GP-based probabilistic modelling of dynamical systems

Thomas Schön, Uppsala University

The University of Sheffield, Gaussian Process Summer School, September 4, 2018.
What we do in the team — who we are

We automate the extraction of knowledge and understanding from data.

Both basic research and applied research (with companies).

Create probabilistic models of 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.
A dynamical system evolves over time and it has a memory.

\[ u(t) \rightarrow ? \rightarrow y(t) \]

Ex. 1 Linear time-invariant dynamical system described by

\[ y(t) = \int_{0}^{\infty} g(\tau) u(t - \tau) d\tau + e(t). \]

Ex. 2 Nonlinear autoregressive model with exogenous (NARX) inputs

\[ y_t = \varphi(y_{t-1}, \ldots, y_{t-n_y}, u_t, \ldots, u_{t-n_u}) + e_t. \]

Ex. 3 State space model (using latent variables \( x_t \))

\[ x_{t+1} = f(x_t, u_t, \theta) + v_t, \]
\[ y_t = g(x_t, u_t, \theta) + e_t. \]
Ex) “What are $x_t$, $\theta$ and $y_t$”?

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

![Image of motion capture setup]

**Show movie!**

Flexible models often give the best performance.

How can we build and work with these flexible models?

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

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

Today we will focus on using the **Gaussian process in modelling dynamical systems.**
What is a Gaussian process (GP)?

The Gaussian process is a non-parametric and probabilistic model of a nonlinear function.

- **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.

**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.
Aim and outline

**Aim:** Provide some useful answers to the following two questions:

1. **General:** How can we mathematically construct probabilistic models of dynamical systems?
2. **Specific:** How can the GP be used to model dynamical systems?

1. Introduction – What is a dynamical system?

2. **Linear dynamical systems**
   a) Impulse response estimation
   b) Autoregressive (AR)
   c) Linear state space model (SSM)

3. Nonlinear dynamical systems
   a) Nonlinear AR
   b) Nonlinear SSM (GP-SSM)

4. Snapshots of some ongoing research (if there is time)

Comment on 44 and $\mu$. 

6/44
Part 2 – Linear dynamical systems
A fundamental concept: The **impulse response** $g(\tau)$ provides knowledge about **everything** there is to know about a linear system.

$$u(t) \rightarrow y(t) \quad \quad y(t) = \int_{0}^{\infty} g(\tau) u(t - \tau) d\tau.$$ 

The impulse response of a dynamical system is its **output** when presented with an "impulse" input signal.

This **impulse** (the Dirac delta function) models the density of an idealized point mass as a function equal to zero everywhere except for zero and whose integral over the entire real line is equal to one.

The impulse function **contains all frequencies**, which means that the impulse response defines the response of a linear time-invariant system for all frequencies.
Consider a linear time-invariant dynamical system described by

\[ y(t) = \int_{0}^{\infty} g(\tau) u(t - \tau) d\tau + e(t). \]

**Task:** Learn a model of the true underlying impulse response \( g(\tau) \).

Placing a GP prior on the impulse response offers better performance than the “classical” system identification approach.


Note that the integral of a GP is also a GP, so this is rather natural.

The GP offers a **data-driven model flexibility tuning**, an automatic **regularization** striking a bias-variance trade-off that is “just right”.
Two common parametric representations

1. One of the simplest parametric rep. is provided by the AR(X) model.

\[ y_t = \varphi(y_{t-1}, \ldots, y_{t-n}, u_t, \ldots, u_{t-n_u}) + e_t. \]

These “classic” parametric approaches and the GP-based impulse response approach are linked via a decision-theoretic formulation.


2. Another very useful parametric representation is offered by introducing latent variables, which results in the so-called state space model.
An autoregressive model of order $n$ is given by

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_n y_{t-n} + e_t, \quad e_t \sim \mathcal{N}(\mu, \tau^{-1})$$

$$= \theta^T z_t + e_t,$$

where $\mu$ and $\tau$ are known explanatory variables ($\mu = 0, \tau \neq 0$).

The unknown model variables are collected as

$$\theta = (a_1, a_2, \ldots, a_n)^T$$

with the prior

$$\theta \sim \mathcal{N}(0, \rho^{-1} I_n), \quad \text{where } \rho \text{ assumed to be known.}$$

**Task:** Compute the posterior $p(\theta \mid y_{1:T})$. 
Bayesian autoregressive model

Full probabilistic model \( p(\theta, y_{1:T}) = p(y_{1:T} \mid \theta)p(\theta) \), where the data distribution is given by

\[
p(y_{1:T} \mid \theta) = p(y_T \mid y_{1:T-1}, \theta)p(y_{1:T-1} \mid \theta) = \cdots = \prod_{t=1}^{T} p(y_t \mid y_{1:t-1}, \theta).
\]

From the model we have that

\[
p(y_t \mid y_{1:t-1}, \theta) = \mathcal{N}(y_t \mid \theta^T z_t, \tau^{-1}),
\]

where \( z_t = (y_{t-1}, y_{t-2}, \ldots, y_{t-n})^T \). Hence,

\[
p(y_{1:T} \mid \theta) = \prod_{t=1}^{T} \mathcal{N}(y_t \mid \theta^T z_t, \tau^{-1}) = \mathcal{N}(\mathbf{y} \mid \mathbf{z} \theta, \tau^{-1} I_T),
\]

where we have made use of \( \mathbf{y} = (y_1, y_2, \ldots, y_T)^T \) and \( \mathbf{z} = (z_1, z_2, \ldots, z_T)^T \).
\[ p(\theta, y) = \mathcal{N}(y \mid z\theta, \tau^{-1}I_T) \mathcal{N}(\theta \mid 0, \rho^{-1}I_n) \]

\[
= \mathcal{N}\left(\begin{pmatrix} \theta \\ y \end{pmatrix} \mid \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \rho^{-1}l_2 & \rho^{-1}z^T \\ \rho^{-1}z & \tau^{-1}I_T + \rho^{-1}zz^T \end{pmatrix}\right) .
\]

The posterior is given by

\[ p(\theta \mid y) = \mathcal{N}(\theta \mid m_T, S_T) , \]

where

\[
m_T = \tau S_T z^T y ,
\]

\[
S_T = \left( \rho^{-1}l_2 + \sigma z^T z \right)^T .
\]
\( y_t = a_1 y_{t-1} + a_2 y_{t-2} + e_t, \quad e_t \sim \mathcal{N}(0, 0.2). \)
\( \mu: \) Ex) Situation after \( y_1 \) is obtained

Likelihood

Posterior

7 samples from the posterior
μ: Ex) Situation after $y_{1:2}$ and $y_{1:20}$

Likelihood  Posterior  7 samples from the posterior
An abstract idea

In Bayesian linear regression

\[ y_t = \theta^T z_t + e_t, \quad e_t \sim \mathcal{N}(0, \sigma^2), \]

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

(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 \( p(\theta \mid y_{1:T}) \)?
An autoregressive model with exogenous (ARX) inputs

\[ y_t = \varphi(y_{t-1}, \ldots, y_{t-n}, u_t, \ldots, u_{t-n_u}) + e_t. \]

Place a GP prior over \( \varphi \) with the following input

\[ x_t = \begin{pmatrix} y_{t-1} & \ldots & y_{t-n} & u_{t} & \ldots & u_{t-n_u} \end{pmatrix}^T \]

There is of course no reason to limit ourself to linear models when we are modelling \( \varphi \) using a GP. (a bit more about this later)

---


Model variables that are not observed are called latent (a.k.a. hidden, missing and unobserved) variables.

The idea of introducing latent variables into models is probably one of the most powerful concepts in probabilistic modelling.

Latent variables provide more expressive models that can capture hidden structures in data that would otherwise not be possible.

**Cost:** Learning the model often becomes (significantly) harder.

Standard use within dynamical systems: State space models.
The Markov chain is a probabilistic model that is used for modelling a sequence of states \((x_0, x_1, \ldots, x_T)\).

**Definition (Markov chain)**

A stochastic process \(\{x_t\}_{t \geq 0}\) is referred to as a Markov chain if, for every \(k > 0\) and \(t\),

\[
p(x_{t+k} \mid x_0, x_1, \ldots, x_t) = p(x_{t+k} \mid x_t).
\]

A **Markov chain** is completely specified by:

1. An initial value \(x_0\) and
2. a transition model (kernel) \(\kappa(x_{t+1} \mid x_t)\) describing the transition from state \(x_t\) to state \(x_{t+1}\), according to \(x_{t+1} \mid (x_t = x_t) \sim \kappa(x_{t+1} \mid x_t)\).
The **state** of the Markov chain acts as a **memory** containing all information there is to know about the phenomenon at a particular point in time.

Two important applications of Markov chains:

1. The Markov model is used in the **state space model (SSM)** where we can only observe the state indirectly via a measurement that is related to the state.

2. The Markov chain constitutes the basic ingredient in the **Markov chain Monte Carlo (MCMC)** methods.

Guess I have to mention a third application of the Markov chain as well:

3. Stochastic gradient methods
The linear Gaussian state space model (LG-SSM) is given by

\[ x_t = Ax_{t-1} + Bu_t + v_t, \]
\[ y_t = Cx_t + Du_t + e_t, \]

where \( x_t \in \mathbb{R}^{n_x} \) denotes the state, \( u_t \in \mathbb{R}^{n_u} \) denotes an explanatory variable (known signal) and \( y_t \in \mathbb{R}^{n_y} \) denotes the measurement (data).

The initial state and the noise are distributed according to

\[
\begin{pmatrix}
    x_0 \\
    v_t \\
    e_t
\end{pmatrix} \sim \mathcal{N}
\begin{pmatrix}
    \mu \\
    0 \\
    0
\end{pmatrix},
\begin{pmatrix}
    P_0 & 0 & 0 \\
    0 & Q & S \\
    0 & S^T & R
\end{pmatrix}
\]

Remark: There is a connection between the SSM and the GP, where the GP can sometimes be reformulated as LG-SSMs. Opens up for linear complexity inference via the Kalman filter.

An attempt to illustrate why SSM + GP might make sense

Again, the linear Gaussian state space model

\[
\begin{align*}
x_{t+1} &= Ax_t + v_t, \\
y_t &= Cx_t + e_t.
\end{align*}
\]

is designed to model dynamical behaviour.

However, it is limited in its expressiveness and uncertainty modelling.

Figure 2. An illustration of the advanced uncertainty dynamical modeling possible with the combination of the state-space model and the Gaussian process.
An attempt to illustrate why SSM + GP might make sense

The Gaussian process state space model (GP-SSM)

\[ x_{t+1} = f(x_t) + v_t, \quad \text{where} \quad f \sim \mathcal{GP}, \]
\[ y_t = g(x_t) + e_t, \quad \text{where} \quad g \sim \mathcal{GP}. \]

combines the non-parametric flexibility and the uncertainty representation of the GP with the dynamical modeling capabilities of the SSM.
Aim: Provide some useful answers to the following two questions:

1. **General:** How can we mathematically construct probabilistic models of dynamical systems?
2. **Specific:** How can the GP be used to model dynamical systems?

1. Introduction – What is a dynamical system?
2. Linear dynamical systems
   a) Impulse response estimation
   b) Autoregressive (AR)
   c) Linear state space model (SSM)

3. **Nonlinear dynamical systems**
   a) Nonlinear AR model
   b) Nonlinear SSM (GP-SSM)

4. Snapshots of some ongoing research (if there is time)
Part 3 – Nonlinear dynamical systems
Nonlinear ARX model — GP style

An autoregressive model with exogenous (ARX) inputs

\[ y_t = \varphi(y_{t-1}, \ldots, y_{t-n}, u_t, \ldots, u_{t-n_u}) + e_t. \]

Place a GP prior over \( \varphi \) with the following input

\[ x_t = \begin{pmatrix} y_{t-1} & \ldots & y_{t-n} & u_t & \ldots & u_{t-n_u} \end{pmatrix}^T \]

Challenges (standard challenges with the basic GP):

1. **Computationally** too expensive.
2. It cannot efficiently make use of new measurements **online**.
3. Cannot deal with **stochastic (noisy) inputs**.

As you have heard by now there are ways around all of these.

The state space model (SSM) is a **Markov chain** that makes use of a **latent** variable representation to describe dynamical phenomena.

It consists of two stochastic processes:

1. unobserved (state) process \( \{x_t\}_{t \geq 0} \) modelling the dynamics,
2. observed process \( \{y_t\}_{t \geq 1} \) modelling the measurements and their relationship to the unobserved state process.

\[
\begin{align*}
x_t &= f(x_{t-1}, \theta) + v_t, \\
y_t &= g(x_t, \theta) + e_t,
\end{align*}
\]

where \( \theta \in \mathbb{R}^{n\theta} \) denotes static model parameters.
Three alternative representations, using

1. graphical models,
2. probability distributions or
3. probabilistic programs.

1. Representing the SSM using a graphical model:
2. Representation using probability distributions

\[ x_t \mid (x_{t-1}, \theta) \sim p(x_t \mid x_{t-1}, \theta), \]
\[ y_t \mid (x_t, \theta) \sim p(y_t \mid x_t, \theta), \]
\[ x_0 \sim p(x_0 \mid \theta). \]

3. Representing the SSM using a probabilistic program

\[
\begin{align*}
x[1] & \sim \text{Gaussian}(0.0, 1.0); & p(x_1) \\
y[1] & \sim \text{Gaussian}(x[1], 1.0); & p(y_1 \mid x_1) \\
\text{for (t in 2..T) }
\begin{cases}
  x[t] & \sim \text{Gaussian}(a*x[t - 1], 1.0); & p(x_t \mid x_{t-1}) \\
  y[t] & \sim \text{Gaussian}(x[t], 1.0); & p(y_t \mid x_t)
\end{cases}
\end{align*}
\]

A probabilistic program encodes a probabilistic model (here an LG-SSM) according to the semantics of a particular probabilistic programming language (here Birch).
The full probabilistic model is given by

\[
p(x_{0:T}, \theta, y_{1:T}) = p(y_{1:T} \mid x_{0:T}, \theta) p(x_{0:T}, \theta)
\]

Distribution describing a parametric nonlinear SSM

\[
p(x_{0:T}, \theta, y_{1:T}) = \prod_{t=1}^{T} p(y_t \mid x_t, \theta) \prod_{t=1}^{T} p(x_t \mid x_{t-1}, \theta) p(x_0 \mid \theta) p(\theta)
\]
Based on our generative model, compute the **posterior distribution**

\[
p(x_{0:T}, \theta \mid y_{1:T}) = \frac{p(x_{0:T} \mid \theta, y_{1:T}) p(\theta \mid y_{1:T})}{p(y_{1:T})}.
\]

**Bayesian** formulation – model the unknown parameters as a random variable \( \theta \sim p(\theta) \) and compute

\[
p(\theta \mid y_{1:T}) = \frac{p(y_{1:T} \mid \theta)p(\theta)}{p(y_{1:T})}
\]

**Maximum likelihood** formulation – model the unknown parameters as a deterministic variable and solve

\[
\hat{\theta} = \arg \max_{\theta \in \Theta} p(y_{1:T} \mid \theta).
\]
μ: Central object – the likelihood

The likelihood is computed by marginalizing

\[ p(x_0:T, y_1:T \mid \theta) = p(x_0 \mid \theta) \prod_{t=1}^{T} p(y_t \mid x_t, \theta) \prod_{t=1}^{T} p(x_t \mid x_{t-1}, \theta), \]

w.r.t the state sequence \( x_0:T \),

\[ p(y_1:T \mid \theta) = \int p(x_0:T, y_1:T \mid \theta) dx_0:T. \]

(We are averaging \( p(x_0:T, y_1:T \mid \theta) \) over all possible state sequences.)

Equivalently we have

\[ p(y_1:T \mid \theta) = \prod_{t=1}^{T} p(y_t \mid y_{1:t-1}, \theta) = \prod_{t=1}^{T} \int p(y_t \mid x_t, \theta) p(x_t \mid y_{1:t-1}, \theta) dx_t. \]

---

Learning a model based on data leads to computational challenges:

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

  $$p(y_{1:T}) = \int p(y_{1:T} | z)p(z)dz.$$ 

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

  $$\hat{z} = \arg \max_z p(y_{1:T} | z)$$

Impossible to compute exactly, approximations are needed:

- Monte Carlo (MC), Markov chain MC, and sequential MC.
- Variational inference (VI).
- Stochastic optimization.
The idea underlying a non-parametric SSM via the GP

\[
x_{t+1} = f(x_t) + v_t, \quad \text{s.t.} \quad f(x) \sim \mathcal{GP}(0, \kappa, f(x, x')), \\
y_t = g(x_t) + e_t, \quad \text{s.t.} \quad g(x) \sim \mathcal{GP}(0, \kappa, g(x, x')).
\]

Results in a **flexible** non-parametric model where the GP prior takes on the **role of a regularizer**. Enables regularization also in nonlinear models.

**Challenge:** Approximate the posterior distribution

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

---


Approximate Gaussian processes

We use a “reduced-rank” GP approximation:

\[ f \sim \mathcal{GP}(0, k) \iff f(x) \approx \sum_{j=0}^{m} w_j \phi_j(x) \]

with prior

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

For \( x \in [-L, L] \subset \mathbb{R} \):

\[ \phi^j(x) = \frac{1}{\sqrt{L}} \sin \left( \frac{\pi j(x+L)}{2L} \right). \]

Computationally feasible GP-SSM

Original formulation:

\[ x_{t+1} = f(x_t) + \nu_t, \quad \nu_t \sim \mathcal{N}(0, Q), \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R), \]
\[ f(x) \sim \mathcal{GP}(0, \kappa_{\eta, f}(x, x')) \]

Formulation using the reduced-rank GP approximation:

\[ x_{t+1} = \sum_{j=0}^{m} w_j \phi_j(x_t) + \nu_t, \quad \nu_t \sim \mathcal{N}(0, Q), \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R), \]
\[ w^j \sim \mathcal{N}(0, S(\lambda^j)). \]

Linear in the parameters \( w^j \) and nonlinear in the states \( x_t \).
The learning problem (dynamical systems)

Compute the posterior distribution

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

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 \( \theta \) is rather straightforward in this GP-SSM.

The states \( x_{1:T} \) are still challenging. We use a combination of SMC and MCMC.

GP-SSM — A Zoubin cube to describe it

The GP for time series modelling

Control for GP time series models (GP-MPC)

State estimation in the GP-SSM

GP regression

System identification

Reinforcement learning, PILCO

System identification of the GP-SSM

Control with unknown states (GP-MPC, GP-POMDP)

Joint learning and control with unobserved states

Latent states

Control

Latent states

System identification

Control

Latent states

System identification

Control
Part 4 – Snapshots of some ongoing research and message
**Innovation:** Modification of the covariance function in a GP to correctly account for **known linear operator** constraints.

**Contribution:**

1. A probabilistic model that is **guaranteed** to fulfil known linear operator constraints.

2. A **constructive procedure** for designing the transformation.
**Tomographic reconstruction goal:** Build a map of an unknown quantity within an object using information from irradiation experiments.

Ex1) Modelling and reconstruction of strain fields.

Ex2) Reconstructing the internal structure from limited x-ray projections.
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


Machine learning with state-space models, Gaussian processes and Monte Carlo methods.

By Andreas Svensson

Link to the thesis:

www.it.uu.se/katalog/andsv164/main/thesis_andreas_svensson_webb.pdf
Results in a **stochastic quasi-Newton** method.

**Summary:** Stochastic quasi-Newton integral:

\[
y_k = D_k \int_0^1 \tilde{B}(r_k(\tau))d\tau + e_k,
\]

with the following model for the Hessian

\[
\tilde{B}(\theta) \sim \text{GP}(\mu(\theta), \kappa(\theta, \theta')).
\]

Talk more about this on Thursday.
Message: The Gaussian process can be used to construct useful representations of dynamical systems.

I have hinted at how we can combine standard dynamical models like

1. Linear – impulse response
2. Nonlinear – Autoregressive (AR)
3. Nonlinear – State-space model (SSM)

and the Gaussian process to achieve useful constructions.

"If you have a new nonlinear construction, always make sure that it "does the right thing" in the linear Gaussian special case."
System identification

the Gaussian process

Remember to talk to people who work on different problems with different tools!!