Learning flexible models of nonlinear dynamical systems

"Inspired by the Gaussian process and enabled by the particle filter"

UPPSALA
UNIVERSITET

Thomas Schön
Division of Systems and Control
Department of Information Technology
Uppsala University.

Email: thomas.schon@it.uu.se,
www: `user.it.uu.se/~thosc112`

Joint work with: **Roger Frigola** (indep. consultant), **Carl Jidling** (UU), **Michael Jordan** (UC Berkeley), **Fredrik Lindsten** (UU), **Carl Rasmussen** (University of Cambridge), **Arno Solin** (Aalto university), **Andreas Svensson** (UU), **Petre Stoica** (UU), **Simo Särkkä** (Aalto university), **Niklas Wahlström** (UU), **Adrian Wills** (University of Newcastle), **Johan Wågberg** (UU) and **Dave Zachariah** (UU).

UPPSALA UNIVERSITET

A probabilistic approach to modelling

Data on its own is typically useless, it is only when we can extract knowledge from the data that it becomes useful.

Representation of the data: A **model with unknown** (a.k.a. latent or missing) **variables** related to the knowledge we are looking for.

Key concept: **Uncertainty**.

Key ingredient: **Data**.

Probability theory and statistics provide the theory and practice that is needed for representing and manipulating uncertainty about data, models and predictions.

Machine learning gives computers the ability to **learn without being explicitly programmed** for the task at hand.

UPPSALA UNIVERSITET

The three cornerstones

Cornerstone 1 **(Data)** Typically we need lots of it.

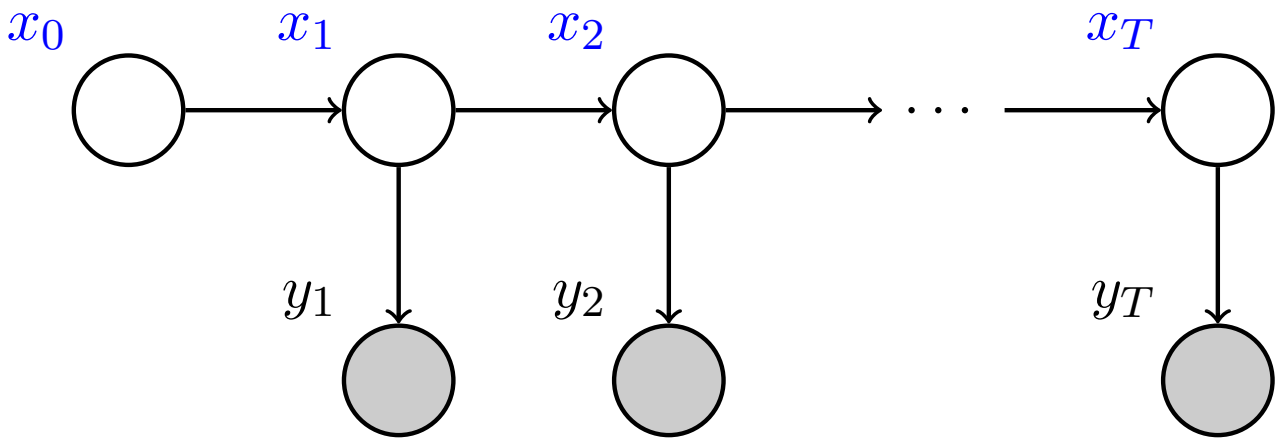
Cornerstone 2 **(Mathematical model)** A mathematical model is a compact representation of the data that in precise mathematical form captures the key properties of the underlying situation.

Cornerstone 3 **(Learning algorithm)** Used to compute the unknown variables from the observed data using the model.

What is a dynamical system?

Something evolving over time with a memory.



UPPSALA UNIVERSITET

Probabilistic modeling of dynamical systems

Probabilistic modeling allow for **representing and manipulating uncertainty** in data, models, decisions and predictions.

A **parametric** state space model is given by:

$$\begin{aligned} x_{t+1} &= f_\theta(x_t, u_t) + v_{\theta,t}, \\ y_t &= g_\theta(x_t, u_t) + e_{\theta,t}, \\ x_1 &\sim p_\theta(x_1), \\ \theta &\sim p(\theta). \end{aligned} \qquad \begin{aligned} x_{t+1} \,|\, x_t &\sim p_\theta(x_{t+1} \,|\, x_t, u_t), \\ y_t \,|\, x_t &\sim p_\theta(y_t \,|\, x_t, u_t), \\ x_1 &\sim p_\theta(x_1), \\ \theta &\sim p(\theta). \end{aligned}$$

The **full probabilistic model** is given by

$$p(x_{1:T}, \theta, y_{1:T}) = \underbrace{p(y_{1:T} \,|\, x_{1:T}, \theta)}_{\text{data distribution}} \underbrace{p(x_{1:T}, \theta)}_{\text{prior}}$$

UPPSALA UNIVERSITET

Probabilistic modeling of dynamical systems

Distribution describing a parametric nonlinear state space model:

$$p(x_{1:T}, \theta, y_{1:T}) = \underbrace{\prod_{t=1}^{T} \underbrace{p(y_t \mid x_t, \theta)}_{\text{observation}}}_{\text{data distribution}} \underbrace{\prod_{t=1}^{T-1} \underbrace{p(x_{t+1} \mid x_t, \theta)}_{\text{dynamics}} \underbrace{p(x_1 \mid \theta)}_{\text{state}} \underbrace{p(\theta)}_{\text{param.}}}_{\text{prior}}$$

Model = probability distribution!

Aim: Construct a flexible model and compute its posterior distribution

$$p(x_{1:T}, \theta \mid y_{1:T}) = \underbrace{p(x_{1:T} \mid \theta, y_{1:T})}_{\text{state}} \underbrace{p(\theta \mid y_{1:T})}_{\text{parameter}}.$$

Example – "what are x_t, θ and y_t"?

Aim (motion capture): Compute x_t (position and orientation of the different body segments) of a person (θ describes the body shape) moving around indoors using measurements y_t (accelerometers, gyroscopes and ultrawideband).

Manon Kok, Jeroen D. Hol and Thomas B. Schön. **Indoor positioning using ultrawideband and inertial measurements**. *IEEE Transactions on Vehicular Technology*, 64(4):1293-1303, April, 2015.

Manon Kok, Jeroen D. Hol and Thomas B. Schön. **Using inertial sensors for position and orientation estimation** Pre-print, *arXiv:1704.06053*, April, 2017.

UPPSALA UNIVERSITET

Use flexible models

Key lesson from modern Machine Learning:

Flexible models often gives the best performance.

How can we build flexible models?

1. Models that use a large (but fixed) number of parameters. (**parametric**, ex. deep learning)

   LeCun, Y., Bengio, Y., and Hinton, G. **Deep learning**, *Nature*, Vol 521, 436–444, 2015.

2. Models that use more parameters as we get more data. (**non-parametric**, ex. Gaussian process)

   Ghahramani, Z. **Bayesian nonparametrics and the probabilistic approach to modeling**. *Phil. Trans. R. Soc. A* 371, 2013.

   Ghahramani, Z. **Probabilistic machine learning and artificial intelligence**. *Nature* 521:452-459, 2015.

UPPSALA UNIVERSITET

Outline

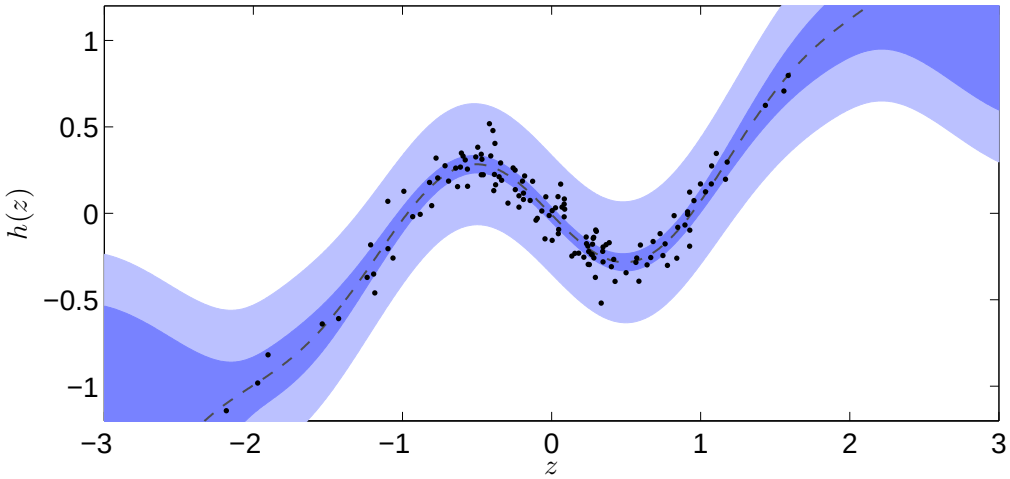
1. Probabilistic modeling of dynamical systems
2. **Gaussian Process (GP) introduction**
3. GP state space model
   a) Model construction
   b) Sequential Monte Carlo (SMC)
   (c) Learning using SMC within Gibbs)
4. Examples
5. Some related ongoing research
6. Conclusions

Probabilistic modeling allows us to systematically **represent** and **manipulate** uncertainty.

The Gaussian process model

The Gaussian process (GP) is a **non-parametric** and **probabilistic** model for nonlinear functions.

- **Non-parametric** means that it does not rely on any particular parametric functional form to be postulated.
- **Probabilistic** means that it takes uncertainty into account in every aspect of the model.



Fredrik Lindsten, Thomas B. Schön and Michael I. Jordan. **Bayesian semiparametric Wiener system identification**. *Automatica*, 49(7): 2053-2063, July 2013.

An abstract idea

In probabilistic linear regression

$$y_i = \underbrace{\beta^{\mathsf{T}} x_i}_{f(x)} + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2),$$

we place a prior on β, $\beta \sim \mathcal{N}(0, \sigma^2 I_p)$.

(Abstract) idea: What if we instead place a prior directly on the function $f(\cdot)$

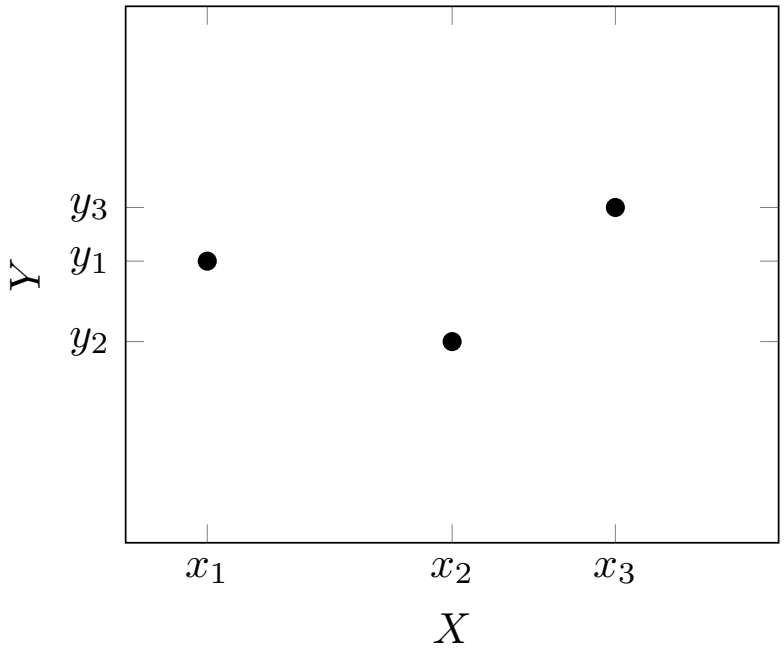
$$f \sim p(f)$$

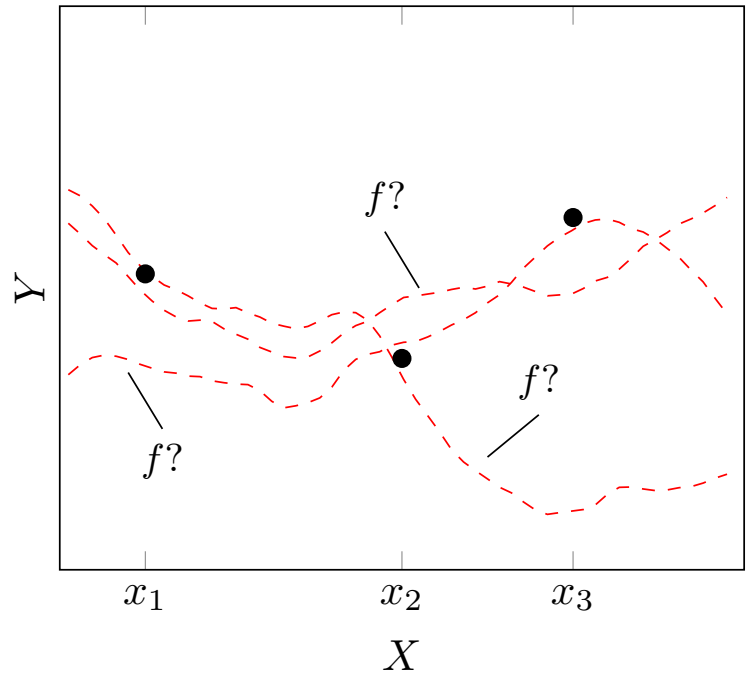
and look for $p(f \,|\, \mathbf{y})$ rather than $p(\beta \,|\, \mathbf{y})$?!

UPPSALA UNIVERSITET

An abstract idea – pictures

What does it actually mean to have a prior over functions?





Can we construct a probabilistic object operating on functions?

One concrete construction

Well, one (arguably simple) idea on how we can reason probabilistically about an unknown function f is by assuming that $f(x)$ and $f(x')$ are jointly Gaussian distributed

$$\begin{pmatrix} f(x) \\ f(x') \end{pmatrix} \sim \mathcal{N}(\mu, K)$$

If we accept the above idea we can without conceptual problems generalize to any *arbitrary* set of input values $\{x_1, x_2, \ldots, x_N\}$.

Definition and its implications

Definition: (Gaussian Process, GP) A GP is a (potentially infinite) collection of random variables such that any finite subset of it is jointly distributed according to a Gaussian.

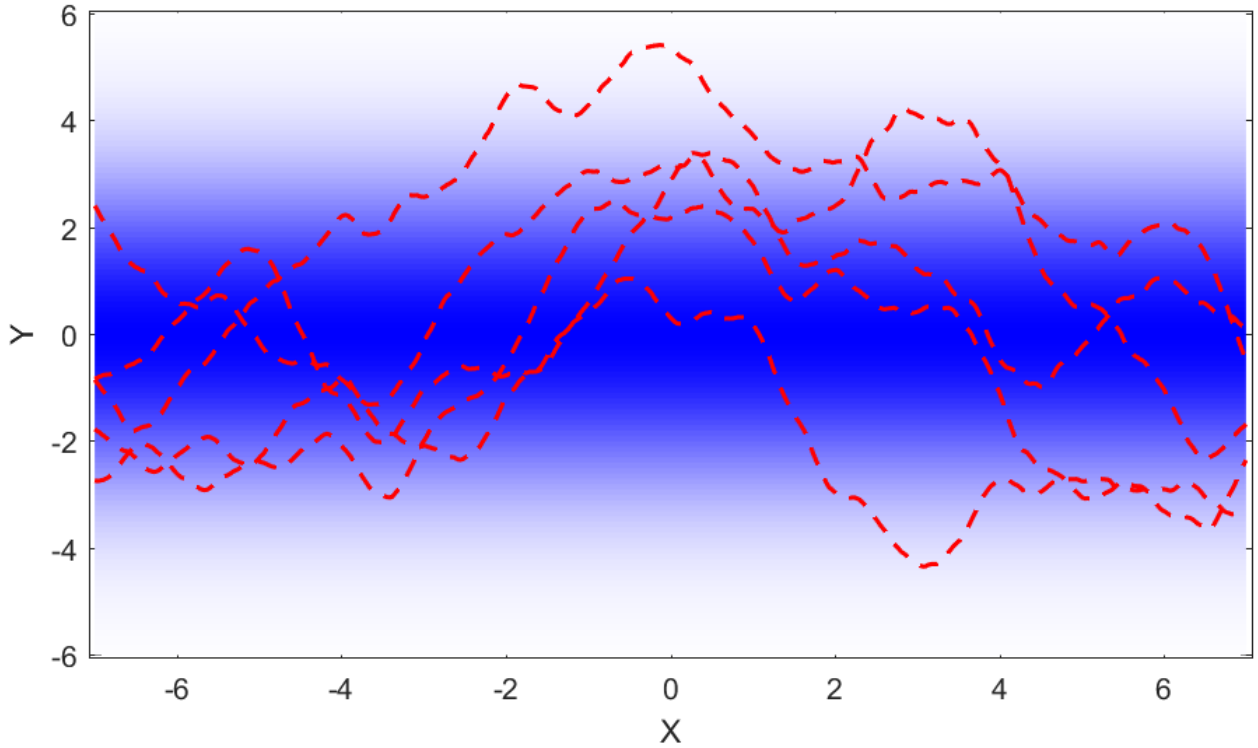
Our definition means that for any *arbitrary* set of input values $\{x_1, x_2, \ldots, x_N\}$ we have

$$\begin{pmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(x_1) \\ \vdots \\ m(x_N) \end{pmatrix}, \begin{pmatrix} k(x_1, x_1) & \ldots & k(x_1, x_N) \\ \vdots & \ddots & \vdots \\ k(x_N, x_1) & \ldots & k(x_N, x_N) \end{pmatrix} \right)$$

We now have a prior!

$$f \sim \mathcal{GP}(m, k)$$

The GP is a **generative** model so let us first sample from the prior.



GP regression

Remaining problem: Given training data $\mathcal{T} = \{x_i, y_i\}_{i=1}^N$ and our GP prior $f \sim \mathcal{GP}(m, k)$ compute $p(f_\star \,|\, \mathbf{y})$ for an arbitrary test point $(x_\star, y_\star)$.

$$\begin{pmatrix} \mathbf{y} \\ f_\star \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(\mathbf{x}) \\ m(x_\star) \end{pmatrix}, \begin{pmatrix} k(\mathbf{x},\mathbf{x}) + \sigma^2 I_N & k(\mathbf{x}, x_\star) \\ k(x_\star, \mathbf{x}) & k(x_\star, x_\star) \end{pmatrix} \right),$$

The conditioning theorem for partitioned Gaussians results in

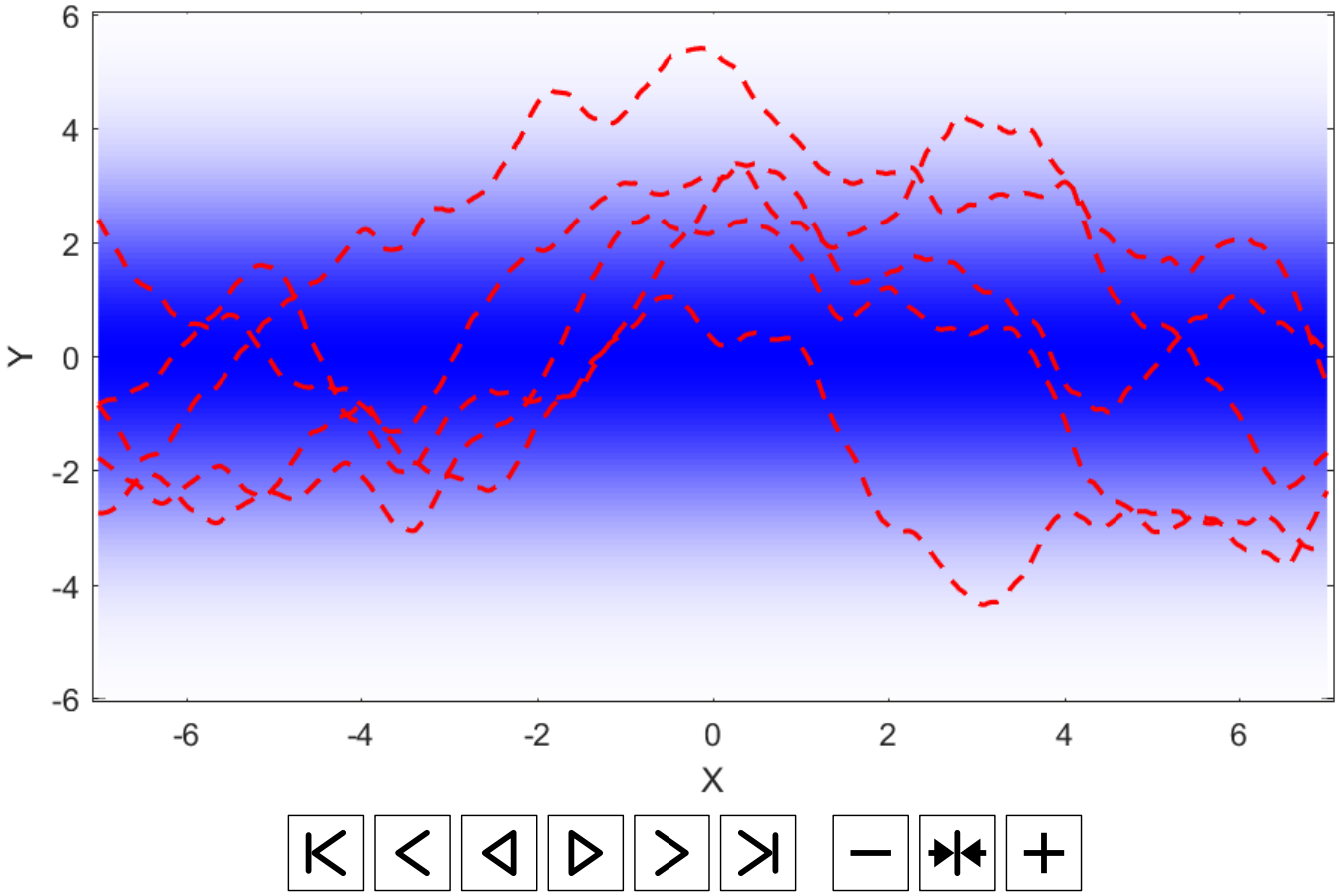
$$\begin{aligned} f_\star \,|\, \mathbf{y} &\sim \mathcal{N}\left(\mu_\star, k_\star\right), \\ \mu_\star &= m(x_\star) + \mathbf{s}^{\mathsf{T}}(\mathbf{y} - m(\mathbf{x})), \\ k_\star &= k(x_\star, x_\star) - \mathbf{s}^{\mathsf{T}} k(\mathbf{x}, x_\star), \end{aligned}$$

where

$$\mathbf{s}^{\mathsf{T}} = k(x_\star, \mathbf{x})(k(\mathbf{x},\mathbf{x}) + \sigma^2 I_N)^{-1}.$$

UPPSALA UNIVERSITET

GP regression – illustration



UPPSALA UNIVERSITET

Outline

1. Probabilistic modeling of dynamical systems
2. Gaussian Process (GP) introduction
3. **GP state space model**
   a) Model construction
   b) Sequential Monte Carlo (SMC)
   (c) Learning using SMC within Gibbs)
4. Examples
5. Some related ongoing research
6. Conclusions

Probabilistic modeling allows us to systematically **represent** and **manipulate** uncertainty.

Gaussian process state space model

Flexible models often gives the best performance.

$$x_{t+1} = f(x_t) + v_t, \qquad \text{s.t.} \quad f(x) \sim \mathcal{GP}(0, \kappa_{\eta,f}(x, x')),$$
$$y_t = g(x_t) + e_t, \qquad \text{s.t.} \quad g(x) \sim \mathcal{GP}(0, \kappa_{\eta,g}(x, x')).$$

The model functions f and g are assumed to be realizations from Gaussian process priors and $v_t \sim \mathcal{N}(0, Q)$, $e_t \sim \mathcal{N}(0, R)$.

We can now find the posterior distribution

$$p(f, g, Q, R, \eta \,|\, y_{1:T}),$$

via some approximation (VI or **particle MCMC**).

Frigola, Roger, Fredrik Lindsten, Thomas B. Schön, and Carl Rasmussen. **Bayesian inference and learning in Gaussian process state-space models with particle MCMC**. In *NIPS*, 2013.

Frigola, Roger. **Bayesian time series learning with Gaussian processes** PhD thesis, University of Cambridge, 2015.

Andreas Svensson and Thomas B. Schön. **A flexible state space model for learning nonlinear dynamical systems**, *Automatica*, 80:189-199, June, 2017.

Approximate Gaussian processes

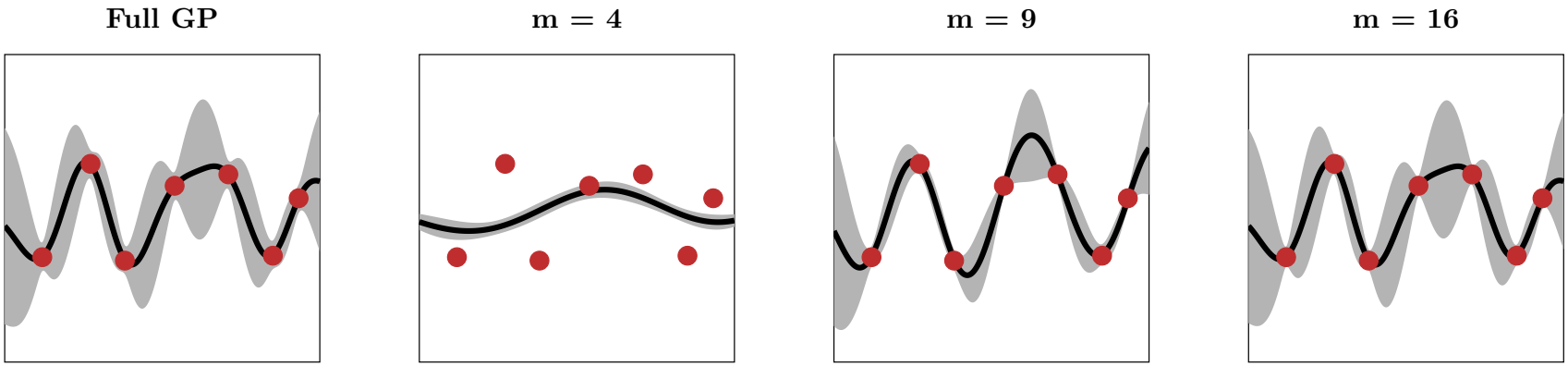
We use a "reduced-rank" GP approximation:

$$f \sim \mathcal{GP}(0, k) \quad \Leftrightarrow \quad f(x) \approx \sum_{j=0}^{m} w^j \phi^j(x)$$

with prior

$$w^j \sim \mathcal{N}(0, S(\lambda^j))$$

For $x \in [-L, L] \subset \mathbb{R}$: $\quad \phi^j(x) = \frac{1}{\sqrt{L}} \sin\left(\frac{\pi j (x+L)}{2L}\right).$



Arno Solin and Simo Särkkä. **Hilbert Space Methods for Reduced-Rank Gaussian Process Regression.** *arXiv:1401.5508*, 2014.

Computationally feasible GP-SSM

Original formulation:

$$
\begin{aligned}
x_{t+1} &= f(x_t) + v_t, & v_t &\sim \mathcal{N}(0, Q), \\
y_t &= g(x_t) + e_t, & e_t &\sim \mathcal{N}(0, R), \\
f(x) &\sim \mathcal{GP}(0, \kappa_{\eta,f}(x, x')) &&
\end{aligned}
$$

Formulation using the reduced-rank GP approximation:

$$
\begin{aligned}
x_{t+1} &= \sum_{j=0}^{m} w^j \phi^j(x_t) + v_t, & v_t &\sim \mathcal{N}(0, Q), \\
y_t &= g(x_t) + e_t, & e_t &\sim \mathcal{N}(0, R), \\
w^j &\sim \mathcal{N}(0, S(\lambda^j)). &&
\end{aligned}
$$

Linear in the parameters w^i and nonlinear in the states x_t.

UPPSALA UNIVERSITET

The learning problem (dynamical systems)

Compute the posterior distribution

$$p(x_{1:T}, \theta \,|\, y_{1:T}) = \underbrace{p(x_{1:T} \,|\, \theta, y_{1:T})}_{\text{state}} \underbrace{p(\theta \,|\, y_{1:T})}_{\text{parameter}}.$$

HD integration/optimization problems without analytical solution.

Sequential Monte Carlo provide approximations to integration problems where there is a **sequential structure** present.

Learning the parameters θ is rather straightforward in the GP-SSM.

The states $x_{1:T}$ are still challenging.

Sequential Monte Carlo (SMC)

The distribution of interest $\gamma(x)$ is called **target distribution**.

(Abstract) problem formulation: **Sample from a sequence** of probability distributions $\{\gamma_t(x_{1:t})\}_{t\geq 1}$ defined on a sequence of spaces of increasing dimension, where

$$\gamma_t(x_{1:t}) = \frac{\bar{\gamma}_t(x_{1:t})}{Z_t},$$

such that $\bar{\gamma}_t(x_t) : \mathsf{X}^t \to \mathbb{R}^+$ is known pointwise and $Z_t = \int \gamma(x_{1:t})\mathrm{d}x_{1:t}$ is often computationally challenging.

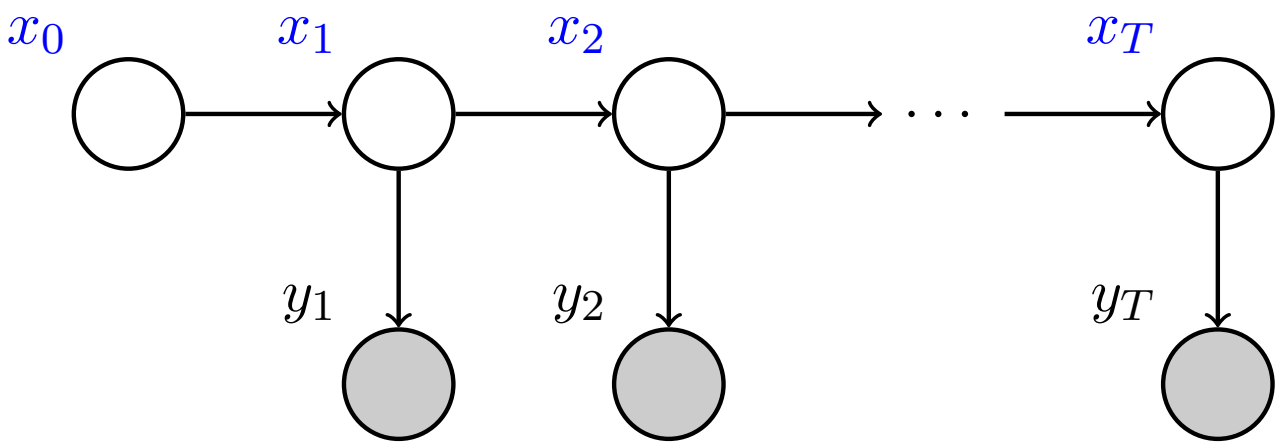
1. Approximate the normalizing constant Z_t.
2. Approximate $\gamma_t(x_t)$ and compute integrals $\int \varphi(x_t)\gamma_t(x_t)\mathrm{d}x_t$.

Important question: How general is this formulation?

Sequential Monte Carlo (SMC)

The sequence of target distributions $\{\gamma_t(x_{1:t})\}_{t=1}^n$ can be constructed in **many** different ways.

The most basic construction arises from **chain-structured graphs**, such as the state space model (SSM).



$$\overbrace{p(x_{1:t} \mid y_{1:t})}^{\gamma_t(x_{1:t})} = \frac{\overbrace{p(x_{1:t}, y_{1:t})}^{\bar{\gamma}_t(x_{1:t})}}{\underbrace{p(y_{1:t})}_{Z_t}}$$

Sequential Monte Carlo (SMC)

The **particle filter** approximates $p(x_{1:t} \mid y_{1:t})$ for

$$\begin{aligned} x_{t+1} &= f_\theta(x_t, u_t) + v_{\theta,t}, \\ y_t &= g_\theta(x_t, u_t) + e_{\theta,t}, \end{aligned}$$

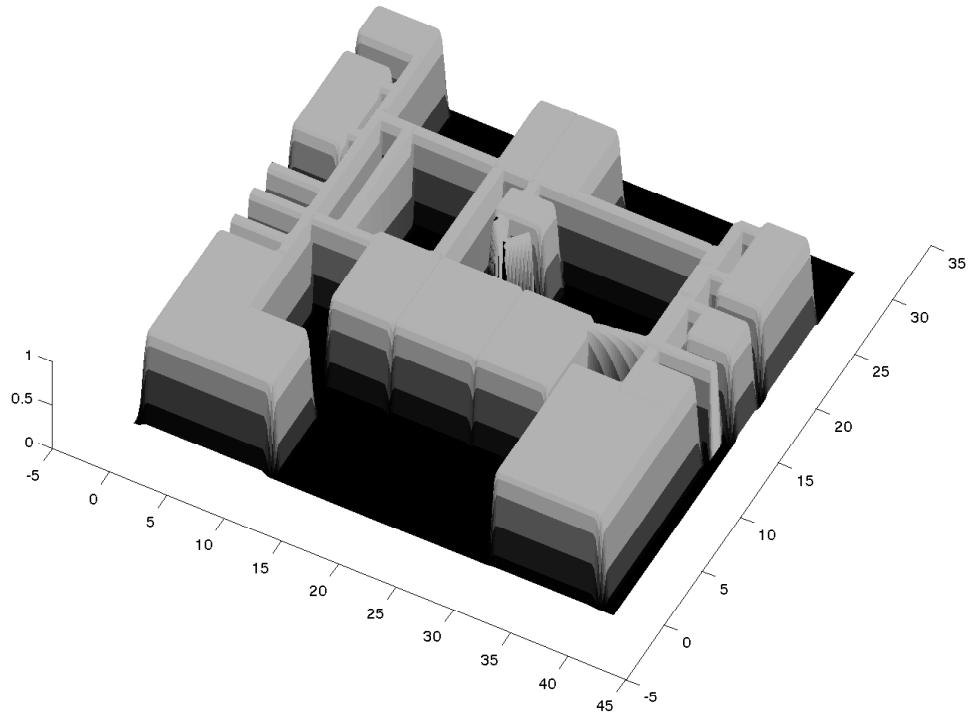
by maintaining an empirical distribution made up of N samples (particles) $\{x_{1:t}^i\}_{i=1}^N$ and corresponding weights $\{w_{1:t}^i\}_{i=1}^N$

$$\underbrace{\widehat{p}(x_{1:t} \mid y_{1:t})}_{\widehat{\gamma}(x_{1:t})} = \sum_{i=1}^N \frac{w_t^i}{\sum_{j=1}^N w_t^j} \delta_{x_{1:t}^i}(x_{1:t}).$$

"The particle filter provides a systematic way of exploring the state space."

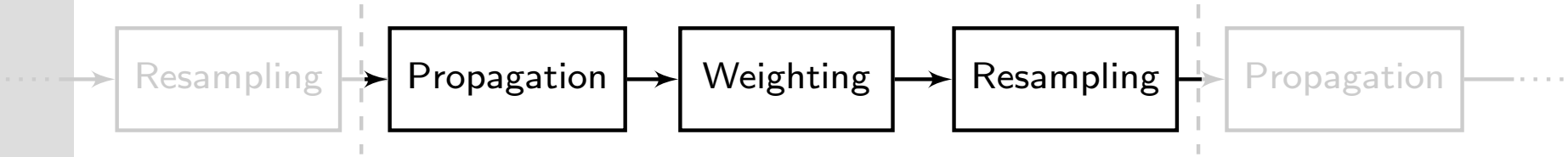
Example – indoor localization

Aim: Compute the position of a person moving around indoors using sensors (inertial, magnetometer and radio) located in an ID badge and a map, i.e. compute $p(x_t \mid y_{1:t})$.



Show movie

Sequential Monte Carlo – particle filter



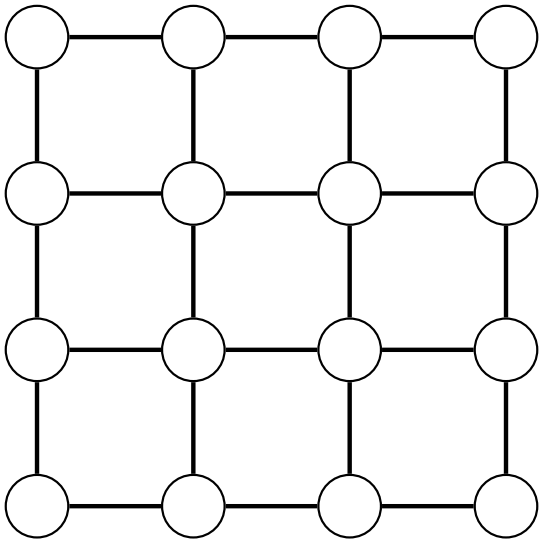
SMC = sequential importance sampling + resampling

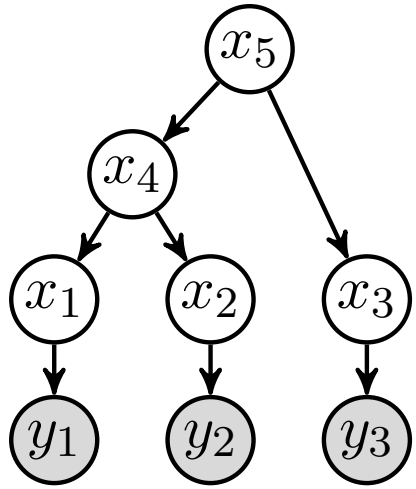
1. **Propagation:** $x_t^i \sim f(x_t \mid x_{1:t-1}^{a_t^i})$ and $x_{1:t}^i = \{x_{1:t-1}^{a_t^i}, x_t^i\}$.
2. **Weighting:** $\bar{w}_t^i = W_t(x_t^i) = g(y_t \mid x_t^i)$.
3. **Resampling:** $\mathbb{P}\left(a_t^i = j\right) = \bar{w}_{t-1}^j / \sum_l \bar{w}_{t-1}^l$.

The **ancestor indices** $\{a_t^i\}_{i=1}^N$ are very **useful** auxiliary variables! They make the stochasticity of the resampling step explicit.

The nonlinear SSM is just a special case...

Constructing an artificial sequence of intermediate target distributions for an SMC sampler is a powerful (and **quite possibly underutilized**) idea.





Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Sequential Monte Carlo methods for graphical models**. *Advances in Neural Information Processing Systems (NIPS)* 27, Montreal, Canada, December, 2014.

Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Nested sequential Monte Carlo**. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, Lille, France, July, 2015.

UPPSALA UNIVERSITET

Particle MCMC = SMC + MCMC

A systematic way of combining SMC and MCMC.

Builds on an extended target construction.

Intuitively: SMC is used as a high-dimensional proposal mechanism on the space of state trajectories X^T.

A bit more precise: Construct a Markov chain with $p(\theta, x_{1:T} \mid y_{1:T})$ (or one of its marginals) as its stationary distribution. Also used for parameter learning.

Exact approximations

Pioneered by the work

Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**, *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

UPPSALA UNIVERSITET

Resulting strategy/algorithm

Strategy Bayesian learning of the GP-SSM

1. **Initialize:** Set $\theta[0]$.
2. **for** $k = 1$ **to** K **do:**
   (a) Sample $x_{1:T}[k]$ using CPF-AS.
   (b) Sample $\theta[k]$ using closed form conjugate relationships.
3. **end for**

For all the details see

Andreas Svensson and Thomas B. Schön. **A flexible state space model for learning nonlinear dynamical systems**, *Automatica*, 80:189-199, June, 2017.

Alternative approach using VI

Roger Frigola, Yutian Chen, and Carl E. Rasmussen. **Variational Gaussian process state-space models**. In *Advances in Neural Information Processing Systems 27 (NIPS)*, 2014.

Toy example

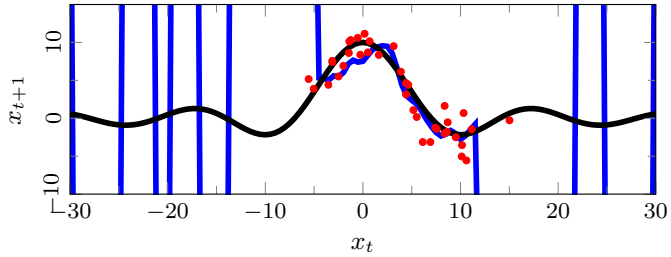
$$x_{t+1} = 10\text{sinc}\left(\frac{x_t}{7}\right) + v_t \quad v_t \sim \mathcal{N}(0,4)$$

$$y_t = x_t + e_t \text{ (known)} \quad e_t \sim \mathcal{N}(0,4)$$

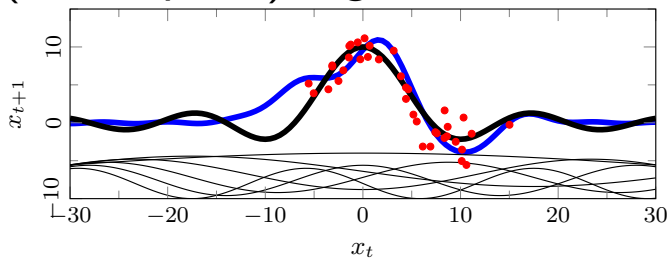
$$T = 40, m = 40$$

Posterior model uncertainty
Learned model
True state transition function
State samples underlying data
Basis functions

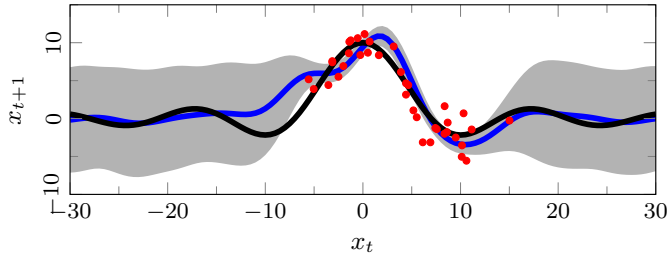
Maximum likelihood without regularization



Maximum likelihood with (GP-inspired) regularization



Bayesian learning: Full posterior



Narendra-Li Benchmark

$$
\begin{aligned}
x_{t+1}^1 &= \left(\frac{x_t^1}{1+(x_t^1)^2} + 1\right)\sin(x_t^2) \\
x_{t+1}^2 &= x_t^2\cos(x_t^2) + x_t^1\exp\left(-\frac{(x_t^1)^2+(x_t^2)^2}{8}\right) \\
&\quad + \frac{(u_t)^3}{1+(u_t)^2+0.5\cos(x_t^1+x_t^2)} \\
y_t &= \frac{x_t^1}{1+0.5\sin(x_t^2)} + \frac{x_t^2}{1+0.5\sin(x_t^1)} + e_t
\end{aligned}
$$

Method	RMSE	T
GP-SSM	**0.06**	2 000
Roll et al.	0.43	50 000
Stenman et al.	0.46	50 000
Xu et al. (AHH)	0.31	2 000
Xu et al. (MARS)	0.49	2 000

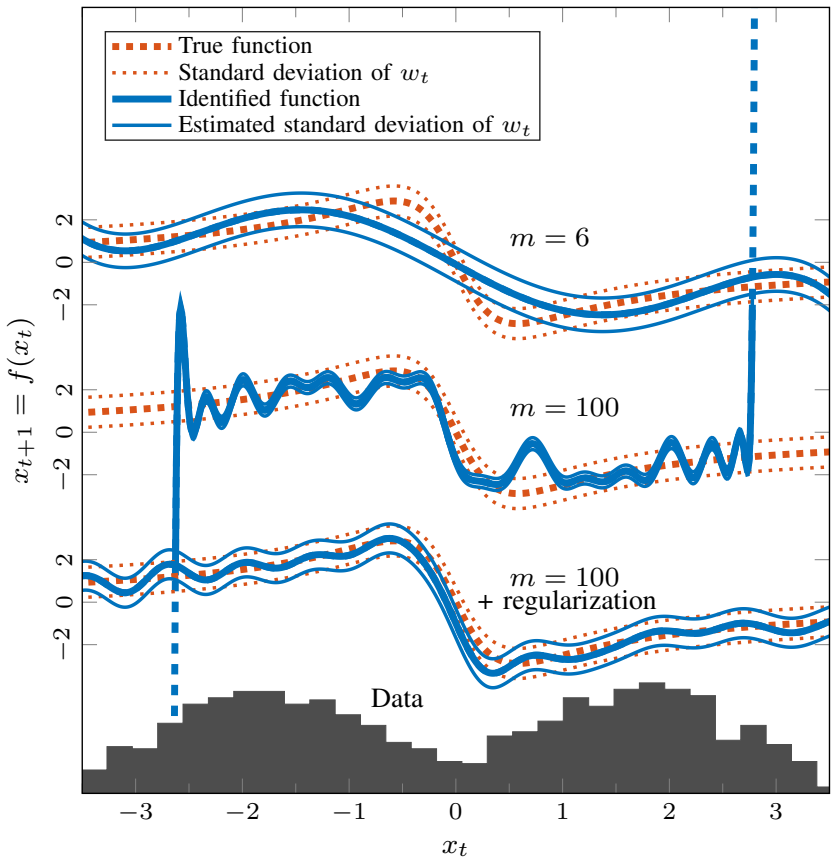
J. Roll, A. Nazin, and L. Ljung. **Nonlinear system identification via direct weight optimization**. *Automatica*, 41(3):475–490, 2005.

A. Stenman. **Model on demand: Algorithms, analysis and applications**. *PhD thesis, Linköping University*, 1999.

J. Xu, X. Huang, and S. Wang. **Adaptive hinging hyperplanes and its applications in dynamic system identification**. *Automatica*, 45(10):2325–2332, 2009.

Regularization in nonlinear state spaces

We can also solve a (regularized) maximum likelihood problem.



Results in a **flexible** non-parametric model where the GP prior on f takes on the **role of a regularizer**.

Provides a data-driven way of **tuning** the model flexibility.

Toy example:

$$x_{t+1} = -10\frac{x_t}{1+3x_t^2} + v_t,$$

$$y_t = x_t + e_t.$$

UPPSALA UNIVERSITET

Outline

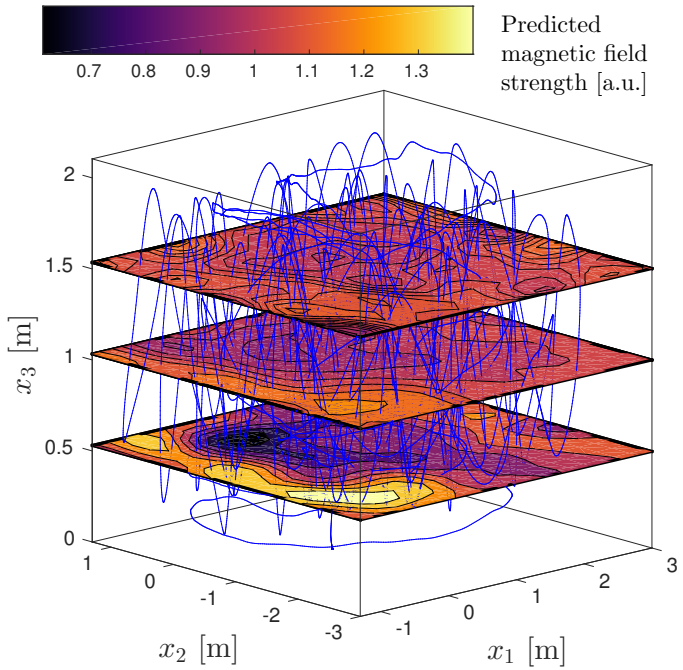
1. Probabilistic modeling of dynamical systems
2. Gaussian Process (GP) introduction
3. GP state space model
   a) Model construction
   b) Sequential Monte Carlo (SMC)
   (c) Learning using SMC within Gibbs)
4. Examples
5. **Some related ongoing research**
6. Conclusions

Probabilistic modeling allows us to systematically **represent** and **manipulate** uncertainty.

Linearly constrained GPs

Problem formulation: Modification of the covariance function in a GP to correctly account for known linear operator constraints.

1. By modelling the target function as a transformation of an underlying function the constraints are **explicitly incorporated** into the model.
2. Result:
   a) A probabilistic model that is **guaranteed** to fulfil the constraints.
   b) A constructive procedure for designing the transformation.



Carl Jidling, Niklas Wahlström, Adrian Wills and Thomas B. Schön. **Linearly constrained Gaussian processes**. Pre-print *arXiv:1703:00787*, March, 2017.

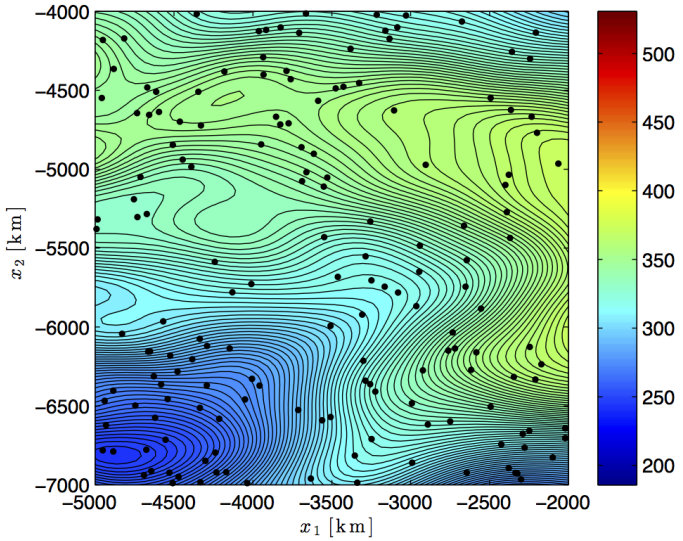
UPPSALA UNIVERSITET

Online learning for distribution-free prediction

Problem formulation: Predict y given $\mathbf{x}$ using observed data generated by some **unknown** distribution $(\mathbf{x}, y) \sim p_0(\mathbf{x}, y)$.

Goals: Learn a predictor function $\hat{y}(\mathbf{x})$

1. in an online manner as $n = 1, 2, \ldots$
2. with performance guarantees
3. and with calibrated uncertainty quantification.



Dave Zachariah, Petre Stoica and Thomas B. Schön. **Online learning for distribution-free prediction**. Pre-print *arXiv:1703.05060*, March, 2017.

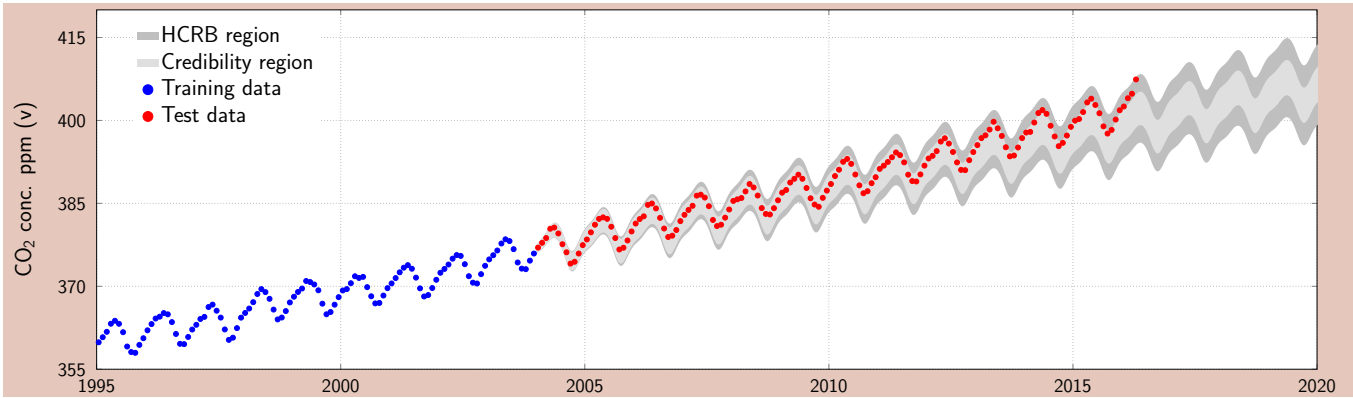
Visit our poster after the talk

GP prediction performance after learning

Problem formulation: Quantify the uncertainty of the prediction $f(x_\star)$ at a test point $x_\star$ when $f(x)$ is a GP model that has been learnt from data $\{x_i, y_i\}_{i=1}^N$, where $y = f(x) + \varepsilon$.

Derive a **fundamental lower bound** of the MSE when the hyper-parameters used in the GP for f are learnt from data.

Available in **closed-form** and **simple and cheap to compute**.



Johan Wågberg, Dave Zachariah, Thomas B. Schön and Petre Stoica. **Prediction performance after learning in Gaussian process regression**. In *AISTATS*, Fort Lauderdale, FL, USA, April, 2017.

Visit our poster after the talk

UPPSALA UNIVERSITET

Conclusions

Constructed a Gaussian process state space model.

Prior = regularization, helping the model to generalize without sacrificing the flexibility offered by the basis function expansion.

A **flexible** model often gives the best performance.

The resulting learning problem require **approximations**.

SMC highly useful in computing these approximations.

Hosting the **SMC workshop in Uppsala**, Aug. 30-Sep. 1, 2017.

www.it.uu.se/conferences/smc2017