Large-Scale Inference in Gaussian Process Models

Edwin V. Bonilla
August 21st, 2014

A Historical Note

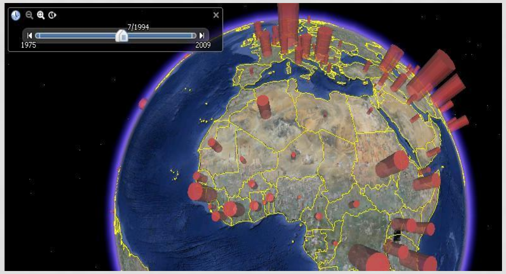
- How old are Gaussian processes (GPs)?
  a) 1970s
  b) 1950s
  c) 1940s
  d) 1880s

Thorvald Nicolai Thiele

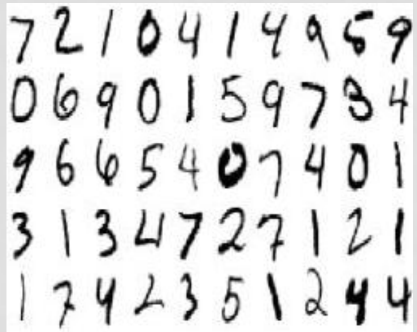
[T. N. Thiele, 1880] *"Om Anvendelse af mindste Kvadraters Methode i nogle Tilfælde, hvor en Komplikation af visse Slags uensartede tilfælde Fejlkilder giver Fejlene en 'systematisk' Karakter"*, Vidensk. Selsk. Skr. 5. rk, naturvid. og mat. Afd., 12, 5, 381– 40.

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

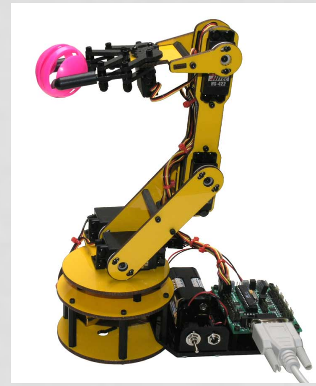
Some Applications of GP Models



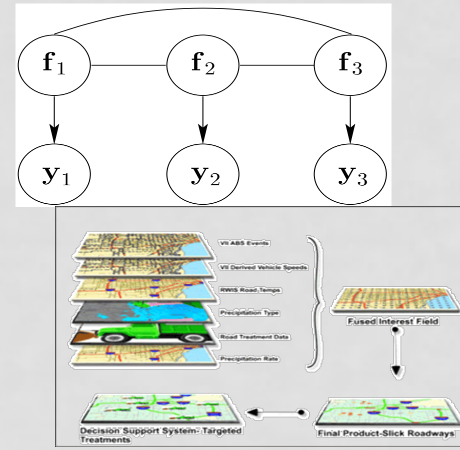
Spatio-temporal modelling



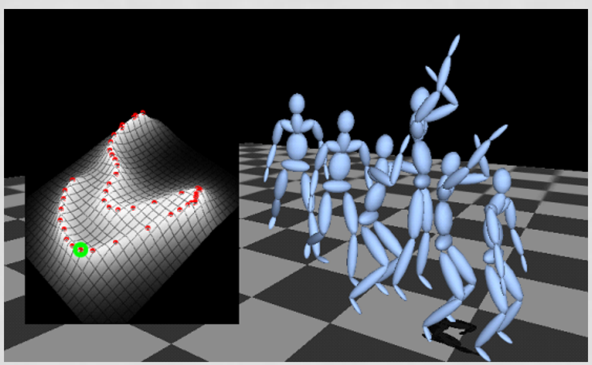
Classification



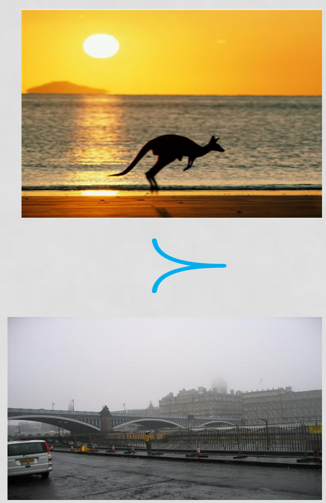
Robot inverse dynamics



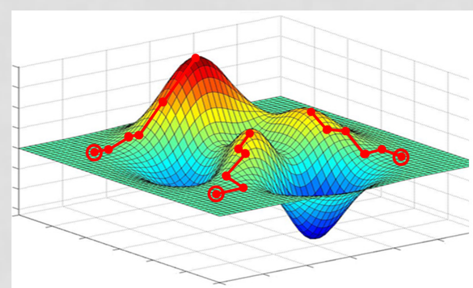
Data fusion / multi-task learning



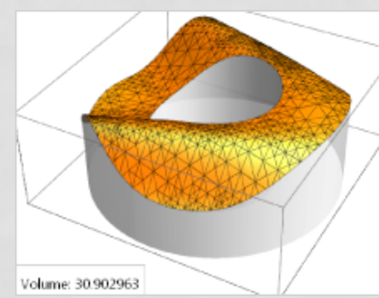
Style-based inverse kinematics



Preference learning



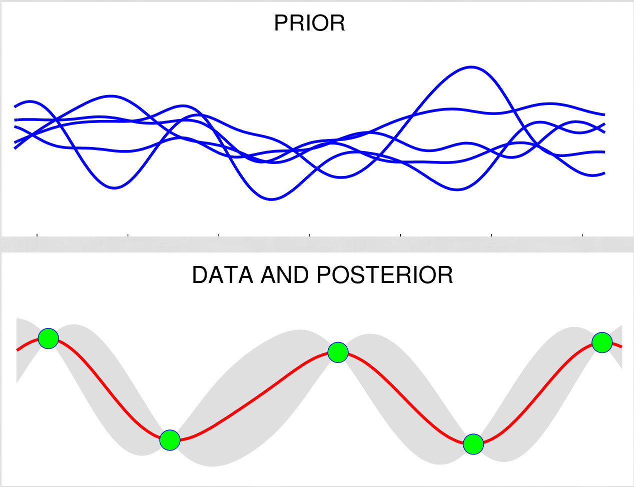
Bayesian optimization



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)

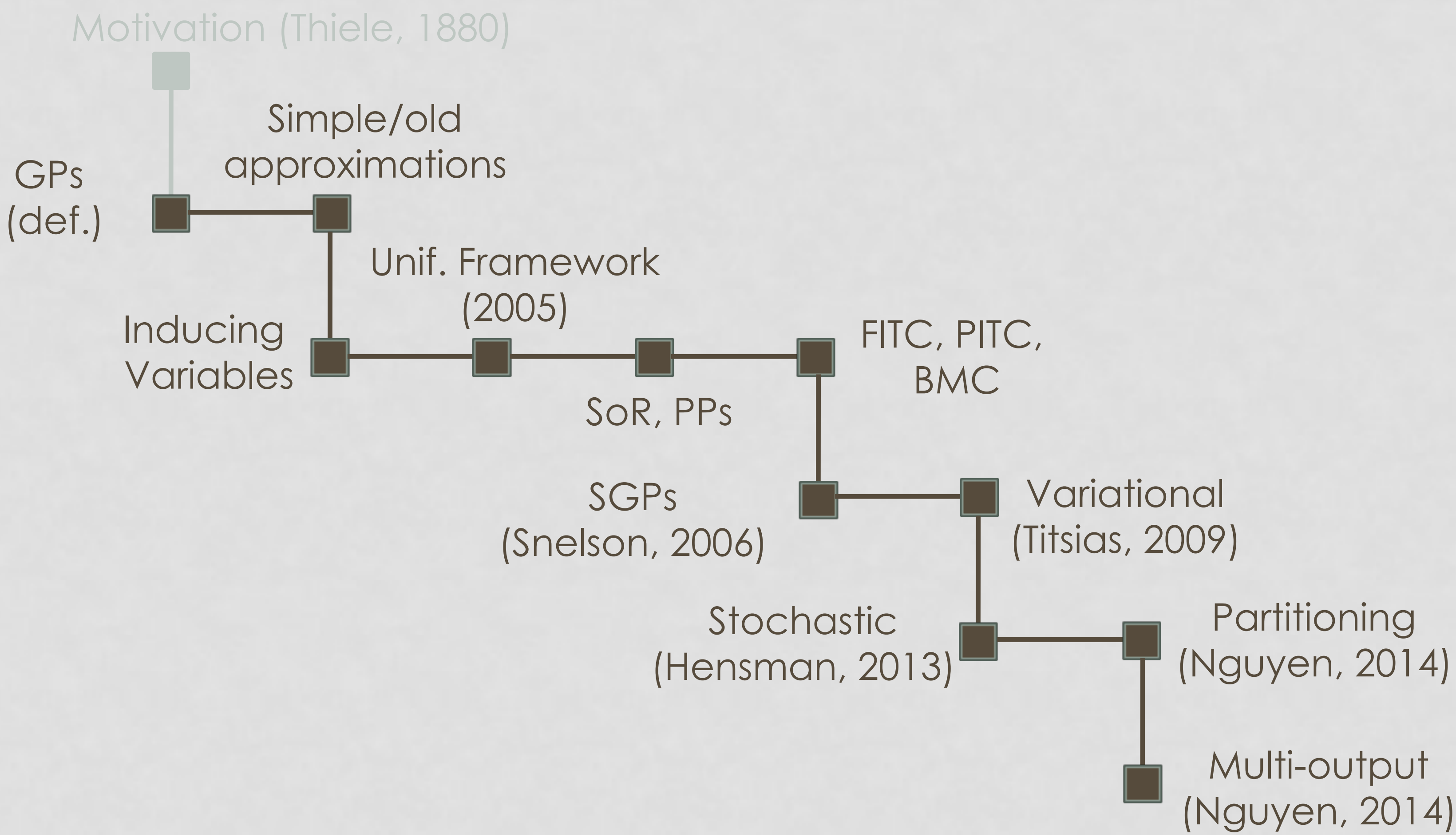


Bayesian non-linear regression

- 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

This talk is about approaches for scalability to large datasets when having Gaussian likelihoods (i.e. regression problems)

This Talk at a Glance: A Journey through GP approximations



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

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

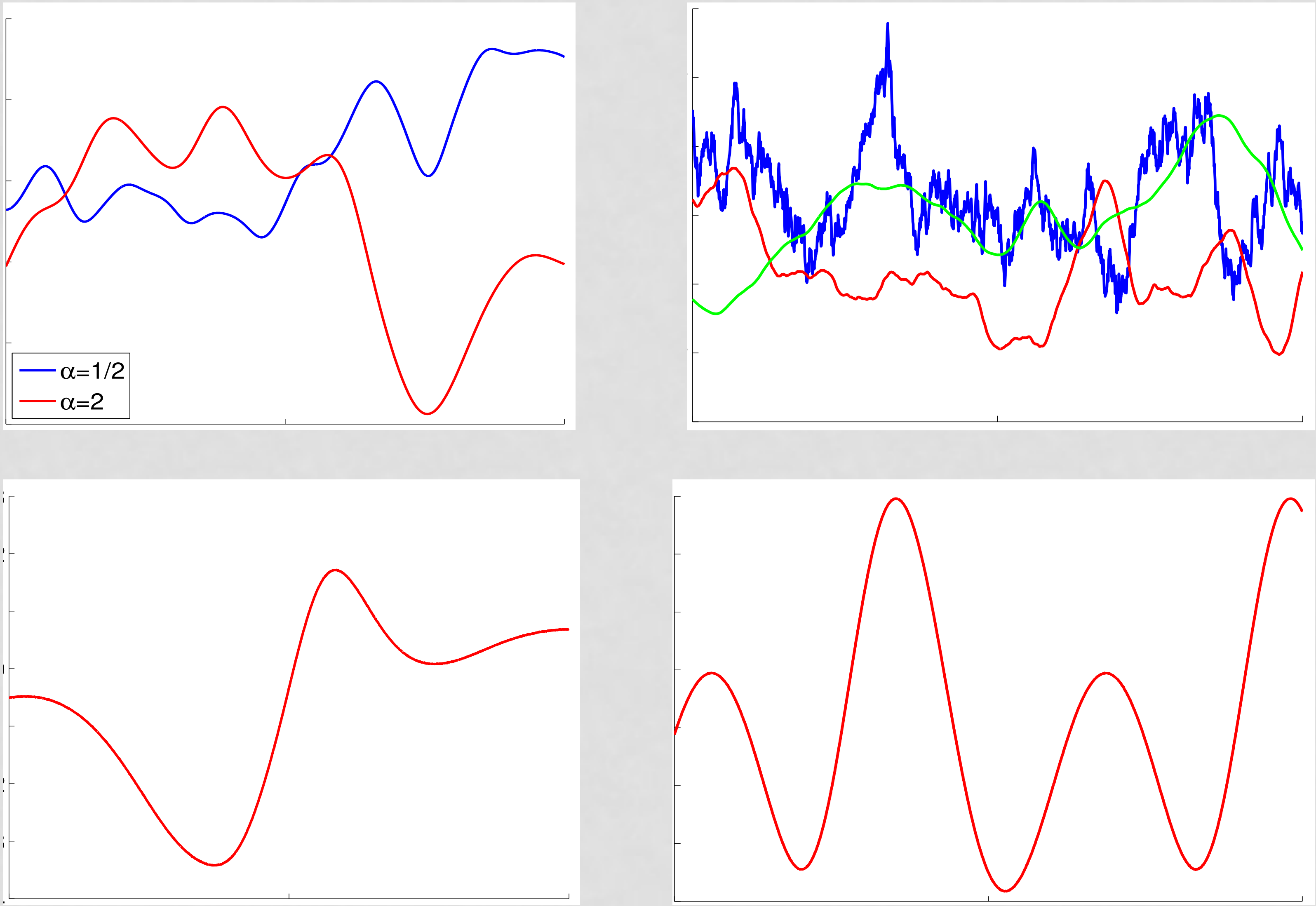
$$\kappa(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}) = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))]$$

Covariance function

Hyper-parameters

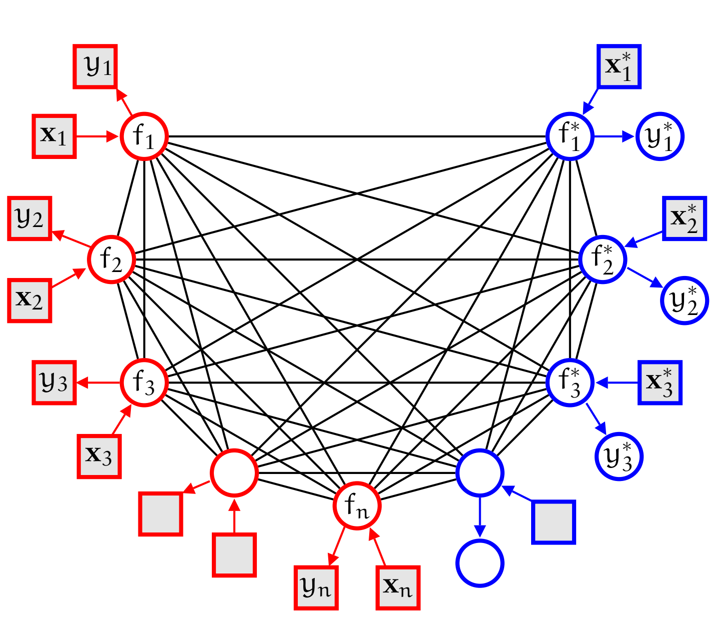
- A GP is a distribution over functions
  - There is not such a thing as the GP method

Samples from a Gaussian Process



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: $(\mathbf{y})_{N \times 1}$

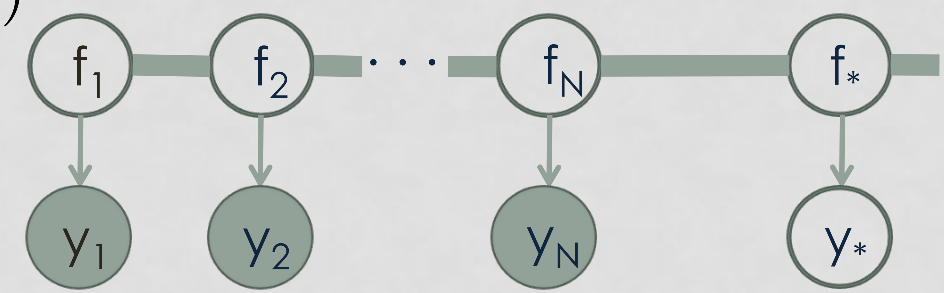


- Model
  - Prior: $f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \kappa(\mathbf{x}, \mathbf{x}'))$
  - Likelihood: $y_i = f(\mathbf{x}_i) + \epsilon_i,$
    $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$
- Tasks:
  - Prediction: $p(\mathbf{f}_* | \mathbf{y}, \mathbf{X}, \mathbf{X}_*)$
  - Hyper-parameter 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: $\mathbf{y}|\mathbf{f} \sim \mathcal{N}\left(\mathbf{f}, \sigma^2 \mathbf{I}\right)$



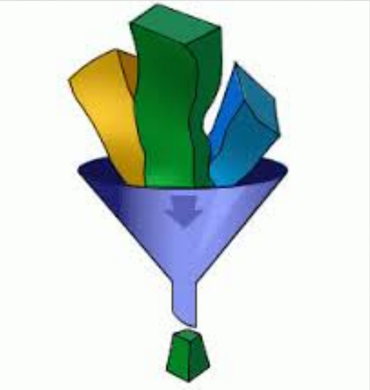
- Posterior predictive:

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

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

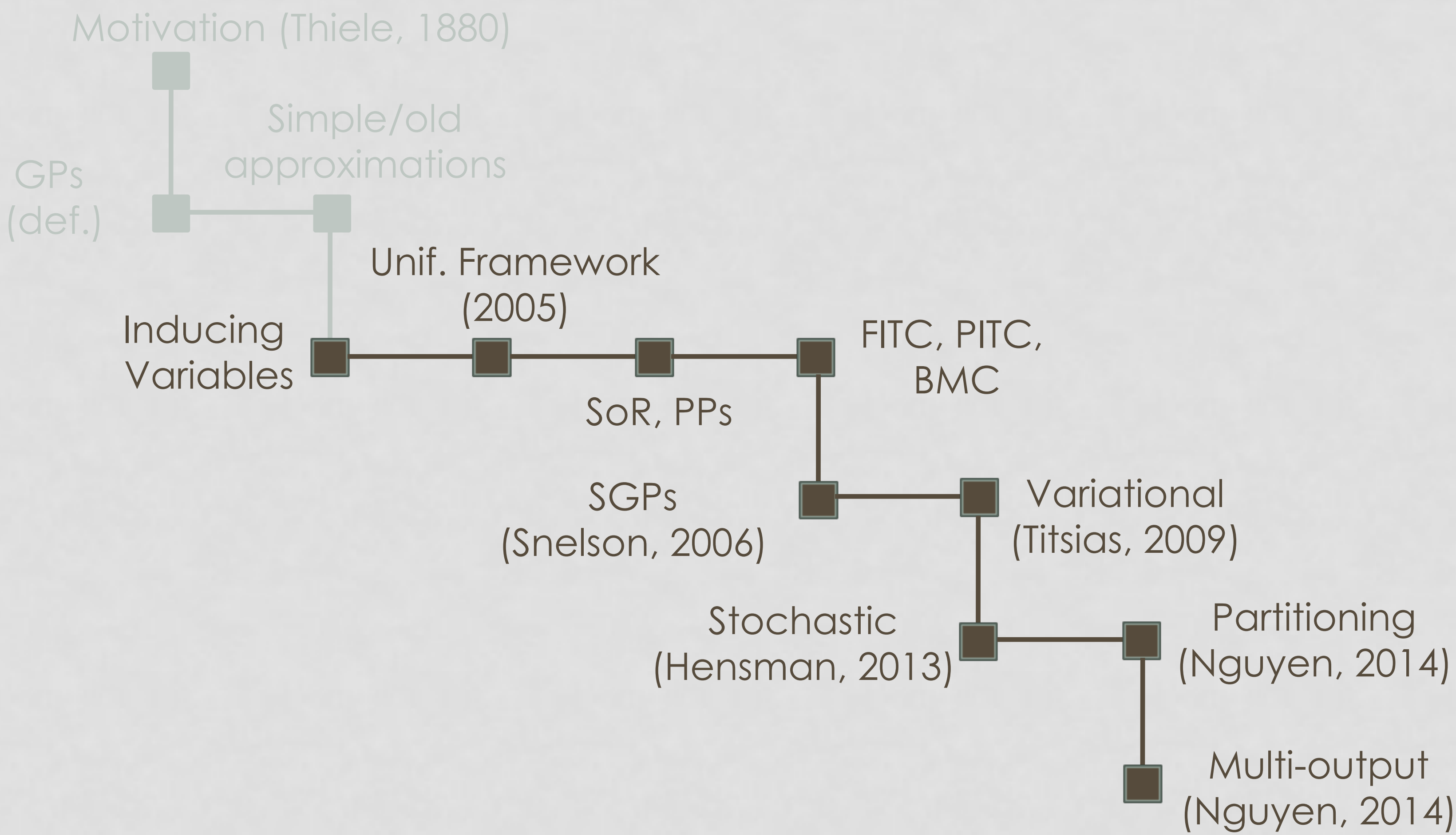
  - Computational cost: $O(N^3)$ in time and $O(N^2)$ in memory
  - Similarly for hyper-parameter learning
    - Via maximization of the marginal likelihood

Simple / Old Approximations

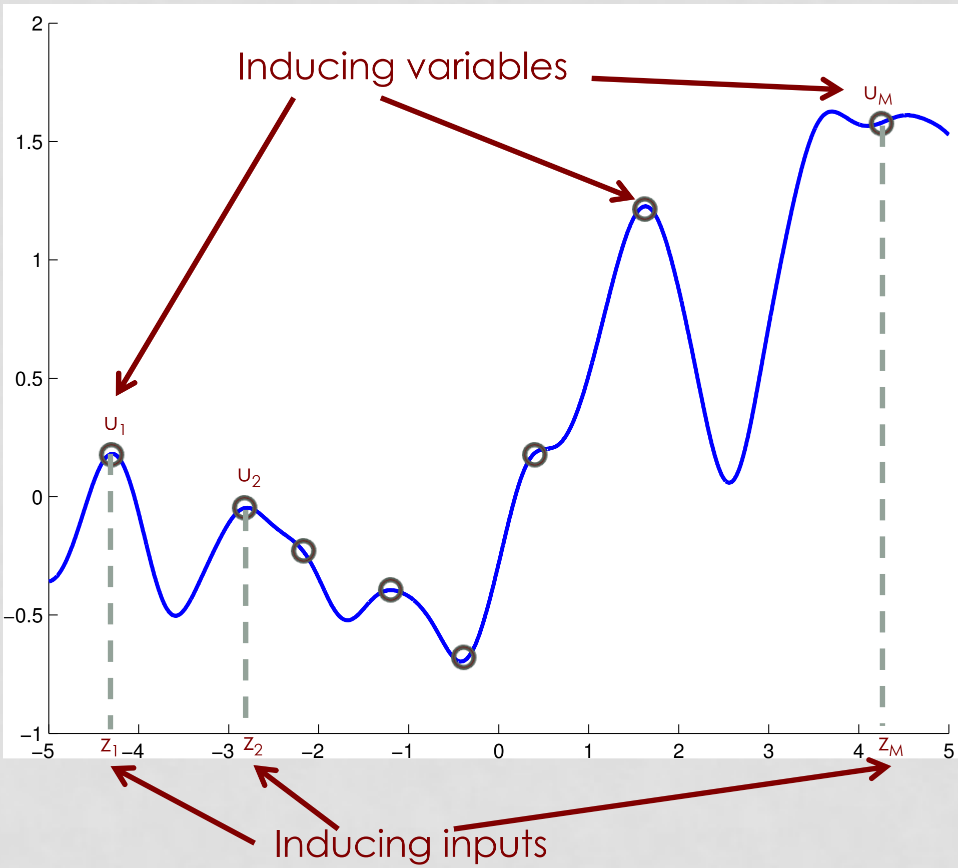


- Simplest approach: Throw data away
  - Exact GP on M < N data-points → O(M^3)
  - Can be selected at random or more smartly
    - E.g. Lawrence et al (NIPS, 2003)
  - Very hard to get a good picture of uncertainties
- Iterative solution of linear systems
  - Exact when run for N iterations
  - Approximate when run for I < N iterations → O(IN^2)
- ML approach: Approximate/decompose $\tilde{\mathbf{K}}_{\mathbf{f},\mathbf{f}}$
  - E.g. use M **inducing points**
    - Apply mathematical tricks (e.g. Woodbury's formula)
    - Computation usually O(M^2N)
    - This uses all the data

Inducing Variables & Unifying Framework



What Are the Inducing Points?



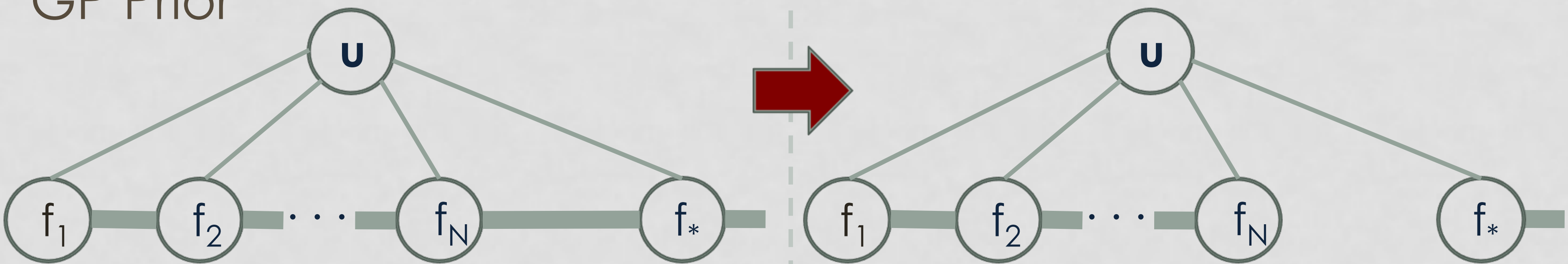
- Inducing variables $\mathbf{u}$
  - Latent values of the GP (as $\mathbf{f}$ and $\mathbf{f}_*$)
  - Usually marginalized
- Inducing inputs $\mathbf{z}$
  - Corresponding input locations (as $\mathbf{x}$)
  - Imprint on final solution

- Generalization of "support points", "active set", "pseudo-inputs"
  - 'Good' summary statistics → *induce* statistical dependencies
  - Can be a subset of the training set
  - Can be arbitrary inducing variables

A Unifying Framework for GP Approximations

(Quiñonero-Candela & Rassmussen, 2005)

GP Prior



Full GP (no approximations). All latent functions are fully connected

Training and test values are **conditionally independent** given **u**

- The joint prior is modified through the inducing variables:

$$p(\mathbf{f}_*, \mathbf{f}) \approx q(\mathbf{f}_*, \mathbf{f}) \stackrel{\text{def}}{=} \int q(\mathbf{f}_*|\mathbf{u}) q(\mathbf{f}|\mathbf{u}) p(\mathbf{u}) \; \mathrm{d}\mathbf{u}$$

Test conditional Training conditional Exact from GP prior with $\mathbf{K_{uu}}$

- Most (previously proposed) approx. methods:
  - Different specifications of these conditionals
  - Different **Z**: Subset of training/test inputs, new **z** inputs

SoR: Subset of Regressors

(Silverman, 1985; Wahba, 1999; Smola & Bartlett, 2001)

The mean predictor can be obtained with:

$$f(\mathbf{x}_*) = \sum_{i=1}^{N} \alpha_i \kappa(\mathbf{x}_*, \mathbf{x}_i) \qquad \boldsymbol{\alpha} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\right)$$

- SoR truncates the number of regressors needed:

$$f_{\mathrm{SoR}}(\mathbf{x}_*) = \mathbf{k}_*^T \boldsymbol{\alpha}_u \quad \boldsymbol{\alpha}_u \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}) \rightarrow \mathbf{u} = \mathbf{K}_{\mathbf{u},\mathbf{u}} \boldsymbol{\alpha}_u$$

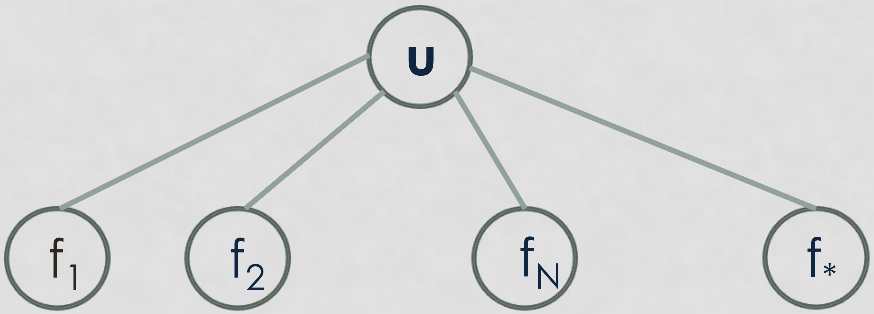
Deterministic relation

- Training conditional: $q_{\mathrm{SoR}}(\mathbf{f}|\mathbf{u}) = \mathcal{N}\left(\mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \mathbf{0}\right)$
  - Similar for the test conditional
- Prediction complexity: $O(M^2N)$
- Projected Processes (Csató & Opper, 2002; Seeger et al, 2003)
  - Similar to SoR but it uses the ‘exact’ test conditional
    - Usually better predictive variances than SoR
    - Not really a GP!

FITC, PITC, BCM

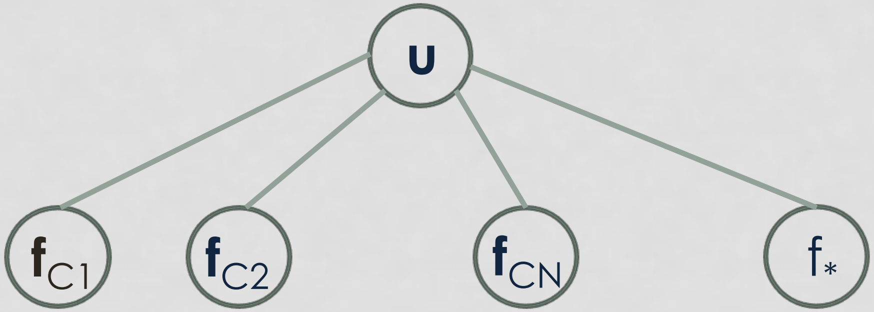
(Snelson & Ghahramanai, 2006; Quiñonero-Candela & Rassmussen, 2005; Tresp, 2000)

FITC: Fully independent training conditionals



Diagonal ('true') covariance for training conditionals

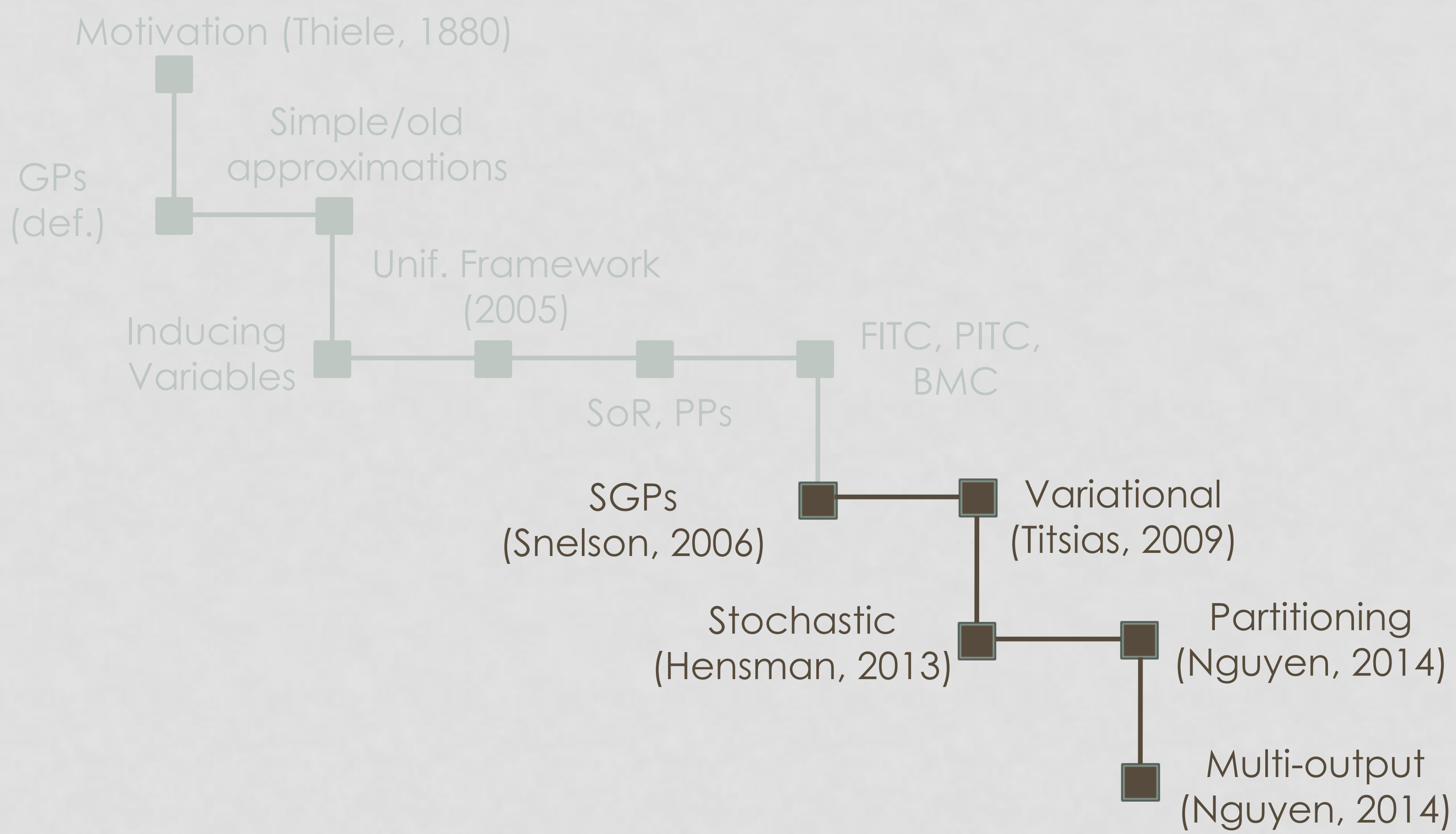
PITC: Partially independent training conditionals



Block diagonal covariance for training conditionals

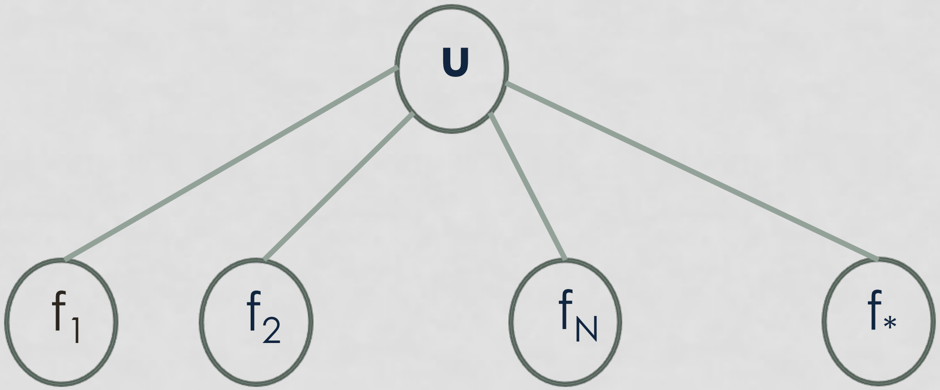
- **BCM**: Bayesian Committee Machine
  - Same as PITC but selection of inducing variables depends on test points
    - Transductive setting
    - Transduction cannot really occur in exact GPs
- Same cost as SoR

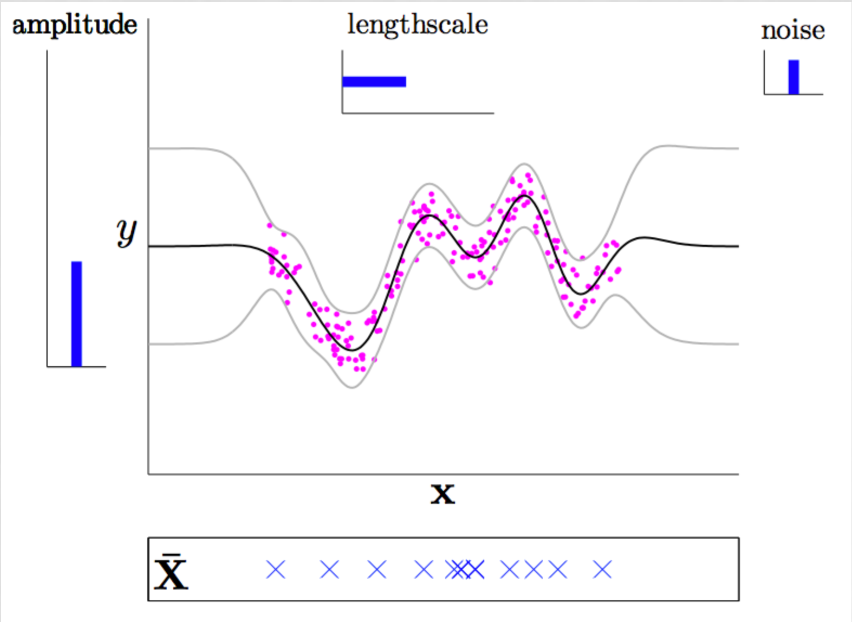
LEARNING THE INDUCING POINTS



SGP: Sparse GPs

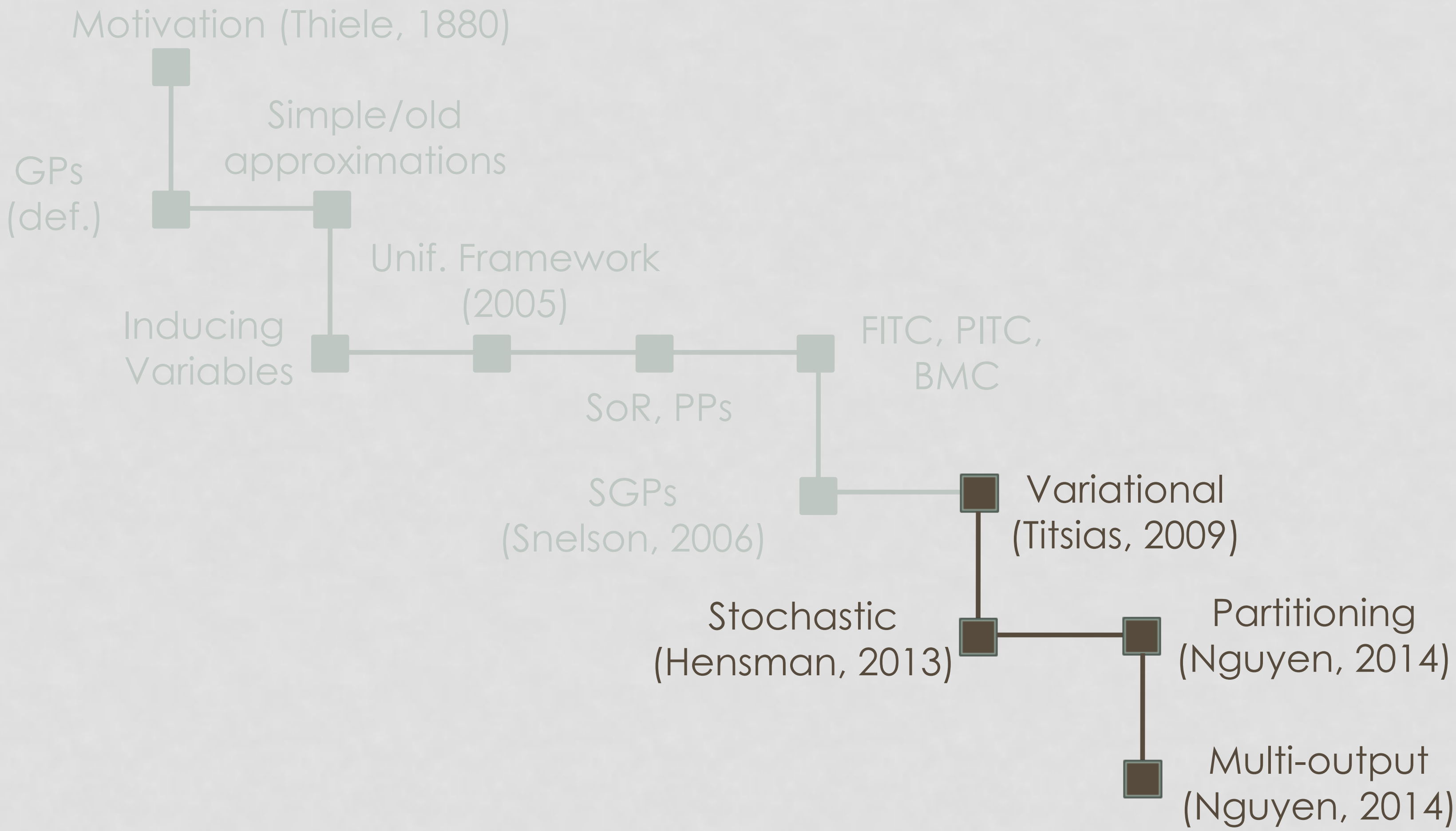
(Snelson & Ghahramani, 2006)





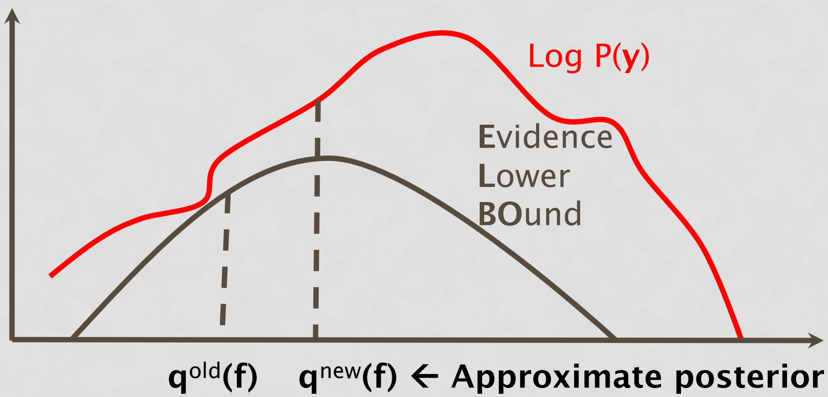
- FITC model but inducing points do not belong to training or test test
  - Instead they are 'free' parameters of the model
  - This facilitates continuous optimization (cf. selecting a subset)
  - Both the locations of the inducing inputs and the GP hyper-parameters are learned by optimization of the approximate marginal likelihood

VARIATIONAL STUFF



VFE: Variational Free Energy Optimization

(Titsias, 2009)

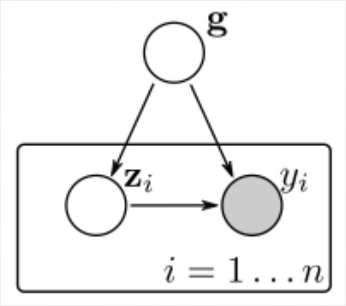


- Inducing-point model
  - Do not modify the (prior) model
  - Approximate posterior over inducing variables

- ELBO: Single consistent objective function
  - Inducing variables are 'marginalized' variationally
  - *Inducing inputs are additional variational parameters*
  - Joint learning of posterior and variational parameters
  - Additional regularization term appears naturally
- Predictive distribution
  - Equivalent to PP
  - $O(M^2N)$ → Good enough?

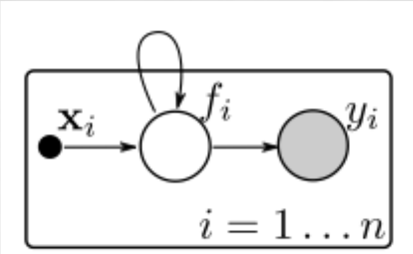
SVI-GP: Stochastic Variational Inference
(Hensman et al, 2013)

SVI for 'big data'



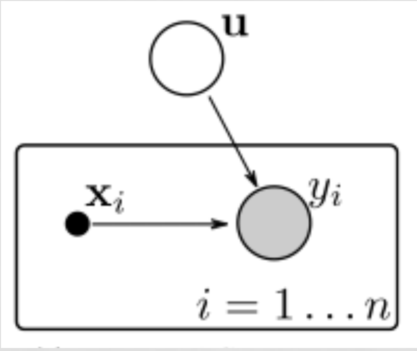
Decomposition across data-points through global variables

GPs



Fully coupled by definition

Large scale GPs



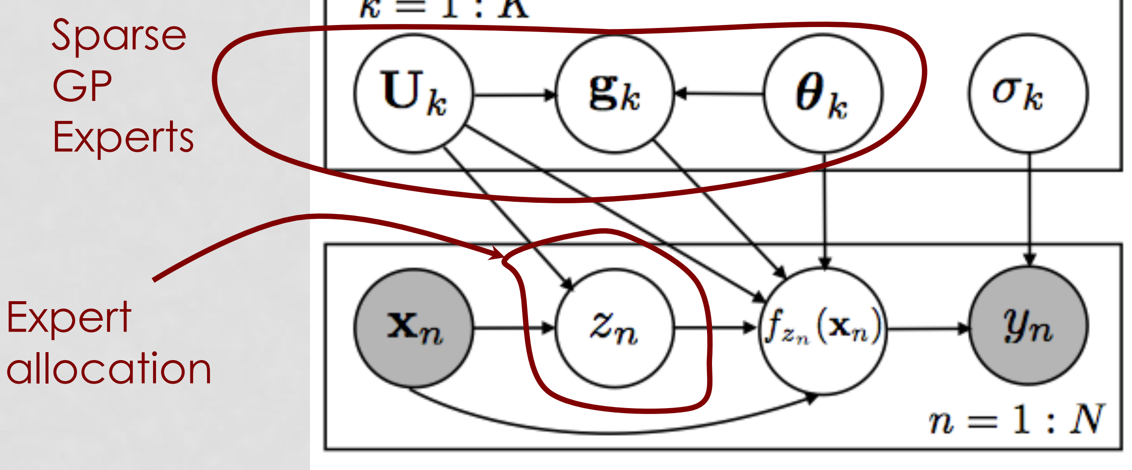
Inducing variables can be such global variables

- Maintain an explicit representation of inducing variables in lower bound (cf. Titsias)
  - Lower bound decomposes across inputs
  - Use stochastic optimization
  - Cost $O(M^3)$ in time → Can scale to very large datasets!

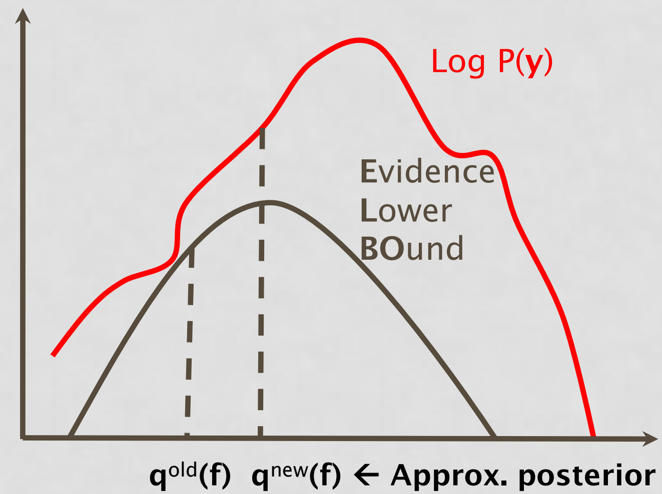
F~~A~~GP: FAST ALLOCATION OF GPS

(NGUYEN & BONILLA, 2014)

Mixture of GPs



Variational inference

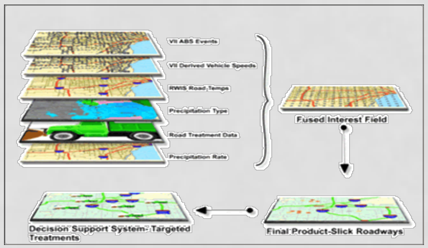


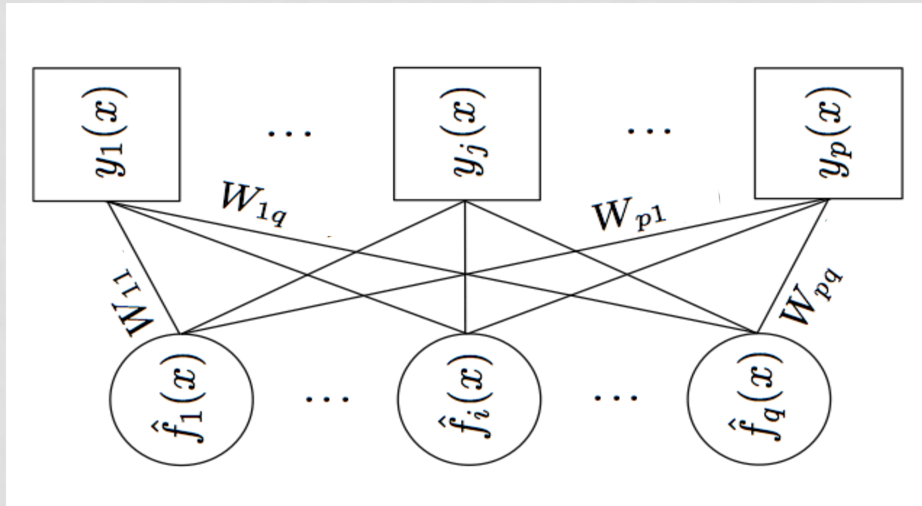
- A single GP for big data is undesirable (why?)
- Mixture of (local) sparse GP experts
  - Allocation is a function of inducing variables
  - Variational inference (learn everything)
  - Non-stationarity for 'free'
  - Cost $O(NM_k^2)$ → Can afford many more inducing points!

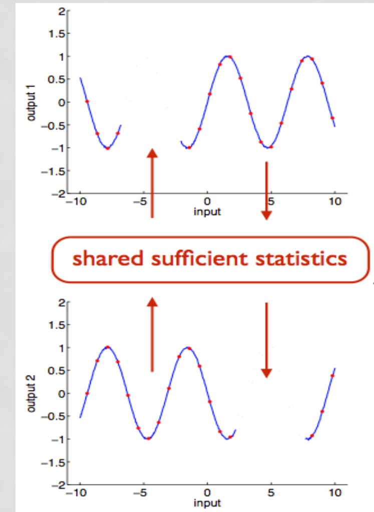
COGP: Collaborative Multi-output GPs

(Nguyen & Bonilla, 2014)

Data fusion / multi-task learning







- True 'big data' GP
  - Learning from multiple sources
  - Mixture of Sparse latent GPs
  - Sharing of additional inducing points
- Variational inference: $O(M_i^3)$
  - Scalable to a large number of inputs and outputs
  - Affords much larger # of inducing inputs

SUMMARY / ACKNOWLEDGEMENTS

Motivation (Thiele, 1880)

GPs (def.)

Simple/old approximations

Inducing Variables

Unif. Framework (2005)

SoR, PPs

FITC, PITC, BMC

SGPs (Snelson, 2006)

Variational (Titsias, 2009)

Stochastic (Hensman, 2013)

Partitioning (Nguyen, 2014)

Multi-output (Nguyen, 2014)