ELLIS Summer School on Probabilistic Machine Learning

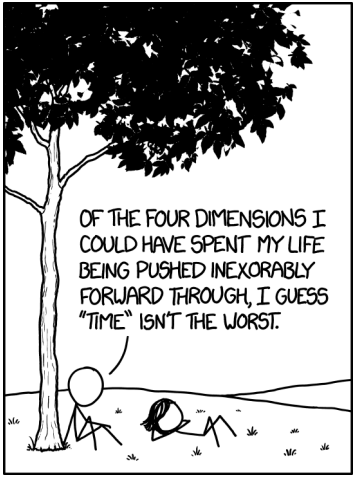
Sequential Inference and Learning

Arno Solin
ELLIS Scholar /
Assistant Professor in Machine Learning
Aalto University

July 2023 · Cambridge, UK

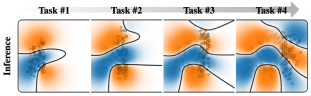
Aalto University

It’s all about the tools in your toolbox



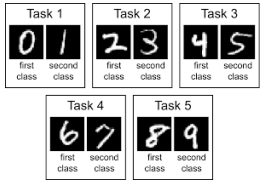
What is sequential machine learning?

Sequential inference / streaming



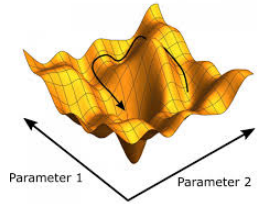
- **Timeseries**
- **Long/unbounded data**
- **Dynamical systems**
- Recurrent NNs *etc.*

Continual learning



- "Life-long learning"
- Non-stationarity
- Model keeps changing
- Data keep changing

Bayesian optimization / active learning



- How to pick points?
- Sequential decisions
- Connections to RL *etc.*

Goals

- Remind you of basic principles of direct links between signal processing and machine learning
- Provide an intuitive hands-on understanding of what stochastic differential equations are all about
- Show how these methods have real benefits in speeding up learning, improving inference, and model building

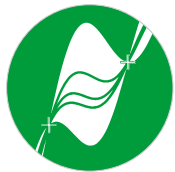
Structure

Part I

Tools from signal processing

Part II

SDEs (continuous-time models)



Part III

Temporal Gaussian processes

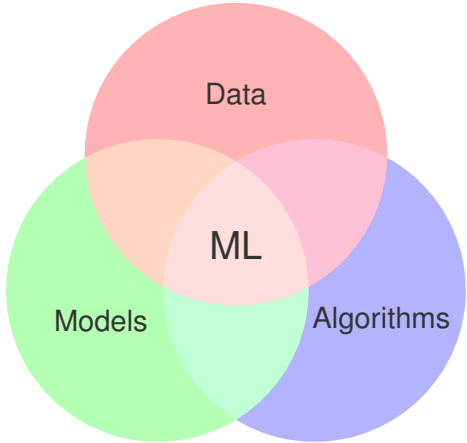
Part IV

Application examples

Motivation: Temporal models

- **One-dimensional problems**
  (the data has a natural ordering)
- **Spatio-temporal models**
  (something developing over time)
- **Long / unbounded data**
  (sensor data streams, daily observations, *etc.*)

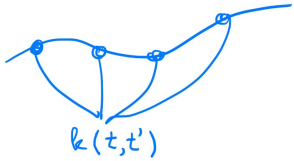
Machine learning



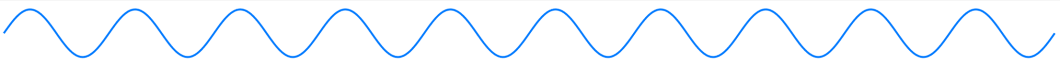
Where does the sequential nature show?

Tools for dealing with time-series

- **Moment representation**
  Considering the statistical properties of the input data jointly over time
- **Spectral (Fourier) representation**
  Analyzing the frequency-space representation of the problem/data
- **State space (path) representation**
  Description of sample behaviour as a dynamic system over time



Spectral (Fourier) representation



- Fourier transform $\mathcal{F}[\cdot]$:

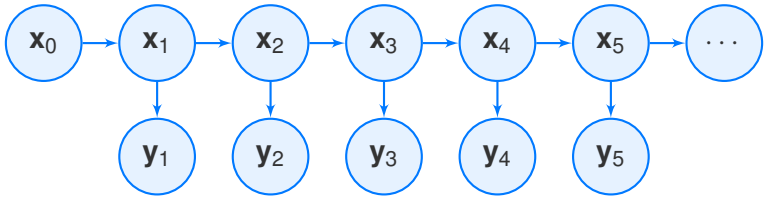
$$\tilde{f}(\omega) = \int f(\mathbf{x}) \, \exp(-\mathrm{i} \, \omega^{\mathsf{T}} \mathbf{x}) \, \mathrm{d}\mathbf{x}$$

- Analyzing properties of 'systems' (input–output mappings) by transfer functions:

$$H(s) = \frac{Y(s)}{X(s)} = \frac{\mathcal{L}[y(t)](s)}{\mathcal{L}[x(t)](s)},$$

  where $\mathcal{L}[\cdot]$ is the Laplace transform

Discrete-time state space models

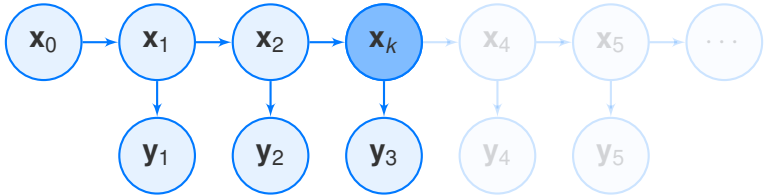


- A canonical state space model:

$$\begin{aligned} \text{Dynamics (prior):} \quad & \mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_k), \quad && \mathbf{q}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_k), \\ \text{Measurement (likelihood):} \quad & \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k), && \mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k) \end{aligned}$$

- The key to efficiency is the directed graph:
  The Markov property.

Kalman filtering and smoothing

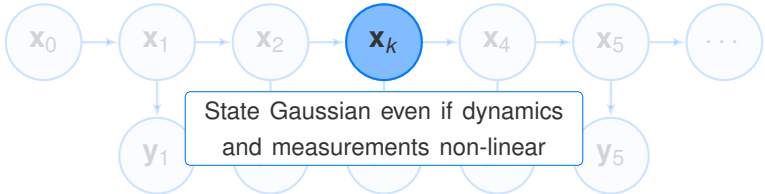


- Closed-form solution to linear-Gaussian filtering problems

$$\mathbf{x}_k = \mathbf{A}\,\mathbf{x}_{k-1} + \mathbf{q}_k, \qquad \mathbf{q}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_k),$$
$$\mathbf{y}_k = \mathbf{H}\,\mathbf{x}_k + \mathbf{r}_k, \qquad \mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$$

- Filtering solution: $p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_{k|k}, \mathbf{P}_{k|k})$
- Smoothing solution: $p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) = \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_{k|T}, \mathbf{P}_{k|T})$

Non-linear filtering



- Typically $\mathbf{x}_k$ is assumed Gaussian:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_k), \qquad \mathbf{q}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_k),$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_k), \qquad \mathbf{r}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$$

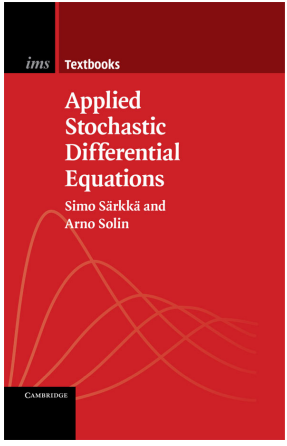
- Filtering solution: $p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \simeq \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_{k|k}, \mathbf{P}_{k|k})$
- Smoothing solution: $p(\mathbf{x}_k \mid \mathbf{y}_{1:T}) \simeq \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_{k|T}, \mathbf{P}_{k|T})$

Probabilistic inertial-visual odometry
for occlusion-robust navigation (`https://youtu.be/_ywmtVzxURk`)

SDEs (continuous-time models)

Everything is more _____ *in continuous time.*

This is also why we prefer modelling things in function space (think of GPs, analyzing NNs, *etc.*).



S. Särkkä and A. Solin (2019). Applied Stochastic Differential Equations. Cambridge University Press. Cambridge, UK.
Book PDF and codes for replicating examples available online.

Differential equations model how things change

- **Ordinary differential equations (ODEs)**
  (deterministic)
- **Stochastic differential equations (SDEs)**
  (stochastic)

What is a stochastic differential equation (SDE)?

- Consider an ordinary differential equation (ODE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t)$$

- Then we add white noise to the right hand side:

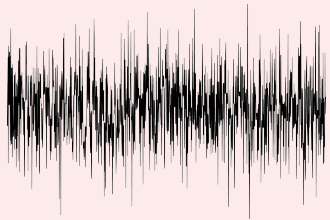
$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\,\mathbf{w}(t)$$

- $\mathbf{f}(\mathbf{x}, t)$ is the drift function and $\mathbf{L}(\mathbf{x}, t)$ is the dispersion matrix (diffusion term)
- Now we have a stochastic differential equation (SDE)

White noise

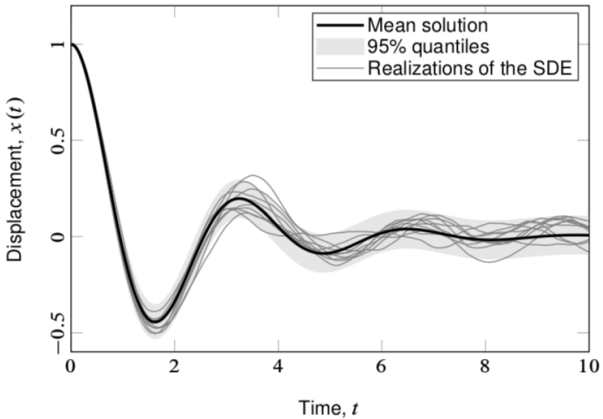
1. $\mathbf{w}(t_1)$ and $\mathbf{w}(t_2)$ are independent if $t_1 \neq t_2$
2. $t \mapsto \mathbf{w}(t)$ is a Gaussian process with mean and covariance:

$$\mathbb{E}[\mathbf{w}(t)] = \mathbf{0},$$
$$\mathbb{E}[\mathbf{w}(t)\,\mathbf{w}^\top(s)] = \delta(t-s)\,\mathbf{Q}$$



- $\mathbf{Q}$ is the spectral density of the process
- The sample path $t \mapsto \mathbf{w}(t)$ is discontinuous almost everywhere
- White noise is unbounded and it takes arbitrarily large positive and negative values at any finite interval

What does a solution of an SDE look like?



Solution paths of a stochastic spring model

$$\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} + \gamma \frac{\mathrm{d}x(t)}{\mathrm{d}t} + \nu^2 x(t) = w(t)$$

SDEs as white noise–driven differential equations

- Treating SDEs as white noise–driven differential equations has its limits

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\,\mathbf{w}(t)$$

- For linear equations the approach works
- But this interpretation breaks down in the general setting:
  - The chain rule of calculus starts giving wrong answers!
  - With non-linear differential equations the behaviour becomes unexpected
  - Trying to prove the existence of solutions becomes tricky
- The source of the problems is the everywhere discontinuous white noise $\mathbf{w}(t)$
- So how should we really formulate SDEs?

Equivalent integral equation

- We have a differential equation of the form

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\,\mathbf{w}(t)$$

- Integrating the differential equation from t_0 to t gives:

$$\mathbf{x}(t) - \mathbf{x}(t_0) = \int_{t_0}^{t} \mathbf{f}(\mathbf{x}(t), t)\,\mathrm{d}t + \int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\,\mathbf{w}(t)\,\mathrm{d}t$$

- The first integral is just a Riemann/Lebesgue integral
- The second integral is the problematic one due to the white noise (this is the interesting part!)

Attempt 1: Riemann integral

- In the Riemannian sense the integral would be defined as

$$\int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\, \mathbf{w}(t)\, \mathrm{d}t = \lim_{n \to \infty} \sum_{k} \mathbf{L}(\mathbf{x}(t_k^*), t_k^*)\, \mathbf{w}(t_k^*)\, (t_{k+1} - t_k),$$

  where $t_0 < t_1 < \ldots < t_n = t$ and $t_k^* \in [t_k, t_{k+1}]$
- Upper and lower sums are defined as the selections of t_k^* such that the integrand $\mathbf{L}(\mathbf{x}(t_k^*), t_k^*)\, \mathbf{w}(t_k^*)$ has its maximum and minimum values, respectively
- The Riemann integral exists if the upper and lower sums converge to the same value

! Because white noise is discontinuous everywhere, the Riemann integral does not exist

Attempt 2: Stieltjes integral [1/2]

- A Stieltjes integral is more general and allows for discontinuous integrands
- We can interpret the increment $\mathbf{w}(t)\,\mathrm{d}t$ as increments of another process $\beta(t)$ such that

$$\int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\,\mathbf{w}(t)\,\mathrm{d}t = \int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\,\mathrm{d}\beta(t).$$

- It turns out that a suitable process for this purpose is Brownian motion...

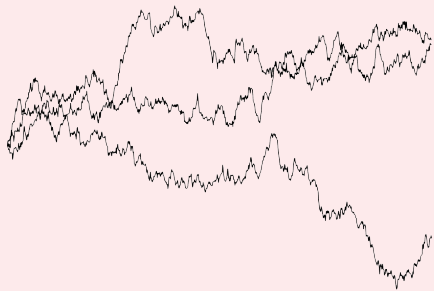
Brownian motion

1. Gaussian increments:

$$\Delta\beta_k \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}\,\Delta t_k),$$

   where $\Delta\beta_k = \beta(t_{k+1}) - \beta(t_k)$ and $\Delta t_k = t_{k+1} - t_k$

2. Non-overlapping increments are independent



- $\mathbf{Q}$ is the diffusion matrix of the Brownian motion.
- Brownian motion $t \mapsto \beta(t)$ has discontinuous derivative everywhere
- White noise can be considered the formal derivative of Brownian motion $\mathbf{w}(t) = \mathrm{d}\beta(t)/\,\mathrm{d}t$

Attempt 2: Stieltjes integral [2/2]

- Stieltjes integral is defined as a limit of the form

$$\int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\, \mathrm{d}\boldsymbol{\beta} = \lim_{n\to\infty} \sum_{k} \mathbf{L}(\mathbf{x}(t_k^*), t_k^*)\, [\boldsymbol{\beta}(t_{k+1}) - \boldsymbol{\beta}(t_k)],$$

  where $t_0 < t_1 < \ldots < t_n$ and $t_k^* \in [t_k, t_{k+1}]$
- The limit t_k^* should be independent of the position on the interval $t_k^* \in [t_k, t_{k+1}]$
- For integration with respect to Brownian motion this is not the case

! Thus, the Stieltjes integral definition does not work either

Attempt 3: Lebesgue integral

- In a Lebesgue integral we could interpret $\beta(t)$ to define a 'stochastic measure'
- Essentially, this will also lead to the definition

$$\int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\, \mathrm{d}\beta = \lim_{n\to\infty} \sum_{k} \mathbf{L}(\mathbf{x}(t_k^*), t_k^*)\, [\beta(t_{k+1}) - \beta(t_k)],$$

  where $t_0 < t_1 < \ldots < t_n$ and $t_k^* \in [t_k, t_{k+1}]$.
- Again, the limit should be independent of the choice $t_k^* \in [t_k, t_{k+1}]$
- Also our 'measure' is not really a sensible measure

! The Lebesgue integral does not work either

Attempt 4: Itô integral

- The solution to the problem is the Itô stochastic integral
- The idea is to fix the choice to $t_k^* = t_k$, and define the integral as

$$\int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\, \mathrm{d}\beta(t) = \lim_{n\to\infty} \sum_k \mathbf{L}(\mathbf{x}(t_k), t_k)\, [\beta(t_{k+1}) - \beta(t_k)]$$

- This Itô stochastic integral turns out to be a sensible definition of the integral
- However, the resulting integral does not obey the computational rules of ordinary calculus
- Instead of ordinary calculus we have Itô calculus

Itô stochastic differential equations

- Consider the white noise–driven ODE

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\,\mathbf{w}(t)$$

- This is actually defined as the Itô integral equation

$$\mathbf{x}(t) - \mathbf{x}(t_0) = \int_{t_0}^{t} \mathbf{f}(\mathbf{x}(t), t)\,\mathrm{d}t + \int_{t_0}^{t} \mathbf{L}(\mathbf{x}(t), t)\,\mathrm{d}\boldsymbol{\beta}(t),$$

  which should be true for arbitrary t_0 and t
- Which can be written (considering the limits 'small') as

$$\boxed{\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\,\mathrm{d}t + \mathbf{L}(\mathbf{x}, t)\,\mathrm{d}\boldsymbol{\beta}}$$

- This is the canonical form of an Itô SDE

Connection with white noise–driven ODEs

- Let's formally divide by $\mathrm{d}t$, which gives

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) + \mathbf{L}(\mathbf{x}, t)\,\frac{\mathrm{d}\boldsymbol{\beta}}{\mathrm{d}t}$$

- Thus we can interpret $\mathrm{d}\boldsymbol{\beta}/\,\mathrm{d}t$ as white noise $\mathbf{w}$ (not an entity as such, only the formal derivative)
- Note that we cannot define more general equations

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(t), \mathbf{w}(t), t),$$

because we cannot re-interpret this as an Itô integral equation

Non-linear SDEs

- There is no general solution method for non-linear SDEs

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\,\mathrm{d}t + \mathbf{L}(\mathbf{x}, t)\,\mathrm{d}\boldsymbol{\beta}$$

- However, numerical simulation of solution trajectories is usually possible (*e.g.*, with stochastic Runge–Kutta)
- The simplest alternative is the Euler–Maruyama method:

$$\hat{\mathbf{x}}(t_{k+1}) = \hat{\mathbf{x}}(t_k) + \mathbf{f}(\hat{\mathbf{x}}(t_k), t_k)\,\Delta t + \mathbf{L}(\hat{\mathbf{x}}(t_k), t_k)\,\Delta\boldsymbol{\beta}_k,$$

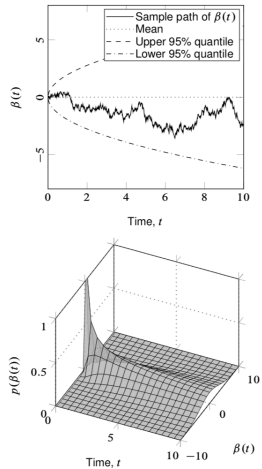
where $\Delta\boldsymbol{\beta}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}\,\Delta t)$

Solution concepts in SDEs

- Path of a Brownian motion which is solution to stochastic differential equation

$$\frac{\mathrm{d}x}{\mathrm{d}t} = w(t)$$

- Strong vs. weak solutions
- Evolution of the probability density of the solution trajectories is given by the Fokker–Planck–Kolmogorov PDE



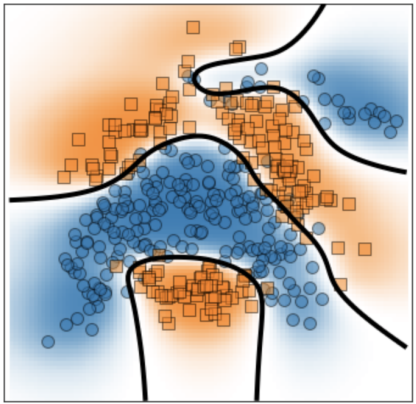
Summary

- Stochastic differential equations (SDE) can be seen as differential equations with a stochastic driving force
- SDEs are typical in physics, engineering, and finance applications
- A heuristic white noise formulation has problems with the chain rule, non-linearities, and solution existence
- Instead, use the Itô stochastic integral (calculus)
- Various solution concepts; in general, non-linear SDEs are tricky to solve (good schemes for simulation exist though)

Temporal Gaussian processes

Gaussian processes

- GPs are powerful tools for model specification and inference.
- Meaningful uncertainty estimates and a direct way to include *prior* knowledge.
- Do not require *ad hoc* tinkering (plug & play).



GP classification with a Bernoulli likelihood

GPs are associated with limitations

Scaling to large data

A naïve solution to dealing with the expanded Gram (covariance) matrix requires $\mathcal{O}(n^3)$ compute and $\mathcal{O}(n^2)$ memory. Infeasible for $n > 10{,}000$.

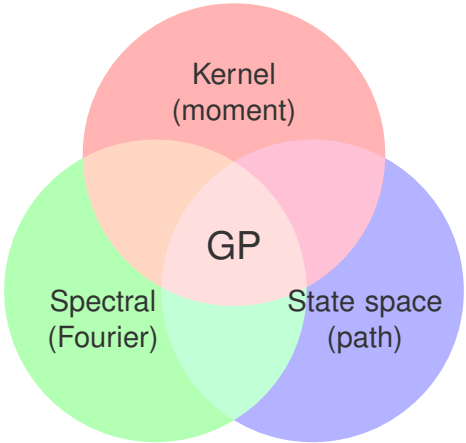
Dealing with non-conjugate likelihoods

For a Gaussian observation model the GP posterior is available in closed-form. For non-conjugate likelihood models one has to resort to approximate inference methods.

Representational power

Gaussian processes are ideal for problems where it is easy to specify *meaningful* priors. For applications such as image classification this is hard.

Three views into (stationary) GPs



Kernel (moment) representation

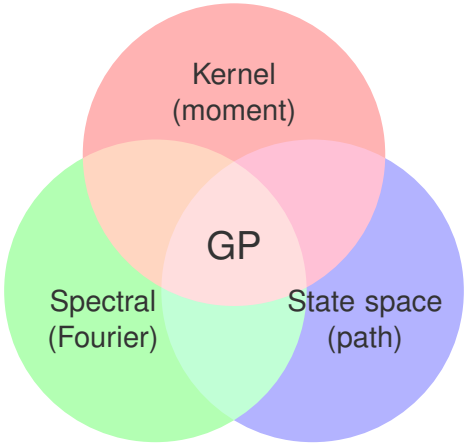
$$f(t) \sim \mathrm{GP}(\mu(t), \kappa(t, t')) \qquad \textit{GP prior}$$
$$\mathbf{y} \mid \mathbf{f} \sim \prod_i p(y_i \mid f(t_i)) \qquad \textit{likelihood}$$

- Let's focus on the GP prior only.
- A temporal Gaussian process (GP) is a random function $f(t)$, such that joint distribution of $f(t_1), \ldots, f(t_n)$ is always Gaussian.
- Mean and covariance functions have the form:

$$\mu(t) = \mathbb{E}[f(t)],$$
$$\kappa(t, t') = \mathbb{E}[(f(t) - \mu(t))(f(t') - \mu(t'))^{\mathsf{T}}].$$

- Convenient for model specification, but expanding the kernel to a covariance matrix can be problematic (the notorious $\mathcal{O}(n^3)$ scaling).

Three views into (stationary) GPs



Spectral (Fourier) representation

- The Fourier transform of a function $f(t) : \mathbb{R} \to \mathbb{R}$ is

$$\mathcal{F}[f](\mathrm{i}\,\omega) = \int_{\mathbb{R}} f(t)\ \exp(-\mathrm{i}\,\omega\, t)\,\mathrm{d}t$$

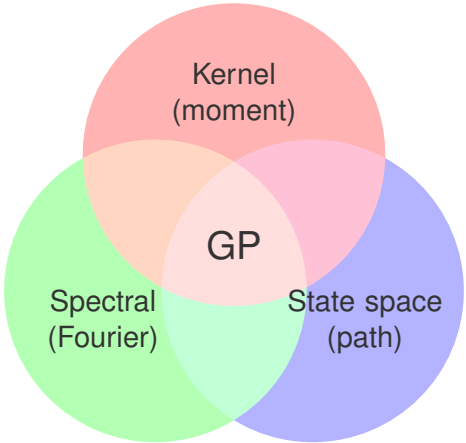
- For a stationary GP, the covariance function can be written in terms of the difference between two inputs:

$$\kappa(t, t') \triangleq \kappa(t - t')$$

- Wiener–Khinchin: If $f(t)$ is a stationary Gaussian process with covariance function $\kappa(t)$ then its spectral density is $S(\omega) = \mathcal{F}[\kappa]$.
- Spectral representation of a GP in terms of spectral density function

$$S(\omega) = \mathbb{E}[\tilde{f}(\mathrm{i}\,\omega)\,\tilde{f}^{\mathsf{T}}(-\mathrm{i}\,\omega)]$$

Three views into (stationary) GPs



State space (path) representation [1/3]

- Path or state space representation as solution to a linear time-invariant (LTI) stochastic differential equation (SDE):

$$\mathrm{d}\mathbf{f} = \mathbf{F}\,\mathbf{f}\,\mathrm{d}t + \mathbf{L}\,\mathrm{d}\boldsymbol{\beta},$$

  where $\mathbf{f} = (f, \mathrm{d}f/\mathrm{d}t, \ldots)$ and $\boldsymbol{\beta}(t)$ is a vector of Wiener processes.
- Equivalently, but more informally

$$\frac{\mathrm{d}\mathbf{f}(t)}{\mathrm{d}t} = \mathbf{F}\,\mathbf{f}(t) + \mathbf{L}\,\mathbf{w}(t),$$

  where $\mathbf{w}(t)$ is white noise.
- The model now consists of a drift matrix $\mathbf{F} \in \mathbb{R}^{m \times m}$, a diffusion matrix $\mathbf{L} \in \mathbb{R}^{m \times s}$, and the spectral density matrix of the white noise process $\mathbf{Q}_c \in \mathbb{R}^{s \times s}$.
- The scalar-valued GP can be recovered by $f(t) = \mathbf{H}\,\mathbf{f}(t)$.

State space (path) representation [2/3]

- The initial state is given by a stationary state $\mathbf{f}(0) \sim \mathrm{N}(\mathbf{0}, \mathbf{P}_\infty)$ which fulfills

$$\mathbf{F}\,\mathbf{P}_\infty + \mathbf{P}_\infty\,\mathbf{F}^\mathsf{T} + \mathbf{L}\,\mathbf{Q}_\mathrm{c}\,\mathbf{L}^\mathsf{T} = \mathbf{0}$$

- The covariance function at the stationary state can be recovered by

$$\kappa(t, t') = \begin{cases} \mathbf{P}_\infty\,\exp((t'-t)\mathbf{F})^\mathsf{T}, & t' \geq t \\ \exp((t'-t)\mathbf{F})\,\mathbf{P}_\infty & t' < t \end{cases}$$

  where $\exp(\cdot)$ denotes the matrix exponential function.

- The spectral density function at the stationary state can be recovered by

$$S(\omega) = (\mathbf{F} + \mathrm{i}\,\omega\,\mathbf{I})^{-1}\,\mathbf{L}\,\mathbf{Q}_\mathrm{c}\,\mathbf{L}^\mathsf{T}\,(\mathbf{F} - \mathrm{i}\,\omega\,\mathbf{I})^{-\mathsf{T}}$$

State space (path) representation [3/3]

- Similarly as the kernel has to be evaluated into covariance matrix for computations, the SDE can be solved for discrete time points $\{t_i\}_{i=1}^n$.
- The resulting model is a discrete state space model:

$$\mathbf{f}_i = \mathbf{A}_{i-1}\,\mathbf{f}_{i-1} + \mathbf{q}_{i-1}, \quad \mathbf{q}_i \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_i),$$

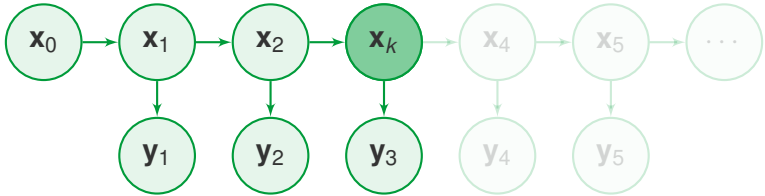
  where $\mathbf{f}_i = \mathbf{f}(t_i)$.
- The discrete-time model matrices are given by:

$$\mathbf{A}_i = \exp(\mathbf{F}\,\Delta t_i),$$
$$\mathbf{Q}_i = \int_0^{\Delta t_i} \exp(\mathbf{F}\,(\Delta t_i - \tau))\,\mathbf{L}\,\mathbf{Q}_\mathrm{c}\,\mathbf{L}^\mathsf{T}\,\exp(\mathbf{F}\,(\Delta t_i - \tau))^\mathsf{T}\,\mathrm{d}\tau,$$

  where $\Delta t_i = t_{i+1} - t_i$
- If the model is stationary, $\mathbf{Q}_i$ is given by

$$\mathbf{Q}_i = \mathbf{P}_\infty - \mathbf{A}_i\,\mathbf{P}_\infty\,\mathbf{A}_i^\mathsf{T}$$

Kalman filtering and smoothing

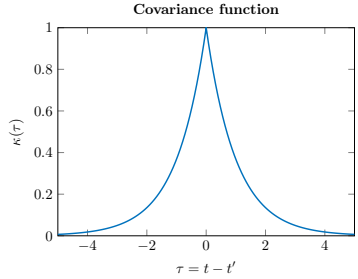


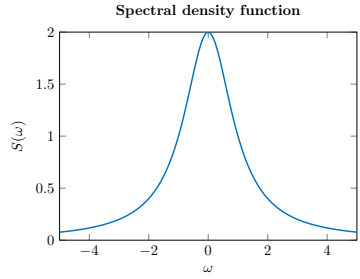
- Closed-form solution to linear-Gaussian filtering problems

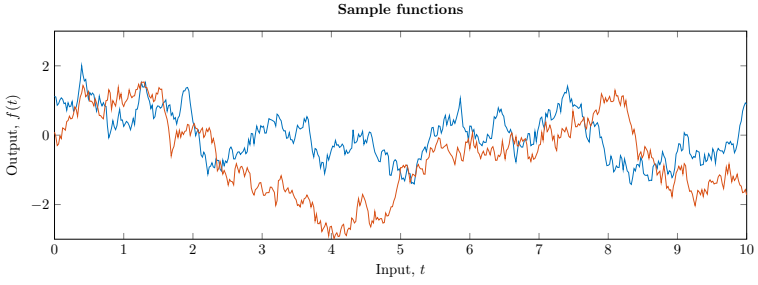
$$\mathbf{x}_i = \mathbf{A}_{i-1}\,\mathbf{x}_{i-1} + \mathbf{q}_{i-1}, \qquad \mathbf{q}_i \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_i),$$
$$\mathbf{y}_i = \mathbf{H}\,\mathbf{x}_i + \mathbf{r}_i, \qquad \mathbf{r}_i \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_i)$$

- Filtering solution: $p(\mathbf{x}_i \mid \mathbf{y}_{1:i}) = \mathrm{N}(\mathbf{x}_i \mid \mathbf{m}_{i|i}, \mathbf{P}_{i|i})$
- Smoothing solution: $p(\mathbf{x}_i \mid \mathbf{y}_{1:T}) = \mathrm{N}(\mathbf{x}_i \mid \mathbf{m}_{i|T}, \mathbf{P}_{i|T})$

Three views into GPs







Example: Exponential covariance function

- Exponential covariance function (Ornstein-Uhlenbeck process):

$$\kappa(t, t') = \exp(-\lambda |t - t'|)$$

- Spectral density function:

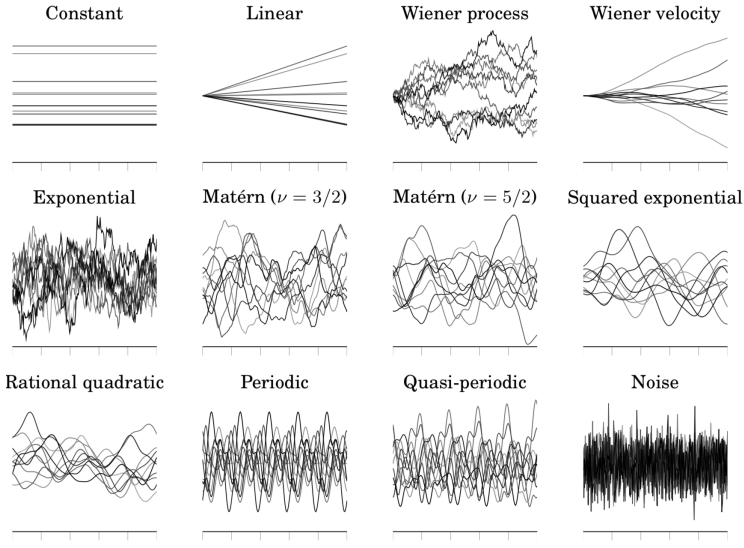
$$S(\omega) = \frac{2}{\lambda + \omega^2/\lambda}$$

- Path representation: Stochastic differential equation (SDE)

$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} = -\lambda f(t) + w(t),$$

or using the notation from before:
$F = -\lambda$, $L = 1$, $Q_c = 2$, $H = 1$, and $P_\infty = 1$.

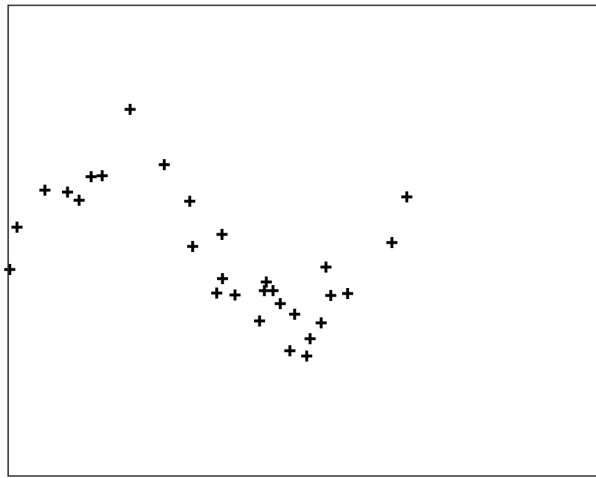
Examples of applicable GP priors



Applicable GP priors

- The covariance function needs to be Markovian (or approximated as such).
- Covers many common stationary and non-stationary models.
- Sums of kernels: $\kappa(t, t') = \kappa_1(t, t') + \kappa_2(t, t')$
  - Stacking of the state spaces
  - State dimension: $m = m_1 + m_2$
- Product of kernels: $\kappa(t, t') = \kappa_1(t, t')\,\kappa_2(t, t')$
  - Kronecker sum of the models
  - State dimension: $m = m_1\, m_2$

Example: GP regression, $\mathcal{O}(n^3)$



The input–output pairs

Example: GP regression, $\mathcal{O}(n^3)$

- Consider the GP regression problem with input–output training pairs $\{(t_i, y_i)\}_{i=1}^n$:

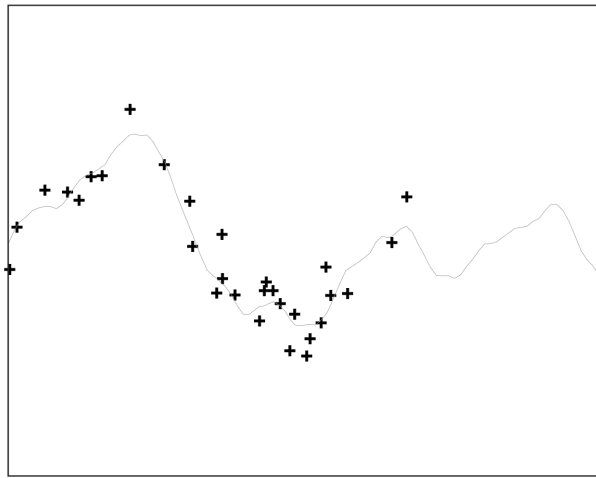
$$f(t) \sim \mathrm{GP}(0, \kappa(t, t')),$$
$$y_i = f(t_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathrm{N}(0, \sigma_\mathrm{n}^2)$$

- The posterior mean and variance for an unseen test input t_* is given by (see previous lectures):

$$\mathbb{E}[f_*] = \mathbf{k}_* \left(\mathbf{K} + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \mathbf{y},$$
$$\mathbb{V}[f_*] = \kappa(t_*, t_*) - \mathbf{k}_* \left(\mathbf{K} + \sigma_\mathrm{n}^2 \mathbf{I}\right)^{-1} \mathbf{k}_*^\mathsf{T}$$

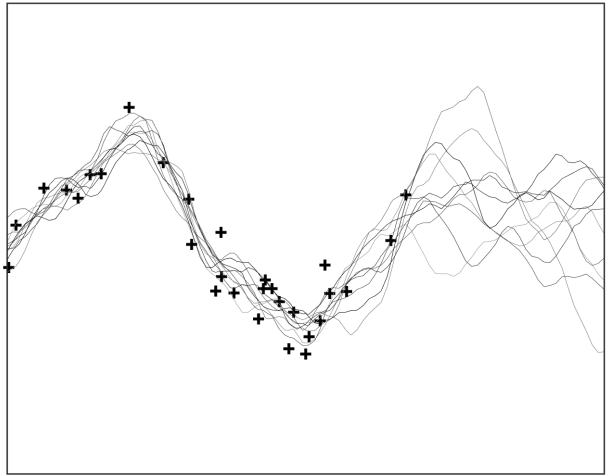
- Note the inversion of the $n \times n$ matrix.

Example: GP regression, $\mathcal{O}(n^3)$



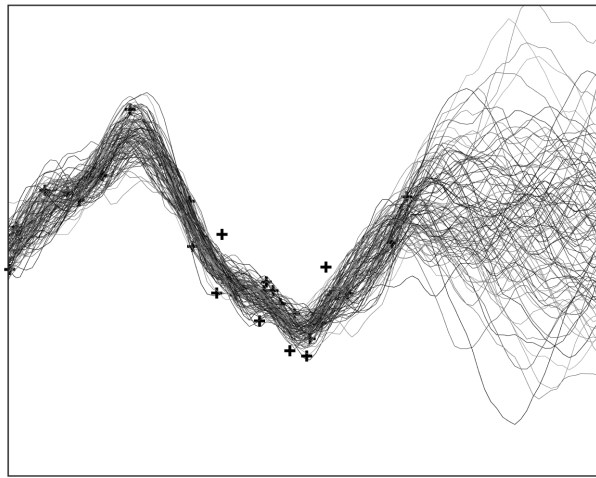
Draw from the GP posterior with a Matérn prior

Example: GP regression, $\mathcal{O}(n^3)$



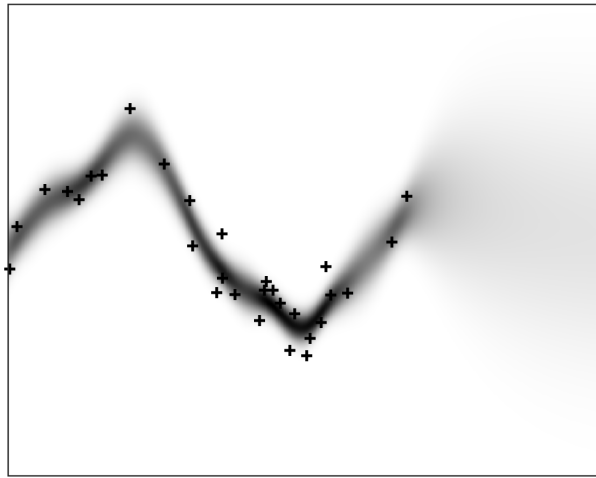
Draws from the GP posterior

Example: GP regression, $\mathcal{O}(n^3)$



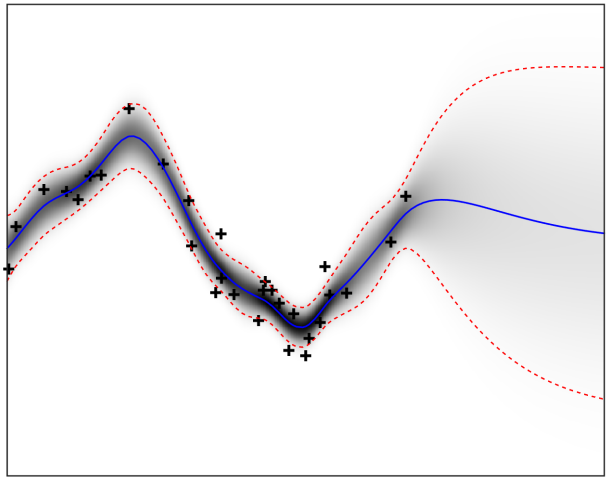
Draws from the GP posterior

Example: GP regression, $\mathcal{O}(n^3)$



The GP posterior marginals

Example: GP regression, $\mathcal{O}(n^3)$



The GP posterior marginals

Example: GP regression, $\mathcal{O}(n)$

- The sequential solution (goes under the name 'Kalman filter') considers one data point at a time, hence the linear time-scaling.
- Start from $\mathbf{m}_0 = \mathbf{0}$ and $\mathbf{P}_0 = \mathbf{P}_\infty$ and for each data point iterate the following steps.
- Kalman prediction:

$$\begin{aligned}\mathbf{m}_{i|i-1} &= \mathbf{A}_{i-1}\,\mathbf{m}_{i-1|i-1},\\ \mathbf{P}_{i|i-1} &= \mathbf{A}_{i-1}\,\mathbf{P}_{i-1|i-1}\,\mathbf{A}_{i-1}^{\mathsf{T}} + \mathbf{Q}_{i-1}.\end{aligned}$$

- Kalman update:

$$\begin{aligned}\mathbf{v}_i &= y_i - \mathbf{H}\,\mathbf{m}_{i|i-1},\\ \mathbf{S}_i &= \mathbf{H}_i\,\mathbf{P}_{i|i-1}\,\mathbf{H}^{\mathsf{T}} + \sigma_{\mathrm{n}}^2,\\ \mathbf{K}_i &= \mathbf{P}_{i|i-1}\,\mathbf{H}^{\mathsf{T}}\,\mathbf{S}_i^{-1},\\ \mathbf{m}_{i|i} &= \mathbf{m}_{i|i-1} + \mathbf{K}_i\,\mathbf{v}_i,\\ \mathbf{P}_{i|i} &= \mathbf{P}_{i|i-1} - \mathbf{K}_i\,\mathbf{S}_i\,\mathbf{K}_i^{\mathsf{T}}.\end{aligned}$$

Example: GP regression, $\mathcal{O}(n)$

- To condition all time-marginals on all data, run a backward sweep (Rauch–Tung–Striebel smoother):

$$\begin{aligned}
\mathbf{m}_{i+1|i} &= \mathbf{A}_i\, \mathbf{m}_{i|i}, \\
\mathbf{P}_{i+1|i} &= \mathbf{A}_i\, \mathbf{P}_{i|i}\, \mathbf{A}_i^\mathsf{T} + \mathbf{Q}_i, \\
\mathbf{G}_i &= \mathbf{P}_{i|i}\, \mathbf{A}_i^\mathsf{T}\, \mathbf{P}_{i+1|i}^{-1}, \\
\mathbf{m}_{i|n} &= \mathbf{m}_{i|i} + \mathbf{G}_i\, (\mathbf{m}_{i+1|n} - \mathbf{m}_{i+1|i}), \\
\mathbf{P}_{i|n} &= \mathbf{P}_{i|i} + \mathbf{G}_i\, (\mathbf{P}_{i+1|n} - \mathbf{P}_{i+1|i})\, \mathbf{G}_i^\mathsf{T},
\end{aligned}$$

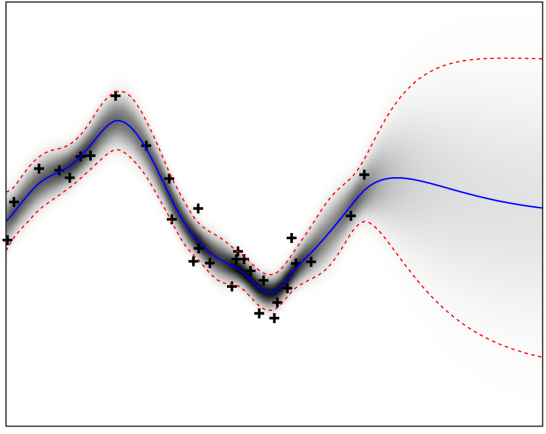
- The marginal mean and variance can be recovered by:

$$\mathbb{E}[f_i] = \mathbf{H}\, \mathbf{m}_{i|n} \quad \text{and} \quad \mathbb{V}[f_i] = \mathbf{H}\, \mathbf{P}_{i|n}\, \mathbf{H}^\mathsf{T}$$

- The log marginal likelihood evaluated as a by-product of the Kalman update:

$$\log p(\mathbf{y}) = -\frac{1}{2} \sum_{i=1}^{n} \log |2\pi\, \mathbf{S}_i| + \mathbf{v}_i^\mathsf{T}\, \mathbf{S}_i^{-1} \mathbf{v}_i$$

Example: GP regression, $\mathcal{O}(n)$



The state space representation enables efficient inference through Kalman filtering

Example: Births in the US

- Number of births in the US
- Daily data between 1969–1988 ($n = 7305$)
- GP regression with a prior covariance function:

$$\begin{aligned}\kappa(t, t') = \kappa_{\text{Mat.}}^{\nu=5/2}(t, t') + \kappa_{\text{Mat.}}^{\nu=3/2}(t, t') \\ + \kappa_{\text{Per.}}^{\text{year}}(t, t')\, \kappa_{\text{Mat.}}^{\nu=3/2}(t, t') + \kappa_{\text{Per.}}^{\text{week}}(t, t')\, \kappa_{\text{Mat.}}^{\nu=3/2}(t, t')\end{aligned}$$

- Learn hyperparameters by optimizing the marginal likelihood

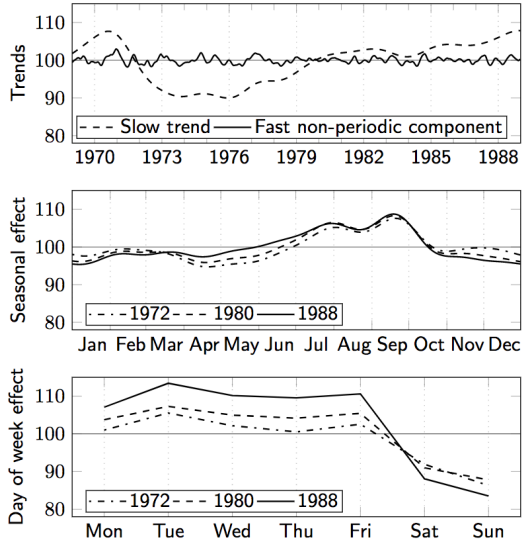
- Number of birth
- Daily data betw
- GP regression

$\kappa(t, t')$

$k_{\nu=3/2}^{\text{Mat.}}(t, t')$

- Learn hyperpar

od



Explaining changes in number of births in the US

Example: Aircraft accidents

- Commercial aircraft accidents 1919–2017
- Log-Gaussian Cox process (Poisson likelihood) by ADF/EP
- Daily binning, $n = 35{,}959$
- GP prior with a covariance function:

$$\kappa(t,t') = \kappa_{\text{Mat.}}^{\nu=3/2}(t,t') + \kappa_{\text{Per.}}^{\text{year}}(t,t')\,\kappa_{\text{Mat.}}^{\nu=3/2}(t,t') + \kappa_{\text{Per.}}^{\text{week}}(t,t')\,\kappa_{\text{Mat.}}^{\nu=3/2}(t,t')$$

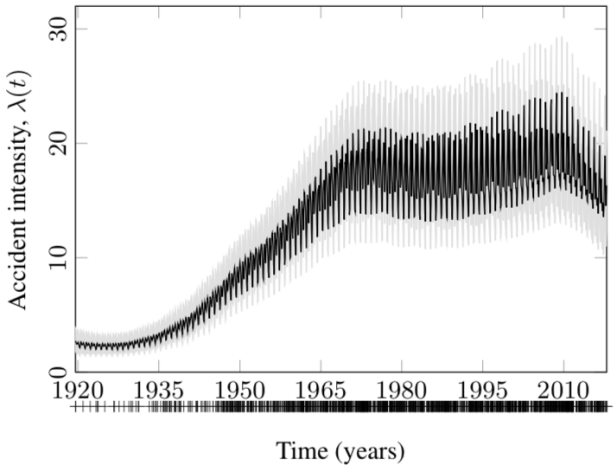
- Learn hyperparameters by optimizing the marginal likelihood

Example: Aircraft accidents

- Commercia
- Log-Gauss
- Daily binnir
- GP prior wi

κ $/2(t, t')$

- Learn hype



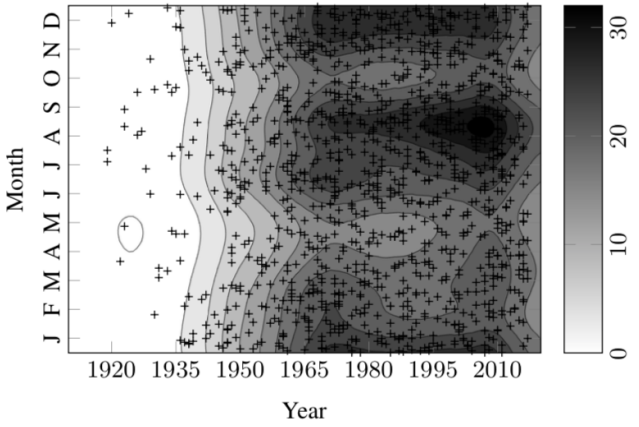
Example: Aircraft accidents

- Commercia
- Log-Gauss
- Daily binnir
- GP prior wi

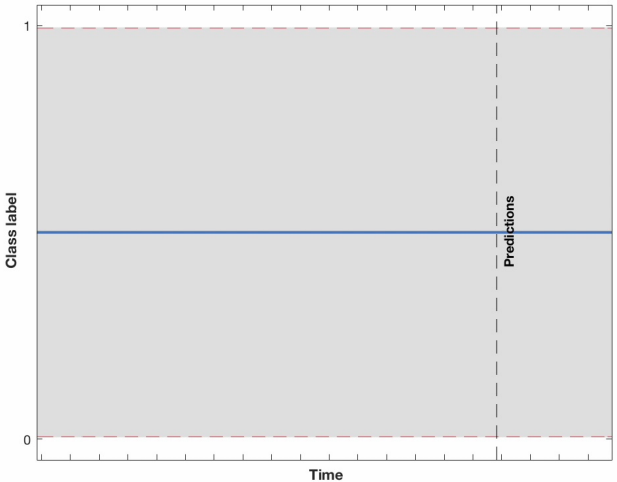
κ

$/2(t, t')$

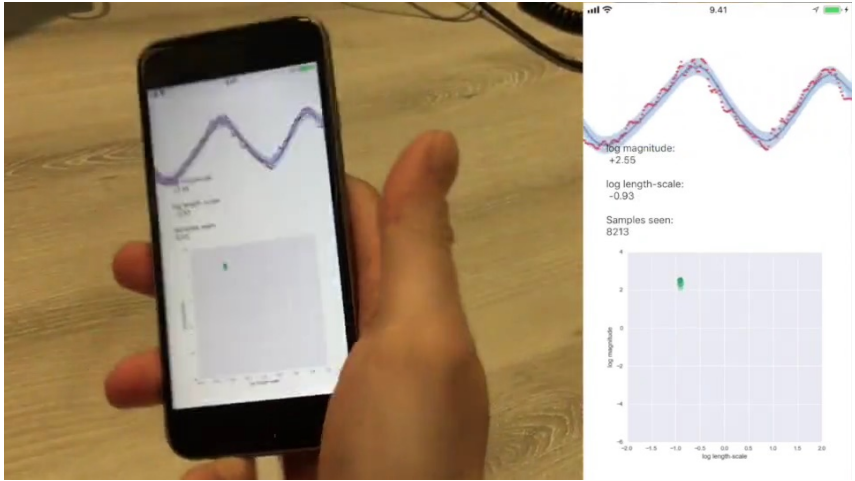
- Learn hype



What if the data really is infinite?



On-line inference by infinite time-horizon GPs



https://youtu.be/myCvUT3XGPc

Spatio-temporal GPs

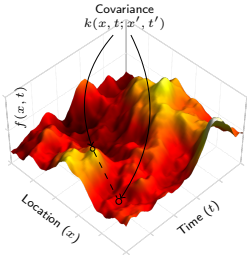
$$f(\mathbf{x}) \sim \mathrm{GP}(0, \kappa(\mathbf{x}, \mathbf{x}'))$$
$$\mathbf{y} \mid \mathbf{f} \sim \prod_i p(y_i \mid f(\mathbf{x}_i))$$

$$f(\mathbf{r}, t) \sim \mathrm{GP}(0, \kappa(\mathbf{r}, t; \mathbf{r}', t'))$$
$$\mathbf{y} \mid \mathbf{f} \sim \prod_i p(y_i \mid f(\mathbf{r}_i, t_i))$$

Spatio-temporal Gaussian processes

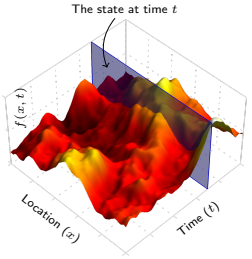
GPs under the kernel formalism

$$f(\mathbf{x}, t) \sim \mathrm{GP}(0, \kappa(\mathbf{x}, t; \mathbf{x}', t'))$$
$$y_i = f(\mathbf{x}_i, t_i) + \varepsilon_i$$

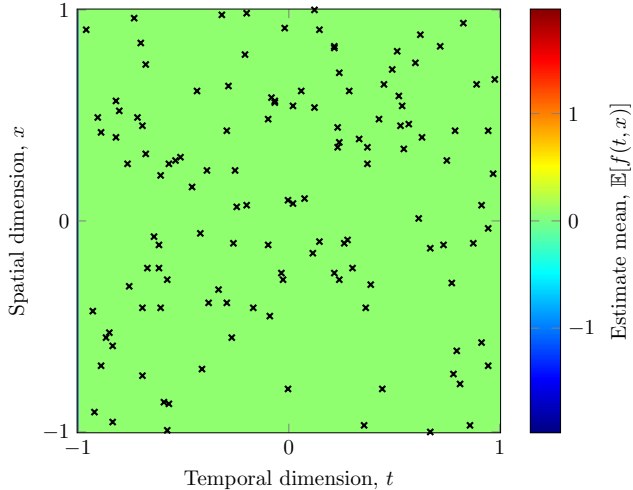


Stochastic partial differential equation formalism

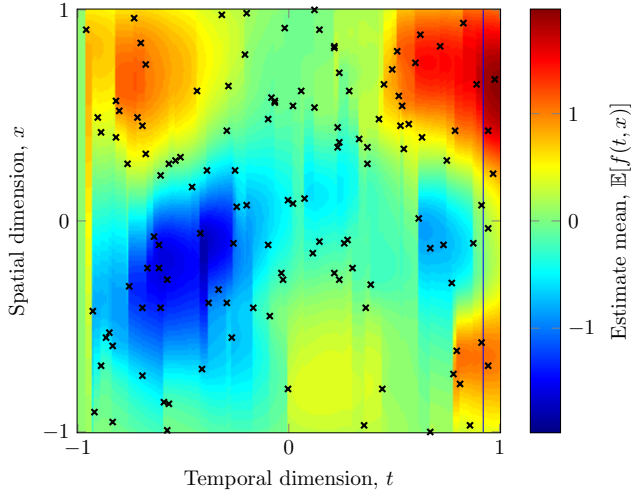
$$\frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial t} = \mathcal{F}\,\mathbf{f}(\mathbf{x}, t) + \mathcal{L}\,w(\mathbf{x}, t)$$
$$y_i = \mathcal{H}_i\,\mathbf{f}(\mathbf{x}, t) + \varepsilon_i$$



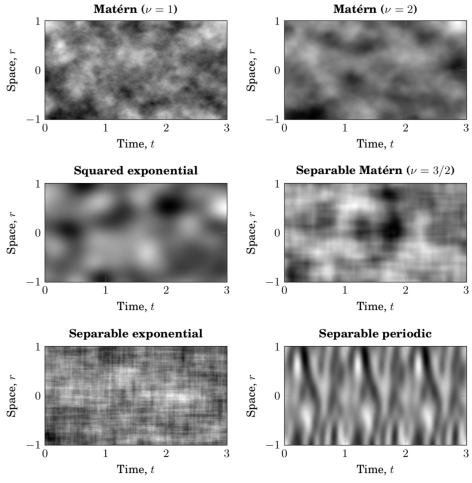
Spatio-temporal GP regression



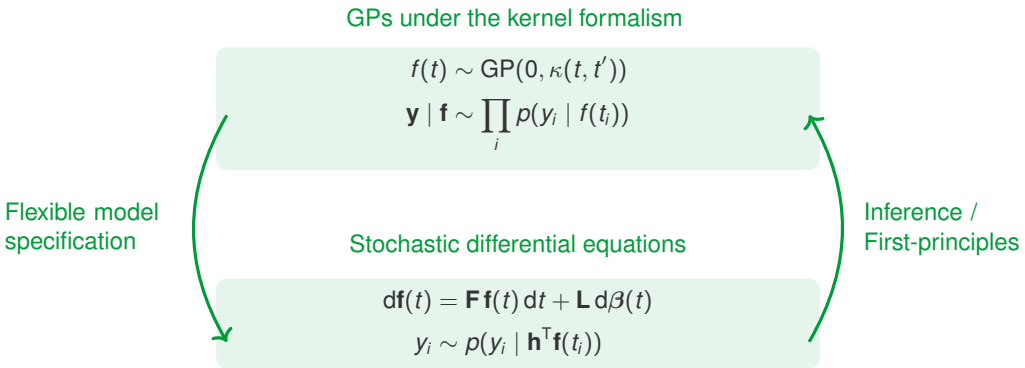
Spatio-temporal GP regression



Spatio-temporal GP priors



Gaussian processes ❤ SDEs



Different representations of GPs

- Gaussian processes have different representations:
  • Covariance function • Spectral density • State space
- Temporal (single-input) Gaussian processes
  $\iff$ stochastic differential equations (SDEs)
- Conversions between the representations can make model building easier
- (Exact) inference of the latent functions, can be done in $\mathcal{O}(n)$ time and memory complexity by Kalman filtering

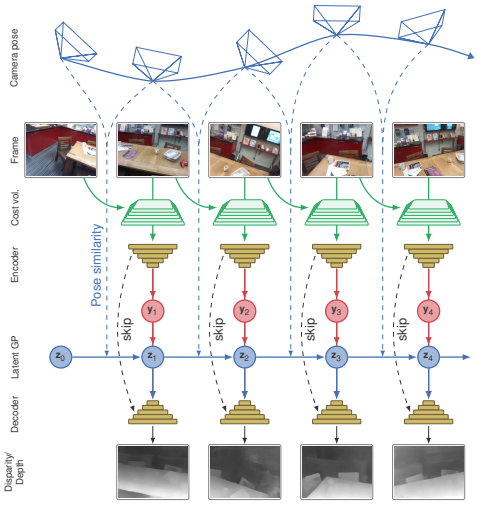
Application examples: Structure from temporal data

Monocular depth estimation

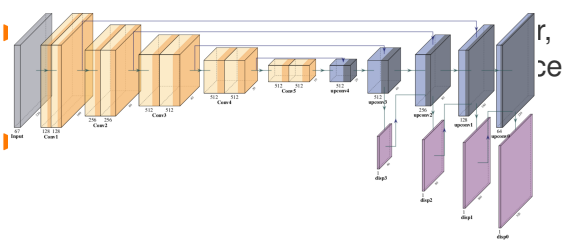
- Eyes on opposite sides:
  Large field-of-view vs. no stereo vision
- Monocular depth-sensing by head wobbling
- ‘Multi-view stereo’ (MVS) in computer vision
- “Structure from temporal data”

Source: Rooster Portrait on Wikimedia Commons

Priors in larger models

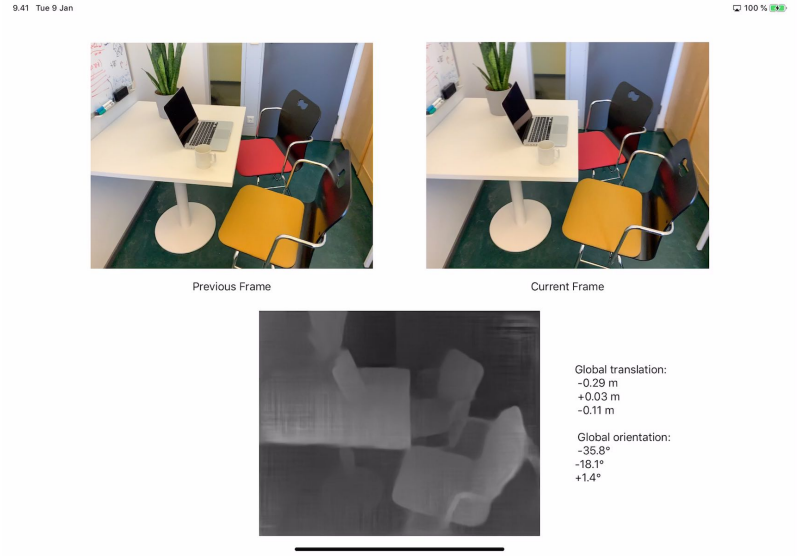


- Inputs: Frame pairs and relative camera poses
- State-of-the-art in CV: Encoder–decoder network for depth estimation



Hou *et al.* (2019). *Multi-view stereo by temporal nonparametric fusion.* ICCV.

Online inference on an iPad

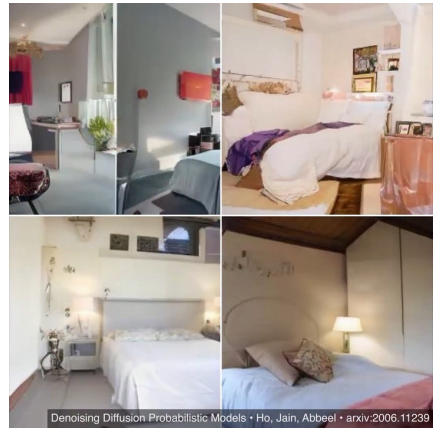


https://youtu.be/iellGrlNW7k

Application examples: Generative heat dissipation

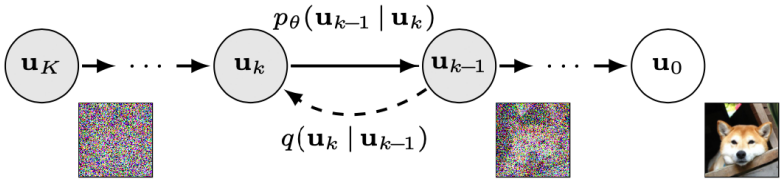
Background: Diffusion models

- Diffusion models: Iterative denoising using a NN $\rightarrow$ generative model
- Benefits:
  - Static training objective
  - No restrictions on NN architecture
  - Allows arbitrary depth
  - Optimizes the likelihood of the data $\rightarrow$ tries to cover entire distribution



Background: Diffusion models

- Deep latent-variable model with Markov structure



Generation (reverse):

$$p_\theta(\mathbf{u}_{k-1} \mid \mathbf{u}_k) \sim \mathcal{N}(\mu_\theta(\mathbf{u}_k, k), \Sigma)$$

$$p_\theta(\mathbf{u}_{0:K}) = p(\mathbf{u}_K) \prod_{k=1}^{K} p_\theta(\mathbf{u}_{k-1} \mid \mathbf{u}_k)$$

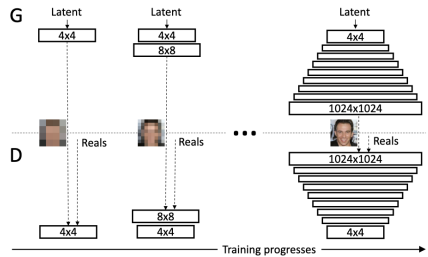
Inference (forward):

$$q(\mathbf{u}_k \mid \mathbf{u}_{k-1}) \sim \mathcal{N}(\sqrt{1-\beta_k}\mathbf{u}_{k-1}, \beta_k \mathbf{I})$$

$$q(\mathbf{u}_{1:K} \mid \mathbf{u}_0) = \prod_{k=1}^{K} q(\mathbf{u}_k \mid \mathbf{u}_{k-1})$$

Motivation for our approach

- Structure of images not directly reflected in the diffusion generative process
- Pixels are next to each other
- Multi-scale behaviour
- Taking this multi-resolution structure into account has lead to quantitative & qualitative improvements in, *e.g.*, GANs



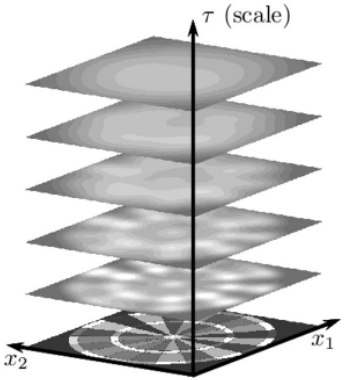
Karras *et al.*, ICLR 2018.

A scale-space view

- **Scale-space:** A way to represent images on multiple scales
- Resolution decrease defined by the heat equation

$$\frac{\partial}{\partial t} u(x, y, t) = \Delta u(x, y, t)$$

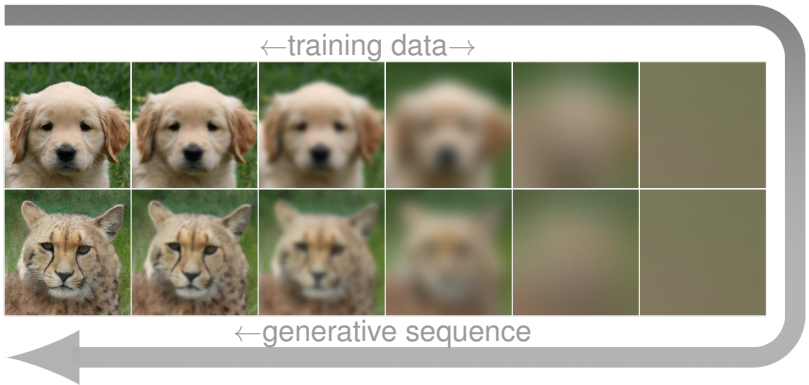
- Satisfies scale-space axioms:
  Scale invariance, rotational symmetry, ...



$$\frac{\partial}{\partial t} u(\mathbf{x}, t) = \Delta u(\mathbf{x}, t)$$

Generative inverse heat dissipation

Information melting forward process $\frac{\partial u}{\partial t} = \Delta u$



Generative inverse problem

Rissanen *et al.* (2023). *Generative inverse heat dissipation.* ICLR.

Comparison of frameworks

Diffusion model

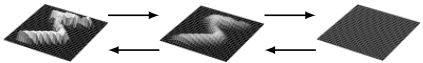
Information destroying forward process

Generative reverse process

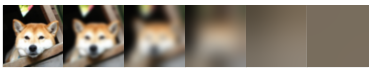
- Increasing dimensionality
- Increasing entropy
- Decreasing smoothness
- Diffusion in pixel space

Inverse heat dissipation model

Information destroying forward process



Generative reverse process



- Decreasing dimensionality
- Decreasing entropy
- Increasing smoothness
- Diffusion in 2D plane of image

The forward process $\frac{\partial u}{\partial t} = \Delta u$

- Choose boundary conditions: $u(x, y)$ derivatives zero at the edges
- The eigenbasis of the Laplace operator Δ in a rectangle is the cosine basis
- Solve discretized version efficiently and accurately using the discrete cosine transform (use the FFT)

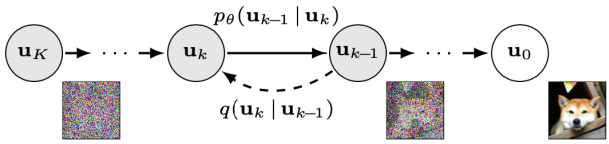
$$\mathbf{u}(t) = \mathbf{F}(t)\,\mathbf{u}(0) = \mathbf{V}\,\exp(\mathbf{\Lambda} t)\,\mathbf{V}^\top \mathbf{u}(0)$$

Pixel vecto — Forward operator — Initial state

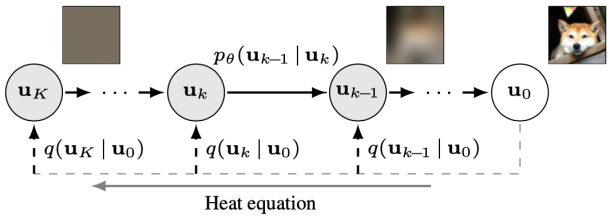
Inverse DCT — Diagonal scaling with freqs — DCT

Model formulation

Diffusion model

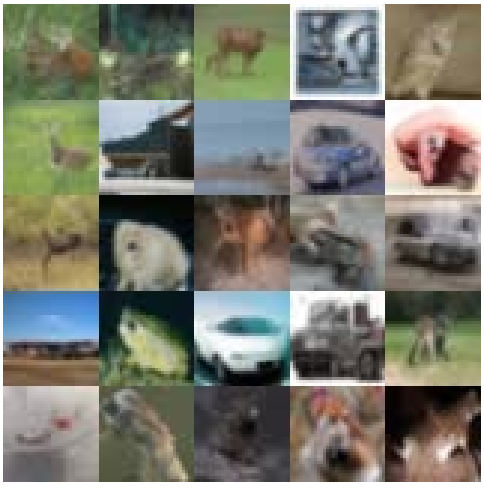


Inverse heat dissipation model



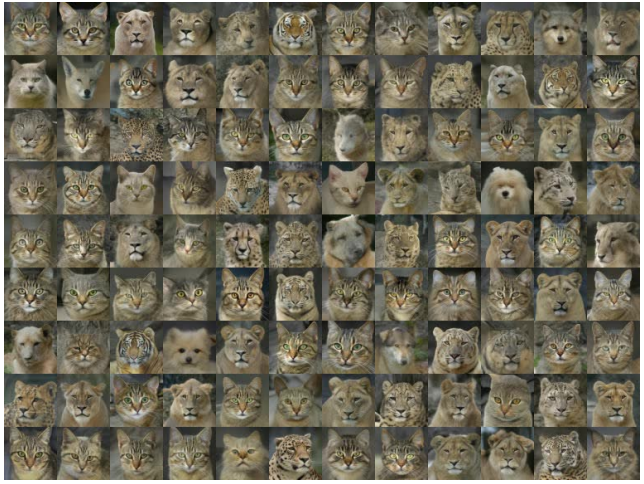
MNIST

CIFAR-10

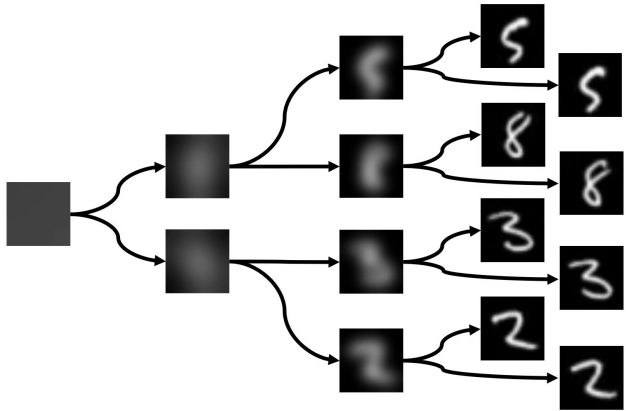


AFHQ

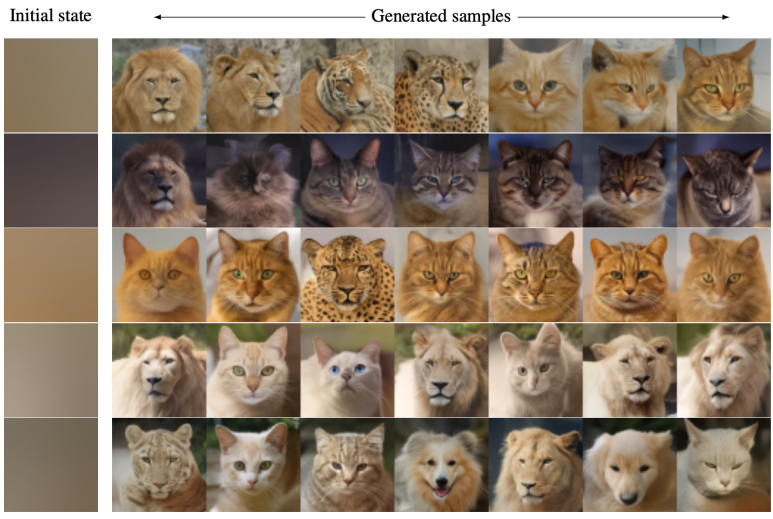
AFHQ: Same initialization



Hierarchy



Sample diversity



Recap

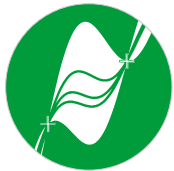
What did we go through now again?

Part I

Tools from signal processing

Part II

SDEs (continuous-time models)



Part III

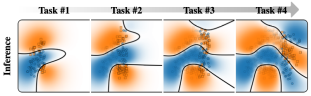
Temporal Gaussian processes

Part IV

Application examples

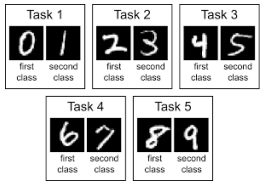
What is sequential machine learning?

Sequential inference / streaming



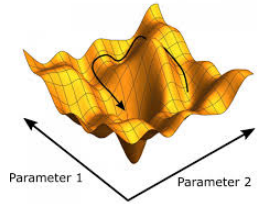
- **Timeseries**
- **Long/unbounded data**
- **Dynamical systems**
- Recurrent NNs *etc.*

Continual learning



- "Life-long learning"
- Non-stationarity
- Model keeps changing
- Data keep changing

Bayesian optimization / active learning



- How to pick points?
- Sequential decisions
- Connections to RL *etc.*

What to take home?

- In ML, we already do well in the large-data, gradient-based, static learning regime.
- We struggle when data is scarce, the model/data changes over time, and we require reliability/trust.
- We (should) build on principled foundations with tools that help us develop the next generation of tools.