UPPSALA
UNIVERSITET

Machine learning – trends and tools

"introducing the field and some of its key concepts"

Thomas Schön, Uppsala University

Elekta, Stockholm

2018-05-24

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

"Anyone making confident predictions about anything having to do with the future of artificial intelligence is either kidding you or kidding themselves."

Andrew McAfee, MIT

What we do in the team

We automate the extraction of knowledge and understanding from data.

Both basic research **and** applied research (with companies).




Create **probabilistic models** for dynamical systems and their surroundings.

Develop methods to **learn** models from data.

The models can then be used by machines (or humans) to **understand** or **take decisions** about what will happen next.

What do I hope to achieve today?

1. Briefly introduce the scientific field of Machine Learning.
2. Create an **awareness/interest** around this technology.
3. Course style introduction to the Gaussian process.

What is machine learning all about?

Machine learning is about learning, reasoning and acting based on data.

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

"It is one of today's most rapidly growing technical fields, lying at the intersection of computer science and statistics, and at the core of artificial intelligence and data science."

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

Jordan, M. I. and Mitchell, T. M. **Machine Learning: Trends, perspectives and prospects**. *Science*, 349(6245):255-260, 2015.

A probabilistic approach

Machine learning is about methods allowing computers/machines **automatically make use of data to solve tasks**.

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.

Learn the unknown variables from the data.

The four 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.

Cornerstone 4 **(Control)** Use the understanding of the current situation to steer it into a desired state.

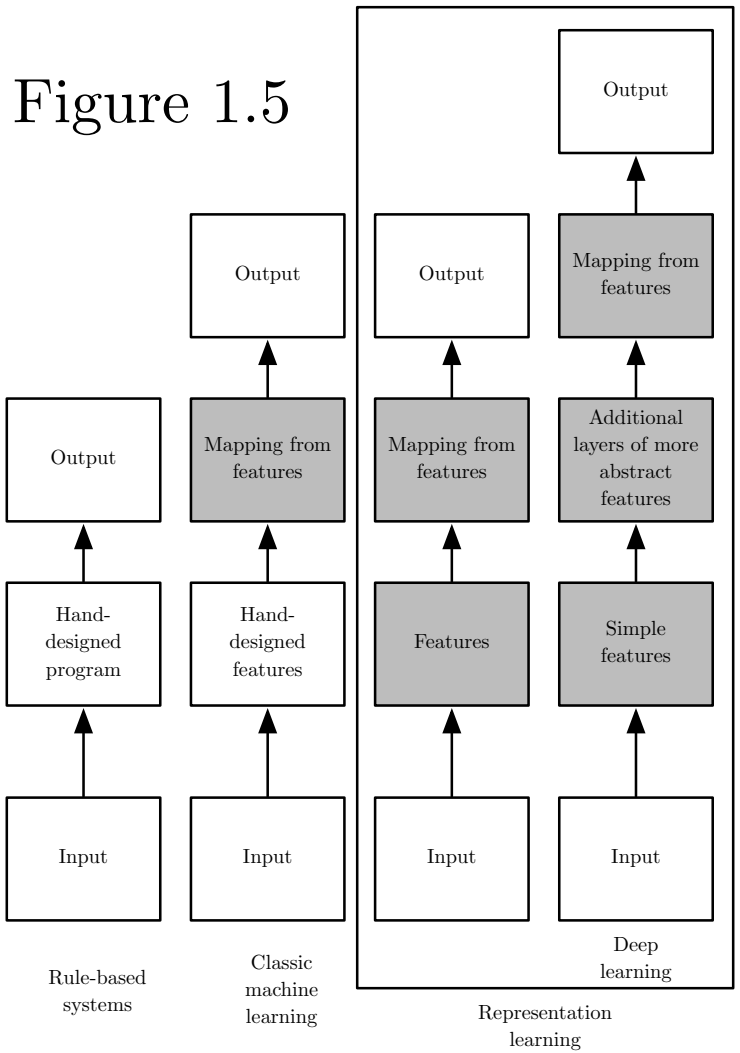
Mathematical models – representations

The performance of an algorithms typically depends on which representation that is used for the data.

Learned representations often provide better solutions than hand-designed representations.

When solving a problem – start by thinking about **which model/representation to use**!

Representation learning



From http://www.deeplearningbook.org/

Problem: How can we **learn** good representations of data?

Ex. Deep learning (DL) solves the problem by introducing representations that are expressed in terms of other, simpler representations.

International Conference on Learning Representations
http://www.iclr.cc/

The two basic rules from probability theory

Let x and y be continuous random variables. Let $p(\cdot)$ denote a general probability density function.

1. Marginalization (integrate out a variable):

$$p(x) = \int p(x, y) \mathrm{d}y.$$

2. Conditional probability:

$$p(x, y) = p(x \mid y) p(y).$$

Combine them into Bayes' rule:

$$p(y \mid x) = \frac{p(x \mid y) p(y)}{p(x)} = \frac{p(x \mid y) p(y)}{\int p(x \mid y) p(y) \mathrm{d}y}.$$

Key objects in learning a model

D - measured data.

z - unknown model variables.

The **full probabilistic model** is given by

$$p(D, z) = \underbrace{p(D \mid z)}_{\text{data distribution}} \underbrace{p(z)}_{\text{prior}}$$

Inference amounts to computing the **posterior distribution**

$$p(z \mid D) = \frac{\overbrace{p(D \mid z)}^{\text{data distribution}} \overbrace{p(z)}^{\text{prior}}}{\underbrace{p(D)}_{\text{model evidence}}}$$

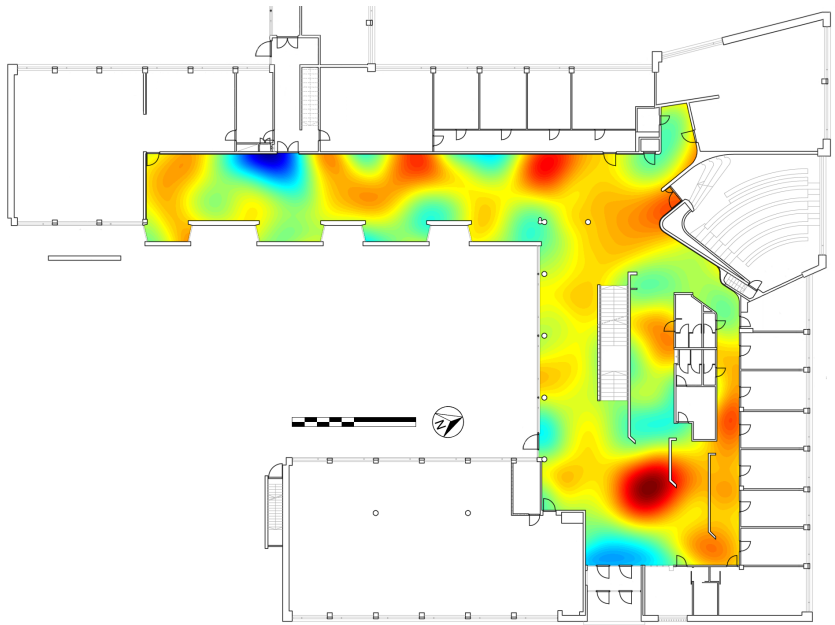
Let's make this more concrete.

Ex) Ambient magnetic field map

The Earth's magnetic field sets a background for the ambient magnetic field. Deviations make the field vary from point to point.

Aim: Build a map (i.e., a model) of the magnetic environment based on magnetometer measurements.

Solution: Customized Gaussian process that obeys Maxwell's equations.



www.youtube.com/watch?v=enlMiUqPVJo

Arno Solin, Manon Kok, Niklas Wahlström, Thomas B. Schön and Simo Särkkä. **Modeling and interpolation of the ambient magnetic field by Gaussian processes**. *IEEE Transactions on Robotics*, 2018.

Carl Jidling, Niklas Wahlström, Adrian Wills and Thomas B. Schön. **Linearly constrained Gaussian processes**. *Advances in Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, December, 2017.

The model – learning relationship

The problem of learning (estimating) a model based on data leads to computational challenges, both

- **Integration:** e.g. the HD integrals arising during marg. (averaging over all possible parameter values z):

$$p(D) = \int p(D \mid z)p(z)\mathrm{d}z.$$

- **Optimization:** e.g. when extracting point estimates, for example by maximizing the posterior or the likelihood

$$\widehat{z} = \arg\max_{z} p(D \mid z)$$

Typically impossible to compute exactly, use approximate methods

- Monte Carlo (MC), Markov chain MC (MCMC), and sequential MC (SMC).
- Variational inference (VI).
- Stochastic gradient.

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 compared with the data set. (**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 access to 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.

Outline

1. What is machine learning?
2. A closer look at the models
3. **Two concrete examples of flexible models**
   a) **Deep learning**
   b) **Gaussian processes**
4. A few other trends/tools
5. Some current research snapshots)
6. Conclusion

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

Deep learning – what is it?

The mathematical model has been around for 70 years, but over the last 5 years there has been a **revolution**. Key reasons:

1. Very large datasets
2. Better and faster computers
3. Enormous industrial interest (e.g. Google, Facebook, MS)
4. Some methodological breakthroughs

The underlying model is a big mathematical function with **multiple layers of abstraction**, commonly with millions of parameters.

The parameter values are **automatically** determined based on a large amount of training data.

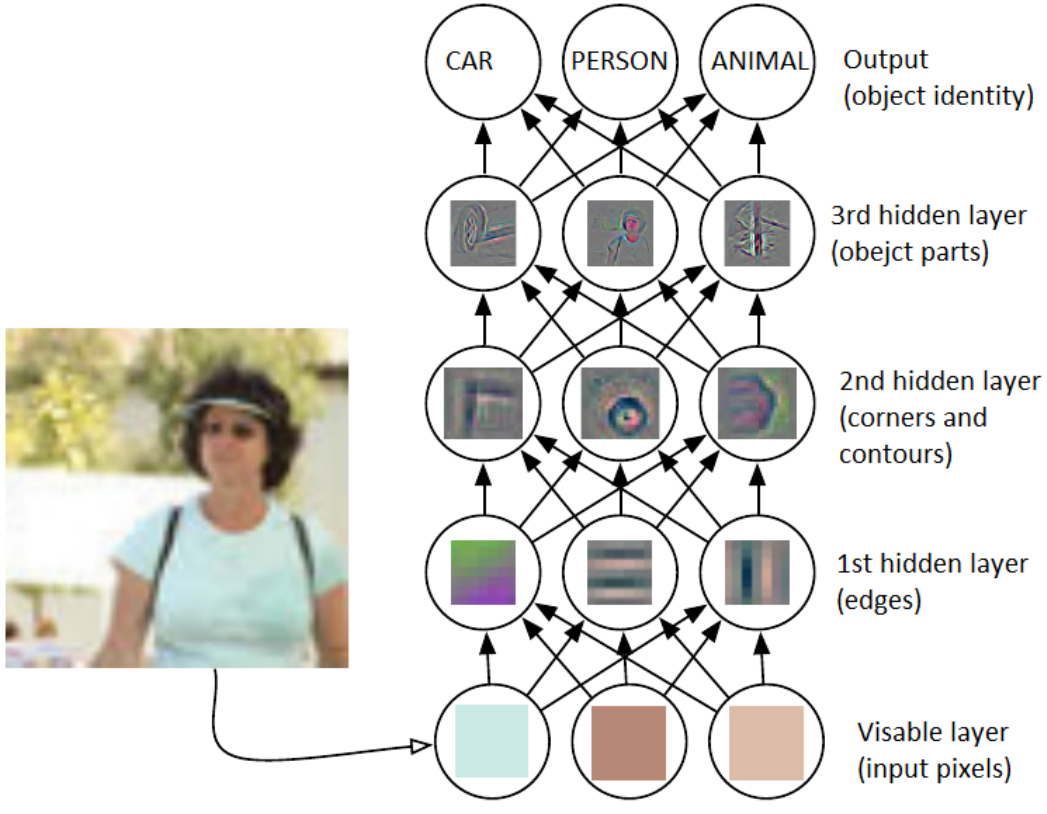
Why deep? - Image classification example

Image classification (Input: pixels of an **image**. Output: **object identity**.)

1 megapixel (B/W)
$\rightarrow 2^{1\,000\,000}$ possible images!

A deep neural network can solve this with a few million parameters!

Each hidden layer extracts increasingly abstract features.



Zeiler, M. D. and Fergus, R. **Visualizing and understanding convolutional networks**. In *European Conference on Computer Vision (ECCV)*, Zürich, Switzerland, September, 2014.

Constructing an NN for regression

A **neural network (NN)** is a hierarchical nonlinear function $y = g_\theta(x)$ from an input variable x to an output variable y parameterized by θ.

Linear regression models the relationship between a continuous output variable y and an input variable x,

$$y = \sum_{i=1}^{n} x_i \theta_i + \theta_0 + \varepsilon = x^\mathsf{T} \theta + \varepsilon,$$

where θ is the parameters composed by the "weights" θ_i and the offset ("bias") term θ_0,

$$\theta = \begin{pmatrix} \theta_0 & \theta_1 & \theta_2 & \cdots & \theta_n \end{pmatrix}^\mathsf{T},$$

$$x = \begin{pmatrix} 1 & x_1 & x_2 & \cdots & x_n \end{pmatrix}^\mathsf{T}.$$

Generalized linear regression and NNs

We can generalize this by introducing nonlinear transformations of the predictor $x^\mathsf{T}\theta$,

$$y = f(x^\mathsf{T}\theta).$$

We can think of the neural network as a **sequential construction** of several generalized linear regressions.

Each layer in a multi-layer NN is modelled as

$$z^{(l+1)} = f\left(\Theta^{(l+1)} z^{(l)} + \theta_0^{(l+1)}\right),$$

starting with the input $z^{(0)} = x$. (The nonlinearity operates element-wise.)

Deep neural networks

Let the computer **learn from experience** and understand the situation in terms of a **hierarchy of concepts**, where each concepts is defined in terms of its relation to simpler concepts.

If we draw a graph showing these concepts of top of each other, the graph is **deep**, hence the name deep learning.

It is accomplished by using **multiple levels of representation**. Each level transforms the representation at the previous level into a new and more abstract representation,

$$z^{(l+1)} = f\left(\Theta^{(l+1)} z^{(l)} + \theta_0^{(l+1)}\right),$$

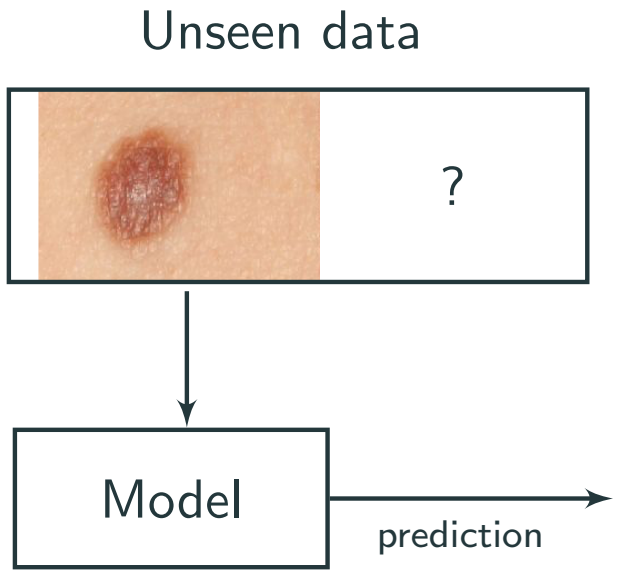
starting from the input (raw data) $z^{(0)} = x$.

Key aspect: The layers are **not** designed by human engineers, they are generated from (typically lots of) data using a learning procedure and lots of computations.

Deep learning – example (skin cancer)

Start from a mathematical model trained on 1.28 million images (**transfer learning**). Make minor modifications of it, specializing to present situation.

Learn new model parameters using 129 450 clinical images ($\sim$ 100 times more images than any previous study).



The results are on par with professional dermatologists on specific tasks. Still, far from being clinically useful, but at least they give us "valid reasons to remain cautiously optimistic" as someone said.

Andre Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M. and Thrun, S. **Dermatologist-level classification of skin cancer with deep neural networks**. *Nature*, 542, 115–118, February, 2017.

Deep learning – Want to know more?

Good introduction

LeCun, Y., Bengio, Y., and Hinton, G. (2015) **Deep learning**, *Nature*, 521(7553), 436–444.

Timely introduction (target audience include: software engineers who do not have a machine learning or statistics background)

I. Goodfellow, Y. Bengio and A. Courville **Deep learning**. http://www.deeplearningbook.org

Deep learning summer school 2018

http://grammars.grlmc.com/DeepLearn2018

Geoffrey Hinton's Coursera course

https://www.coursera.org/learn/neural-networks/home/welcome

Lecture 8-9 in our new MSc level course on Statistical ML

http://www.it.uu.se/edu/course/homepage/sml/lectures

NIPS and ICML conferences and workshops!

The Gaussian process is a model for nonlinear functions

Q: Why is the Gaussian process used everywhere?

It 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.

An abstract idea

In probabilistic (Bayesian) linear regression

$$y_t = \underbrace{\beta^\mathsf{T} \mathbf{x}_t}_{f(\mathbf{x}_t)} + e_t, \qquad e_t \sim \mathcal{N}(0, \sigma^2),$$

we place a prior on β, e.g. $\beta \sim \mathcal{N}(0, \alpha^2 I)$.

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

$$f \sim p(f)$$

and look for $p(f \mid y_{1:T})$ rather than $p(\beta \mid y_{1:T})$?!

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

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

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

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

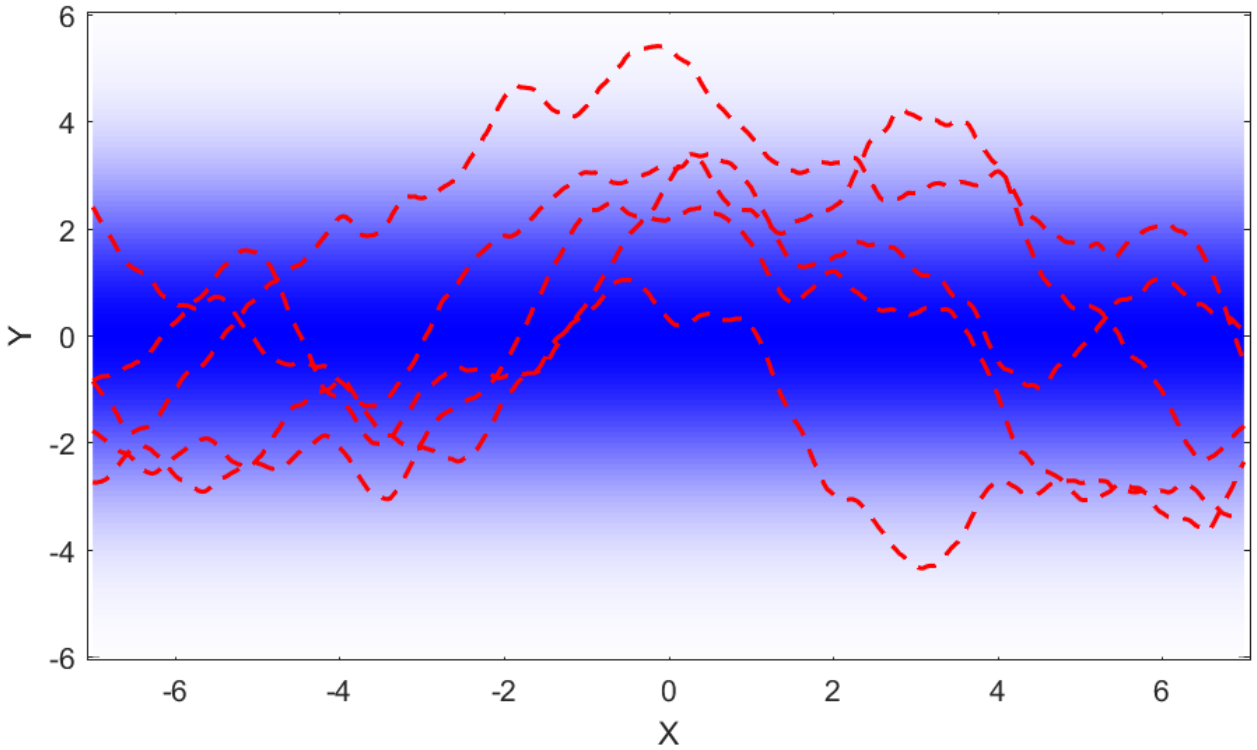
Definition

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.

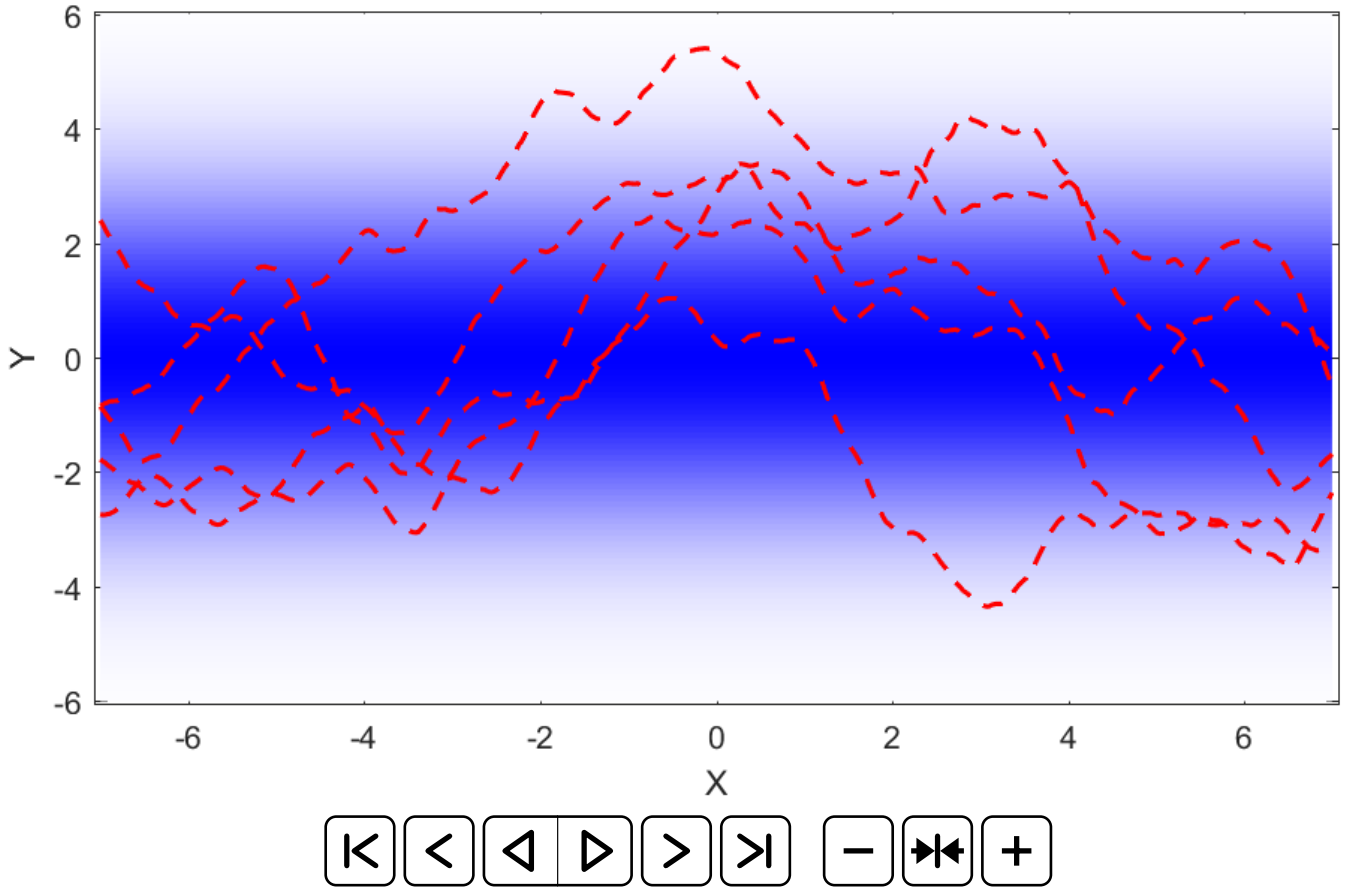
We now have a prior!

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

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



GP regression – illustration



Gaussian Process – Want to know more?

Classic monograph on Gaussian processes

Rasmussen, C. E. and Williams, C. K. I. **Gaussian processes for machine learning**. MIT press Cambridge, MA, 2006.

A review of Bayesian non-parametric modelling

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

Note from our MSc level course

Fredrik Lindsten, Thomas B. Schön, Andreas Svensson and Niklas Wahlström. **Probabilistic modeling – linear regression & Gaussian processes**, 2017. www.it.uu.se/edu/course/homepage/sml/literature/probabilistic_modeling_compendium.pdf

---

The standard GP is too expensive ($\mathcal{O}(N^3)$) to be practically useful, but there are good approximations available, see e.g.

Joaquin Quinonero-Candela and Carl Edward Rasmussen. **A unifying view of sparse approximate Gaussian process regression**. *Journal of Machine Learning Research (JMLR)*, 6:1939–1959, 2005.

Outline

1. What is machine learning?
2. A closer look at the models
3. Two concrete examples of flexible models
   a) Deep learning
   b) Gaussian processes
4. **A few other trends/tools**

(5. Some current research snapshots)

6. Conclusion

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

Probabilistic numerics

Reinterpreting numerical tasks (e.g., linear algebra, int., opt. and solving differential equations) as probabilistic inference problems.

A numerical method **estimates** a certain **latent** property **given** the result of computations, i.e. computation is inference.

Ex: Basic alg. that are equivalent to Gaussian MAP inference

- Conjugate Gradients for linear algebra
- BFGS etc. for nonlinear optimization
- Gaussian Quadrature rules for Integration
- Runge-Kutta solvers for ODEs

Hennig, P., Osborne, M., Girolami, M. **Probabilistic numerics and uncertainty in computations**. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 471(2179), 2015.

Adrian Wills and Thomas B. Schön. **Stochastic quasi-Newton with adaptive step lengths for large-scale problems.** *arXiv:1802.04310*, February, 2018.

Bayesian optimization

Bayesian optimization is a **sequential model-based** approach to solve the problem of **optimizing an unknown (or expensive to evaluate) function** $f(x)$

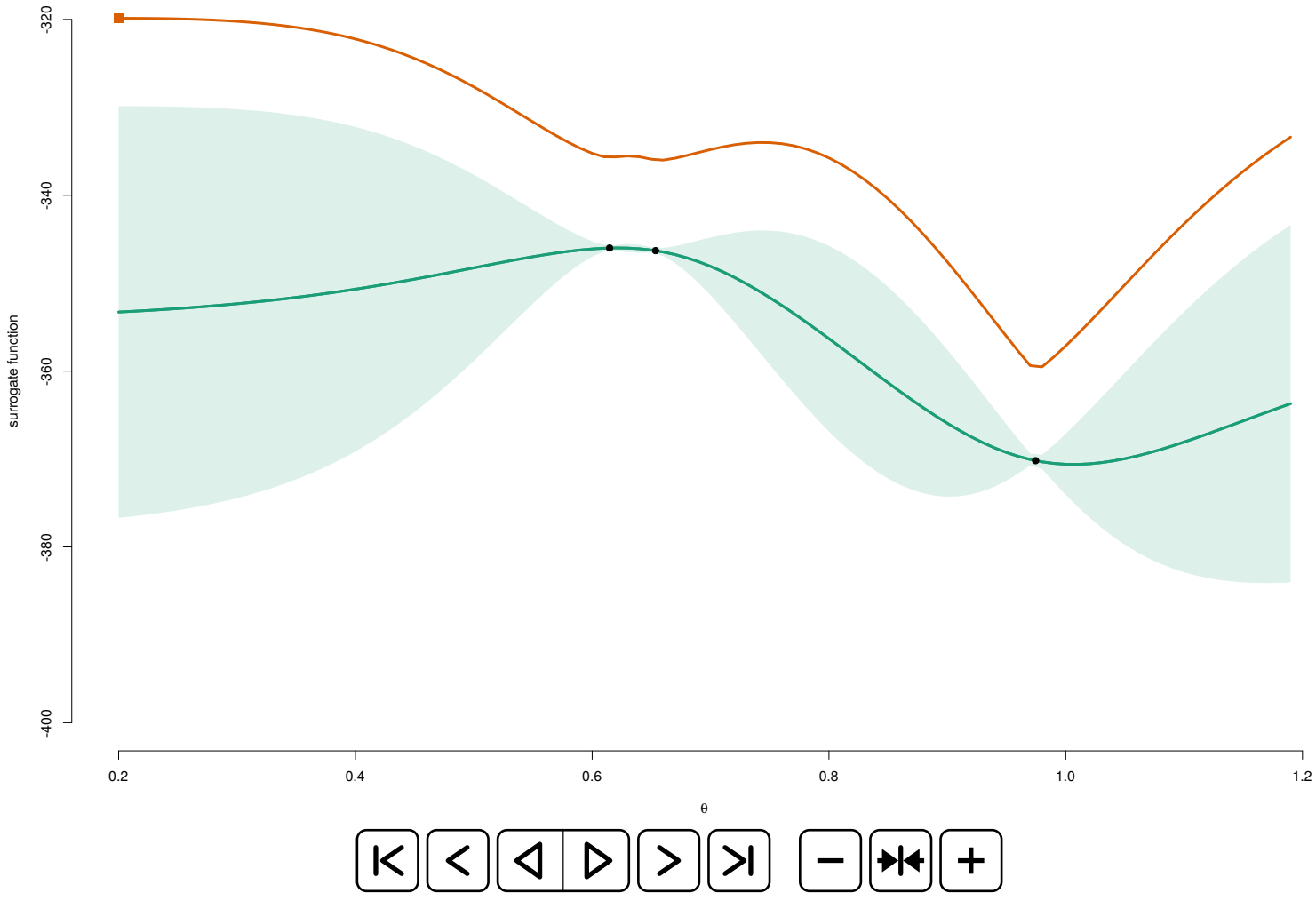
$$x^\star = \arg\max_x f(x)$$

Place a prior over the unknown objective function $f(x)$ (e.g. using a Gaussian process) and sequentially refine it as data becomes available via probabilistic posterior updating.

The posterior distribution is then used to construct an **acquisition function** that determines what the next query point x should be.

Shahriari, B., Swersky, K., Wang, A. Adams, R. P. and de Freitas, N. **Taking the Human Out of the Loop: A Review of Bayesian Optimization**. *Proceedings of the IEEE*, 104(1):148–175, January 2016.

Bayesian optimization – illustration



Probabilistic programming

Probabilistic programming makes use of computer programs to represent probabilistic models.

Probabilistic programming lies on the interesting intersection of

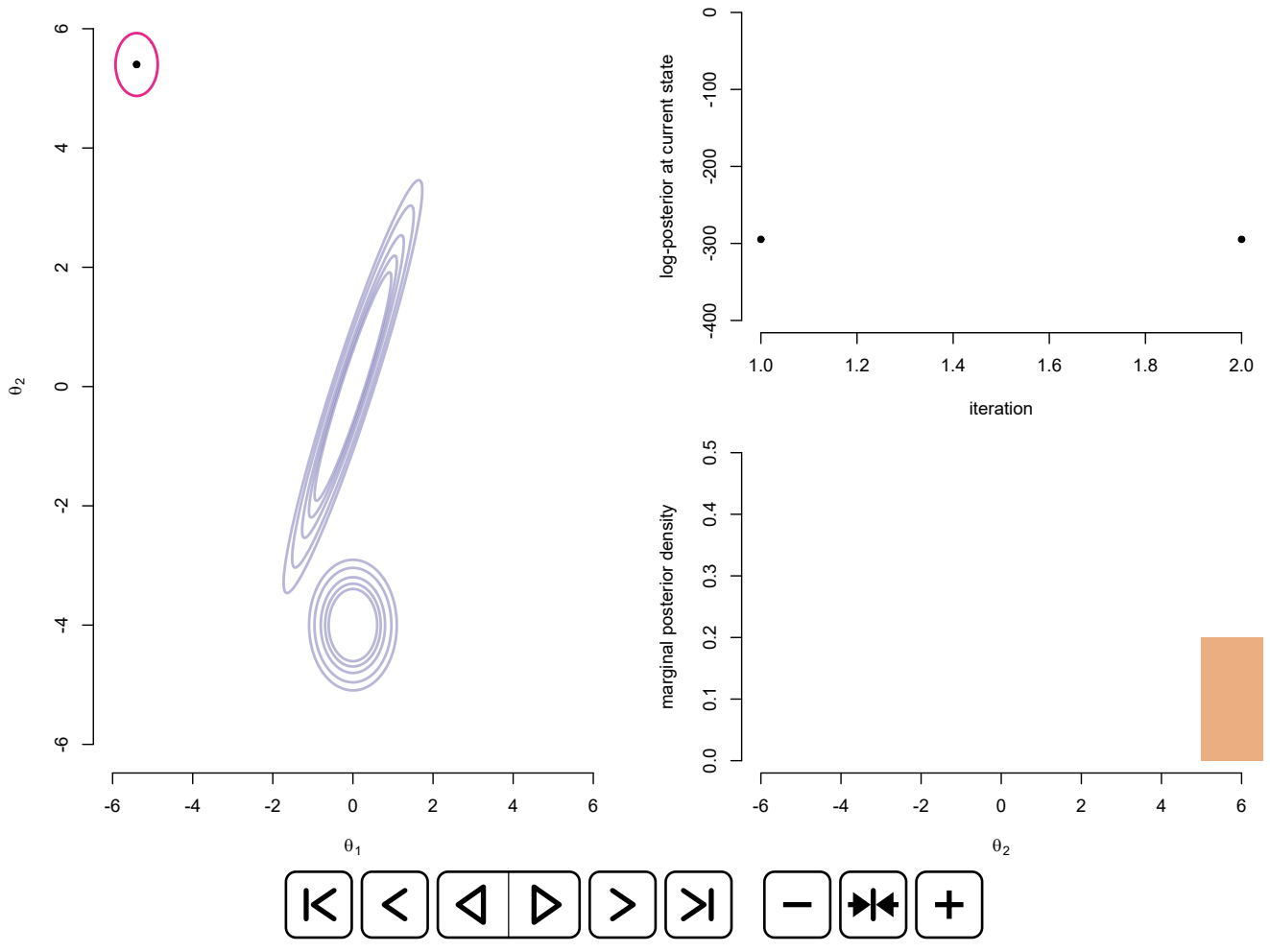
1. Programming languages: Compilers and semantics.
2. Machine learning: Algorithms and applications.
3. Statistics: Inference and theory.

Creates a clear **separation** between the model and the inference methods, encouraging model based thinking. Automate inference!

---

http://probabilistic-programming.org/

Markov chain Monte Carlo – toy example



Animation made by Johan Dahlin

Markov chain Monte Carlo (MCMC)

Represent distributions using a large number of samples.

Markov chain Monte Carlo (MCMC) methods are used to sample from a probability distribution by **simulating a Markov chain** that has the desired distribution as its stationary distribution.

Used to compute numerical approximations of intractable integrals.

Constructive algorithms:

1. The **Metropolis Hastings** sampler
2. The **Gibbs** sampler

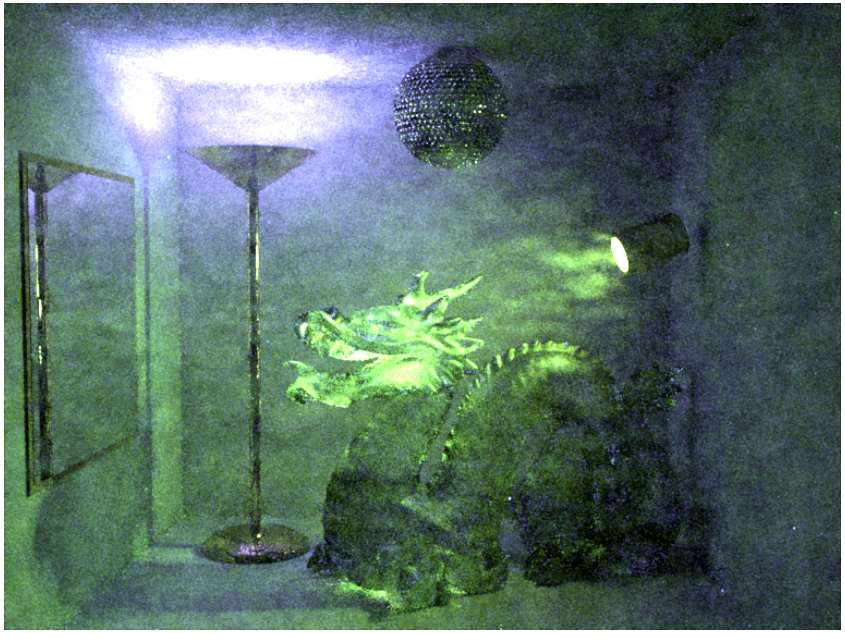
Andrieu, C. De Freitas, N. Doucet, A. and Jordan, M. I. **An Introduction to MCMC for Machine Learning**, *Machine Learning*, 50(1): 5-43, 2003

“Visualizing” a Markov chain

Built a Markov chain to sample light paths connecting the sensor with light sources in the scene.

Results using equal time rendering

Our method that builds on MLT



Metropolis light transport (MLT)

Joel Kronander, Thomas B. Schön and Jonas Unger. **Pseudo-marginal Metropolis light transport**. In *Proceedings of SIGGRAPH ASIA Technical Briefs*, Kobe, Japan, November, 2015.

Sequential Monte Carlo (SMC)

SMC provide approximate solutions to **integration** problems where there is a **sequential structure** present.

Important example where we have a sequential structure present is in dynamical systems, where we call SMC **particle filtering**.

A. Doucet and A. M. Johansen. **A Tutorial on Particle filtering and smoothing: Fiteen years later**. In *The Oxford Handbook of Nonlinear Filtering*, D. Crisan and B. Rozovsky (eds.). Oxford University Press, 2011.

Thomas B. Schön, Fredrik Lindsten, Johan Dahlin, Johan Wagberg, Christian A. Naesseth, Andreas Svensson and Liang Dai. **Sequential Monte Carlo methods for system identification**. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Beijing, China, October 2015.

Variational inference

Variational inference provides an approximation to the posterior distribution by **assuming that it has a certain functional form** that contain unknown parameters.

These unknown parameters are found using **optimization**, where some distance measure is minimized.

Variational inference methods are used to approximate intractable integrals and are thus an alternative to MCMC.

Ex. Variational Bayes (VB) and expectation propagation (EP).

Blei, D. Kucukelbir, A. and McAuliffe, J. **Variational inference: A review for statisticians**, *Journal of the American Statistical Association*, 112(518):859–877, 2017.

A few other trends/tools (brief)

1. **Probabilistic numerics** – Reinterpreting numerical tasks as probabilistic inference problems.
2. **Bayesian optimization** – A sequential model-based approach to solve the problem of optimizing an unknown function $f(x)$.
3. **Probabilistic programming** – Makes use of computer programs to represent probabilistic models.
4. Two strategies to approximate intractable traget distributions
   - **Markov chain Monte Carlo (MCMC)** – Construct a Markov chain with the traget distribution as its stationary distribution. Then, we sample from the chain to eventually collect independent samples from the stationary distribution.
   - **Variational inference** – Assume a family of distributions and find the member (using optimization) of that family which is closest to the target distribution.

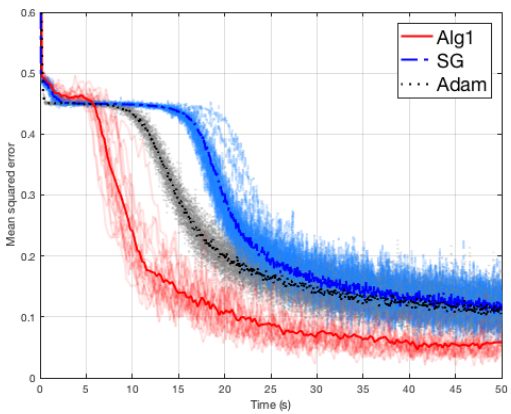
What did I hope to achieve today?

1. Briefly introduce the scientific field of Machine Learning.
2. Create an **awareness/interest** around this technology.
3. Course style introduction to the Gaussian process.

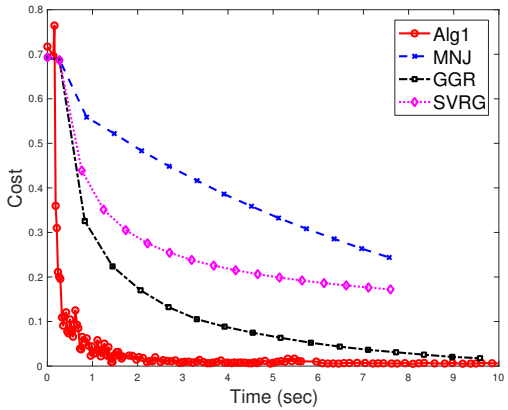
Snapshots of some of our ongoing research

Snapshot 1 – scaling up SG to large problems

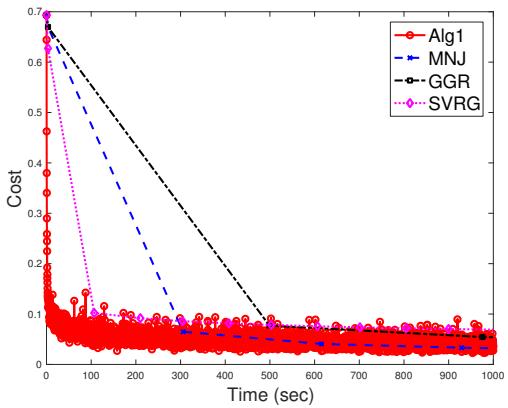
We have developed **stochastic optimization** algorithms that scale to **very large** problems.



Training a deep CNN for MNIST data.



Logistic loss function with an L2 regularizer, `gisette`, 6 000 observations and 5 000 unknown variables.



Logistic loss function with an L2 regularizer, `URL`, 2 396 130 observations and 3 231 961 unknown variables.

Key innovations

- Exploit a receding history of iterates and gradients akin to L-BFGS.
- An auxiliary variable Markov chain construction.

Adrian Wills and Thomas B. Schön. **Stochastic quasi-Newton with adaptive step lengths for large-scale problems**. *arXiv:1802.04310*, February, 2018.

Snapshot 2 – GP-based nonlinear state space model

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

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

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

We can now find the posterior distribution

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

via some approximation (we use **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 *Advances in Neural Information Processing Systems (NIPS)*, 2013.

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

Snapshot 3 – The ASSEMBLE project and Birch

Aim: Automate probabilistic modeling of dynamical systems (and their surroundings) via a formally defined **probabilistic modeling language**.

Keep the model and the learning algorithms **separated**.

Create a **market place** for SMC-based learning algorithms (think CVX).

Birch — Our prototype probabilistic programming language.

Lawrence M. Murray, Daniel Lundén, Jan Kudlicka, David Broman and Thomas B. Schön. **Delayed sampling and automatic Rao-Blackwellization of probabilistic programs**. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, Lanzarote, Spain, April, 2018.

Birch - our prototype probabilistic programming language

1. The basic idea of **probabilistic programming** is to equate probabilistic models with the programs that implement them.
2. Just as we can think of doing inference over models, we can think of doing **inference over programs**.

The particular PPL used here is **Birch**, which is currently being developed at Uppsala University.

Probabilistic and object-oriented language.

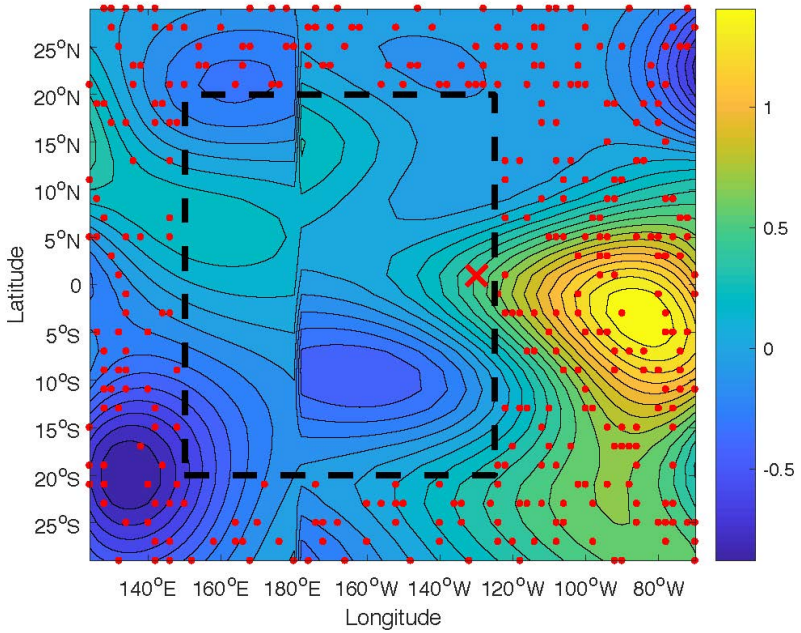
An early pre-release of Birch is available

`birch-lang.org`

Snapshot 4 – Spatio-temporal modelling using GPs

Problem: predicting **spatio-temporal** processes with temporal patterns varying across spatial regions, when data is obtained as a **stream**.

The learned predictor can be **updated sequentially** with each new data point.



Muhammad Osama, Dave Zachariah and Thomas B. Schön. **Learning localized spatio-temporal models from streaming data.** In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, Stockholm, Sweden, July, 2018.

MSc course in Statistical Machine Learning

Given twice (2017 - 2018).

Introductory course to statistical machine learning (SML) focusing on classification and regression. Topics include:

- Regression, classification and boosting
- Regularization (ridge regression and the LASSO)
- Regression and classification trees
- Deep learning and neural networks

Deep learning lab: See how a state-of-the-art deep learning algorithm performs at classifying real world images.

Mini project: Predict which songs one of the TAs like the most.

www.it.uu.se/edu/course/homepage/sml

Conclusion

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

Uncertainty is a key concept!

The best predictive performance is currently obtained from **highly flexible** learning systems, e.g.

1. Deep learning
2. Gaussian processes

Remember to talk to people who work on **different problems** with **different tools**!! (Visit other fields!)