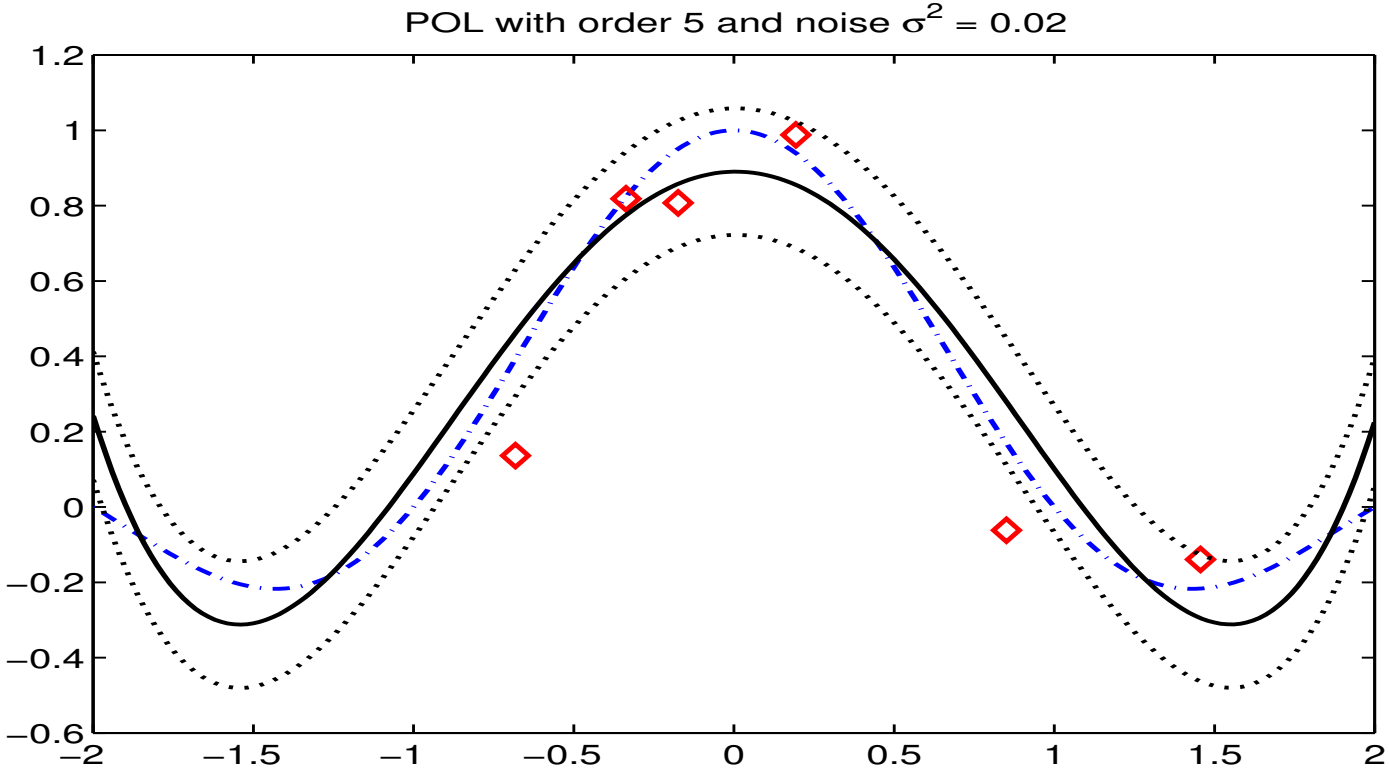
Results

Contents

Gaussian Regression

Artificial data: $y = \sin(x)/x$ and polynomial kernel $K_0(x, x') = (1 + \boldsymbol{x}^T \boldsymbol{x}')^k$.

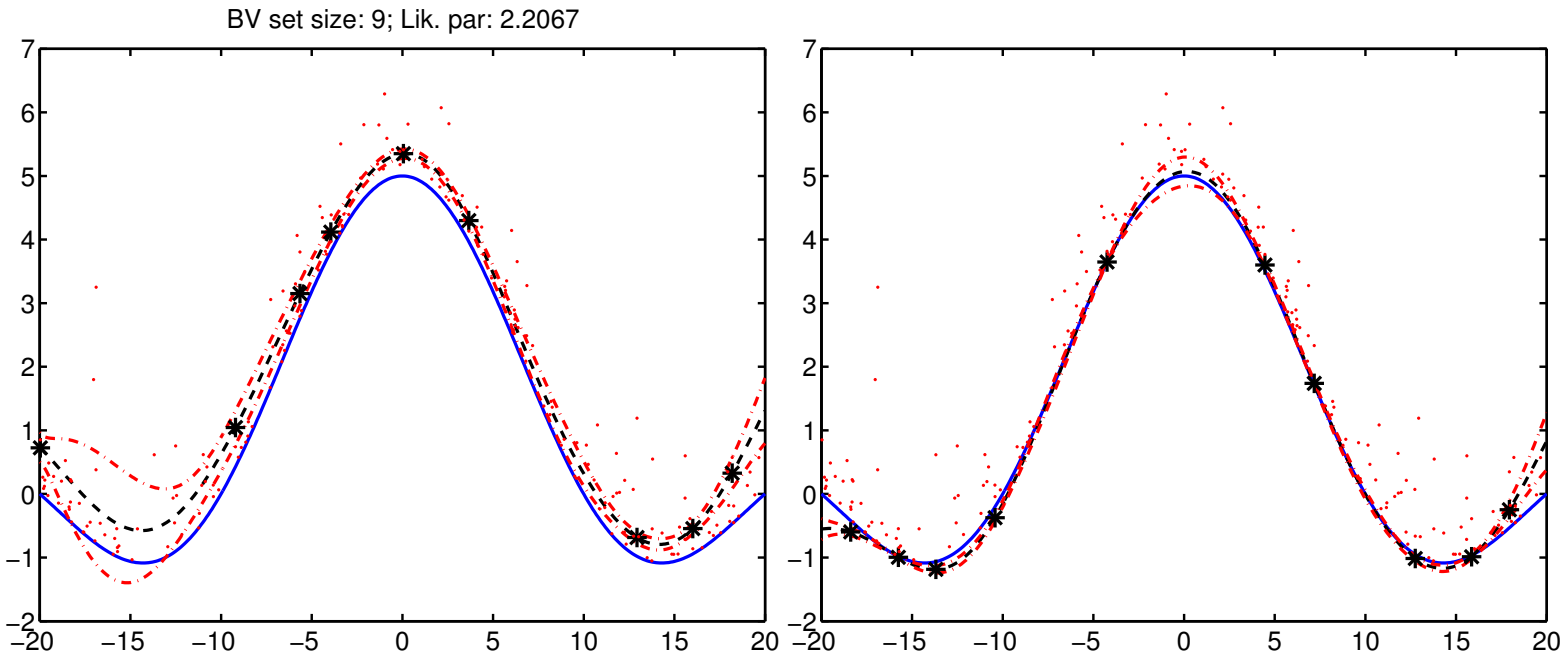
Number of training points: 1000 with added Gaussian noise $\sigma^2 = 0.02$



Robust one-sided regression

Exponential, one-sided, additive noise.

$$P(y|f_{\boldsymbol{x}}) = \begin{cases} \lambda \exp\left[-\lambda(y - f_{\boldsymbol{x}})\right] & \text{if } y > f_{\boldsymbol{x}}. \\ 0 & \text{otherwise.} \end{cases}$$



Classification

For each input $\boldsymbol{x}$ we have ± 1 and we use the probit likelihood:

$$P(y|f(\boldsymbol{x})) = \mathrm{Erf}\left(\frac{y\ f_{\boldsymbol{x}}}{\sigma_0}\right)$$

Erf is the incomplete cumulative Gaussian ($\sim$ sigmoid):

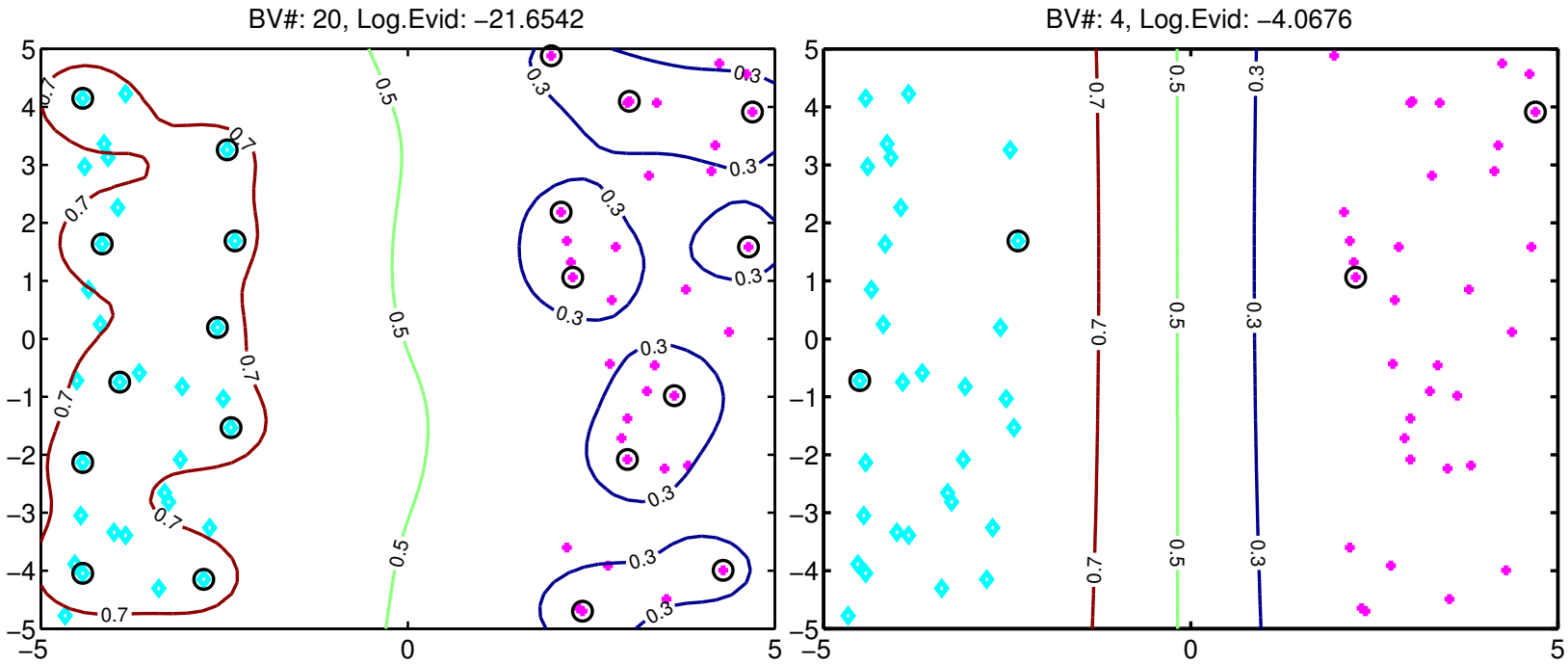
$$\mathrm{Erf}(x) = \int_{-\infty}^{x} dt\ \exp(-t^2/2)/\sqrt{2\pi}$$

- Posterior is not Gaussian.
- For single data, mean-var computable $\Rightarrow$ iterative methods can be used.

Toy Classification

RBF kernel: $K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left[-b - \sum_{i=1}^{d}(x_i - x_i')^2 \beta_i\right]$

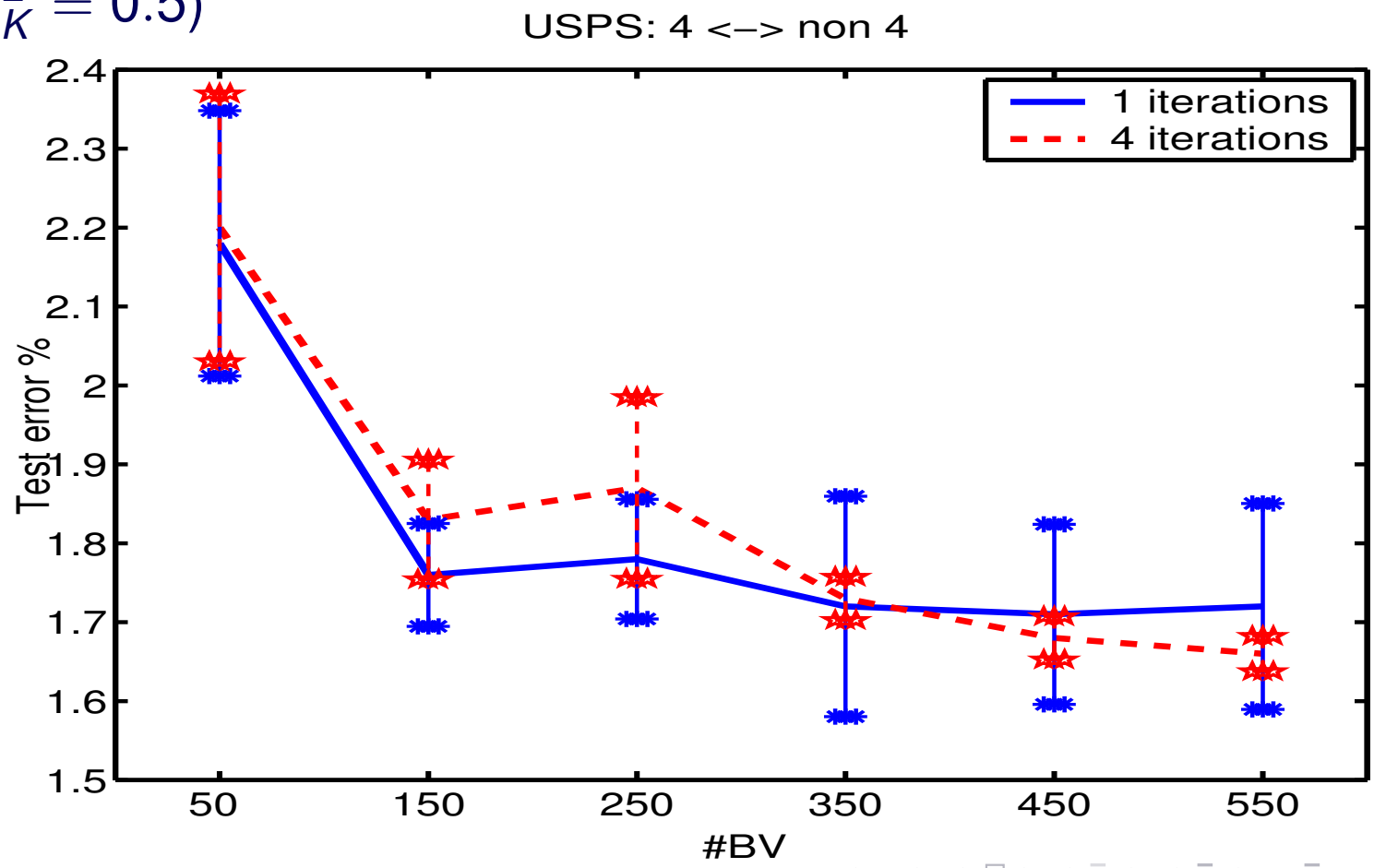
behaviour of the ARD parameters β_i



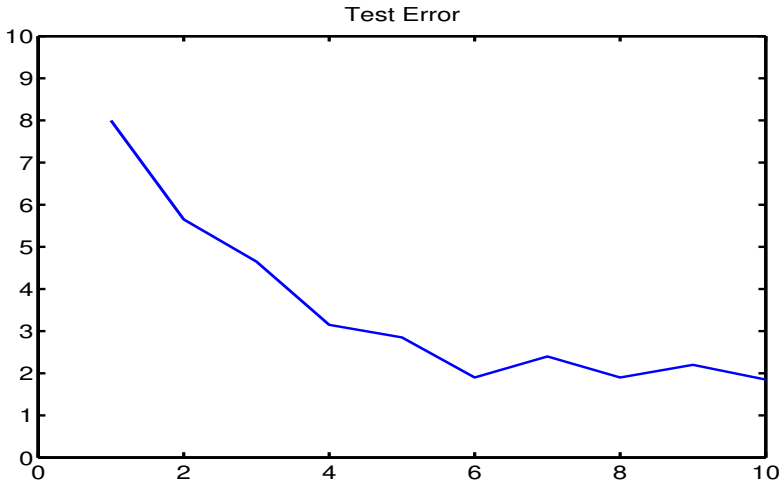
demogp_class_gui

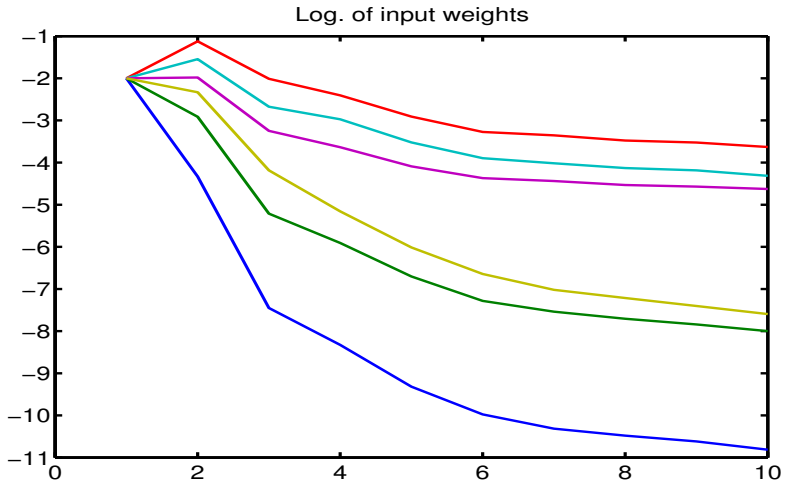
USPS data-set

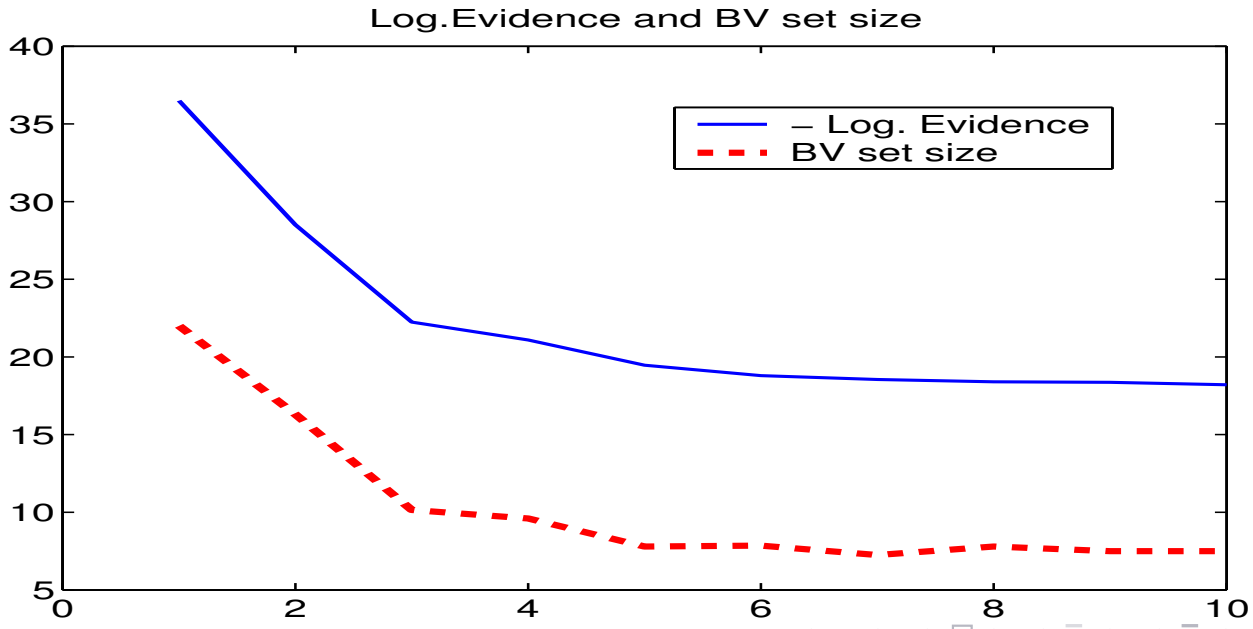
Handwritten image data-set of gray-scale images with 7291 training and 2007 test patterns. (RBF kernel with $\sigma_K^2 = 0.5$)



Using RBF kernels







Multiclass Classification

Problem setup:

- For each location $\boldsymbol{x}$ we have $y \in \{1, \ldots, K\}$.
- Transforming it into $y \in \{0, 1\}^K$ Coding:

$$y = [0, \ldots, 0, 1, 0, \ldots]^T \quad \text{on the } k\text{-th position}$$

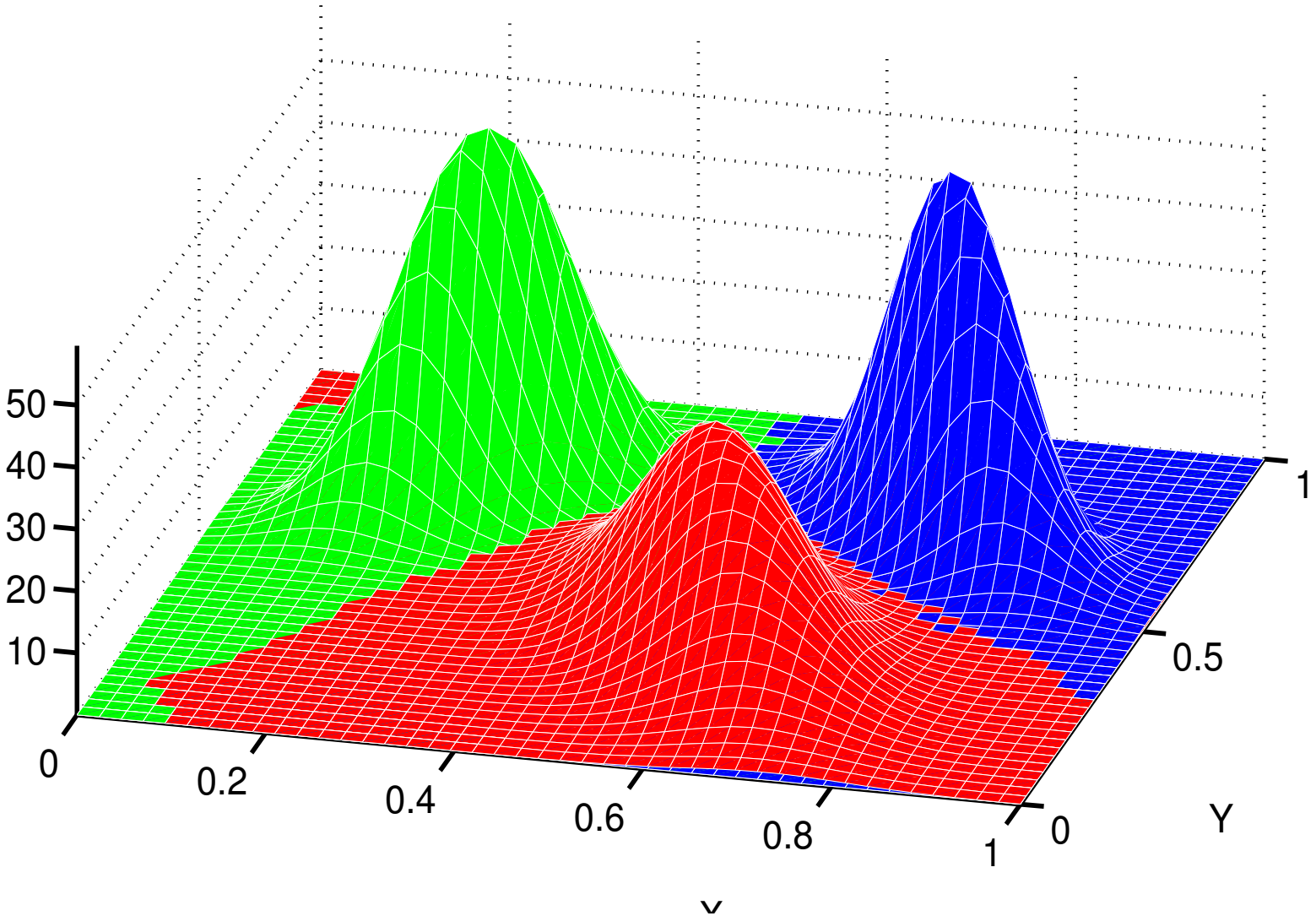
- K independent GP's are used. Indep. is a-priori.
- The likelihood function is:

$$P(y|f(\boldsymbol{x})) = \frac{y^T \boldsymbol{s}}{\mathbf{1}^T \boldsymbol{s}} \text{ where } \boldsymbol{s} = \exp\left([f_1(\boldsymbol{x}), \ldots, f_K(\boldsymbol{x})]^T\right).$$

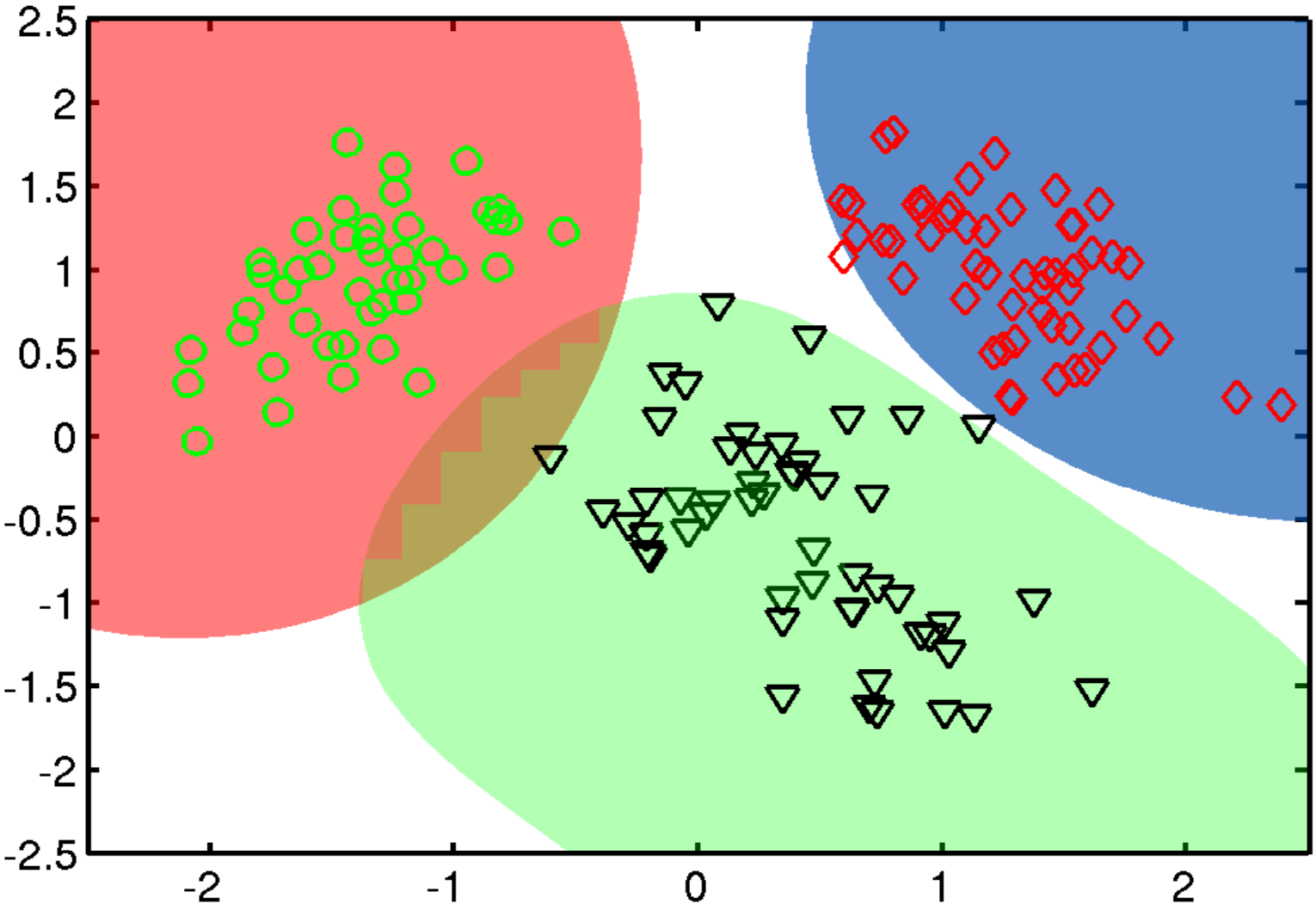
- The posterior processes are not independent.

Multiclass Classification

Two-dimensional demo: class-conditional distributions



Multiclass Classification



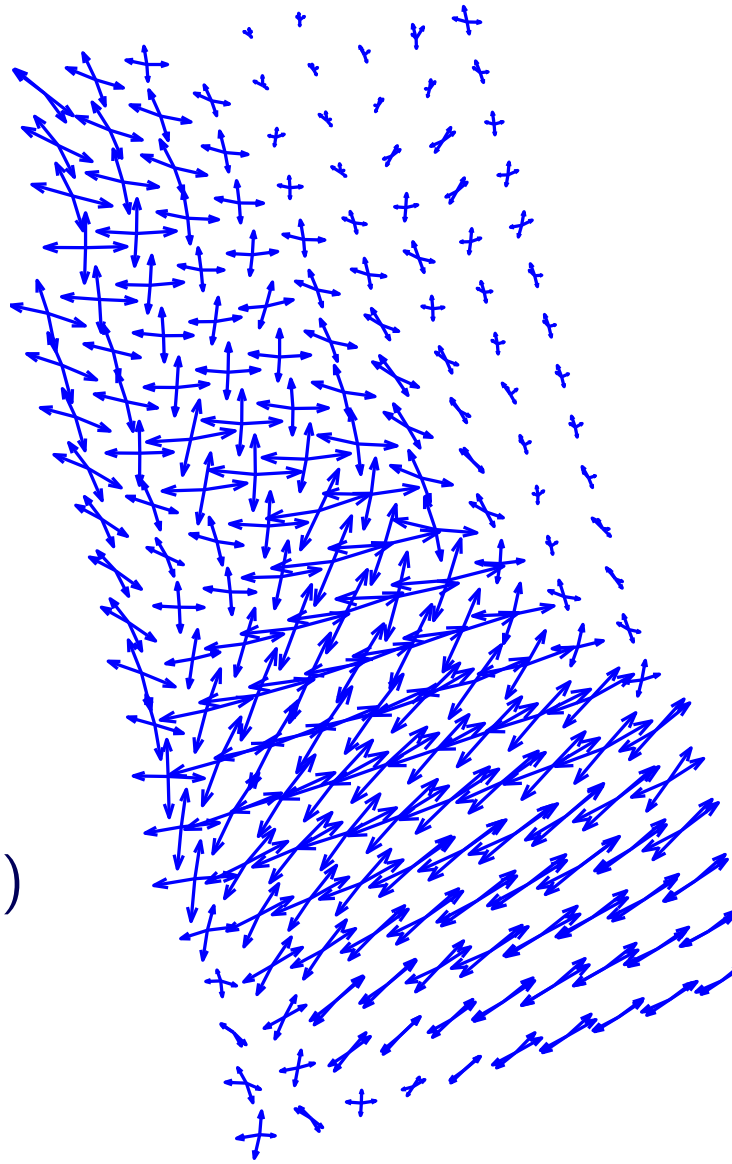
Likelihood:
local observations of global wind-fields (u_i, v_i).

Probabilistic framework preferred due to lack of direct observations.

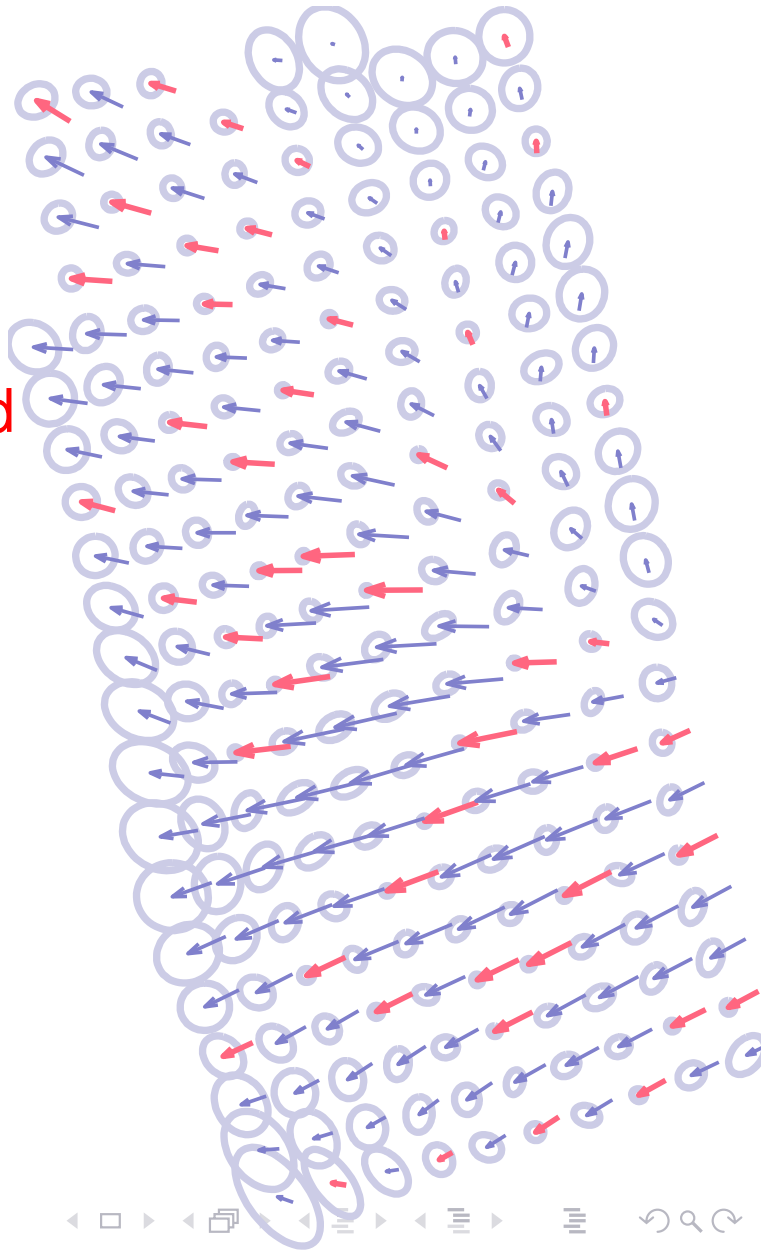
Uncertainty captured in Mixture density networks:

$$P(u_i, v_i|\mathrm{obs}) = \sum_{k=1}^{4} \beta_{il} \mathcal{N}(u_i, v_i|\mu_{il}, A_{il})$$

$\beta_{il}, \mu_{il}, A_{il}$ local parameters.



- Few Inducing Vectors retained
- Approximation preserves information about local uncertainty
- The inference process is fast



Contents

Research questions

- Does the EP algorithm converge?
  - An empirical observation was that for *log-concave* likelihood functions the EP produces the same results.
  - Performance did not depend on the data ordering.
  - *But EP* is a variational approximation.
- Is there a better way to optimise $\mathcal{GP}$ parameters?
  - *Hyper-parameters* govern the quality of sparse $\mathcal{GP}$'s.
  - When $\mathcal{BV}$ set size is restricted, the *locations* also become parameters to the MLII approximation.
  - Is the gradient descent still correct?

It is, but are the results correct?

Depends on the kernels $\Rightarrow$ topology between inputs

Thank you!