Fast Gaussian Processes for Time Series

Will Tebbutt

Machine Learning Group, University of Cambridge
Invenia Labs, Cambridge

29-07-2020

Slides: `https://willtebbutt.github.io`

Invenia is hiring

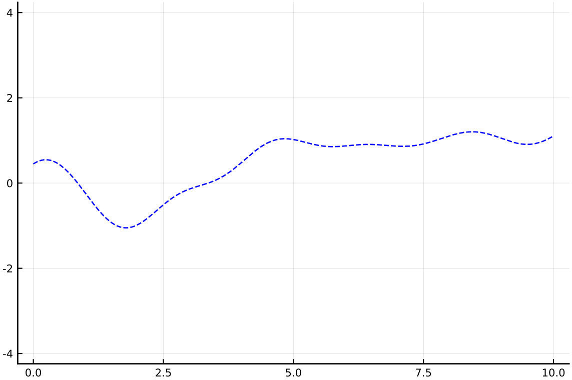
Invenia has over 30 Developers, Research Software Engineers, Machine Learning Reseachers, and Power Systems Researchers working full-time in Julia; **and we would like to have more.**

Come join us and contribute to our codebase of over **400,000 lines of Julia code.**

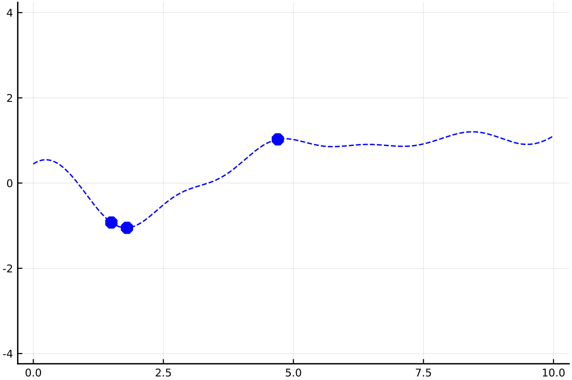
Also come see our JuliaCon 2020 talks

ChainRules.jl	NeuralProcesses.jl
Fast GPs for time series	ScoreDrivenModels.jl
HydroPowerModels.jl	Fancy Array Indexing BoF
NamedDims.jl	Julia In Production BoF

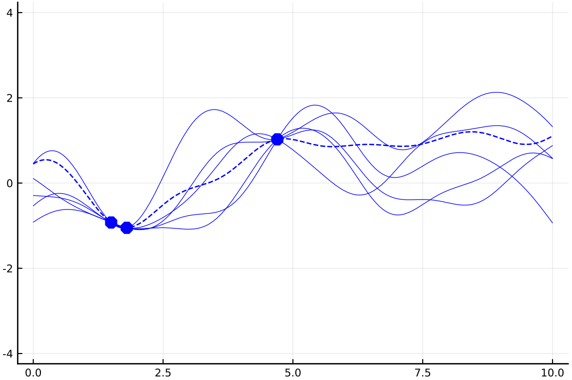
Gaussian processes in 1 Minute



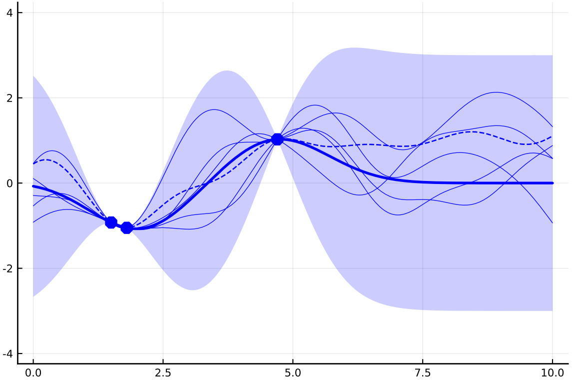
Gaussian processes in 1 Minute



Gaussian processes in 1 Minute



Gaussian processes in 1 Minute



Scalability Issues

- $\mathcal{O}\left(N^3\right)$ temporal complexity
- $\mathcal{O}\left(N^2\right)$ spatial complexity

Scaling

Ooooo that takes a while

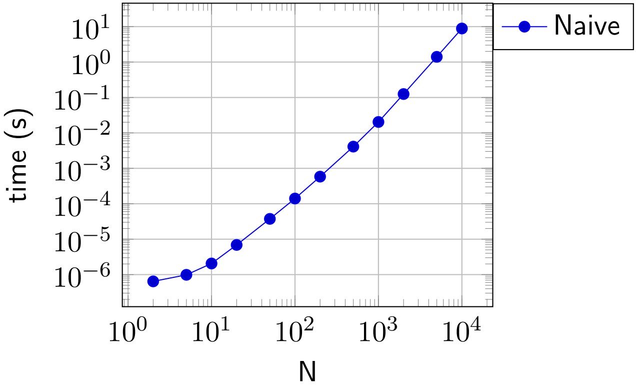


Figure: Naive log marginal likelihood computation requires $\mathcal{O}(N^3)$ time. Single thread. Uses Stheno.jl.

GPs as SDEs in a Nutshell

- Convert GP f into a linear SDE
- Convert linear SDE into Linear Gaussian SSM at times $t_{1:N}$
- Do inference e.g. compute $\log p(\mathbf{y}_{1:N})$

See [Särkkä and Solin, 2019] for details

TemporalGPs.jl Scaling

- Exact or almost exact inference
- $\mathcal{O}(ND^3)$ temporal complexity
- $\mathcal{O}(ND^2)$ spatial complexity
- D is reasonably small in lots of interesting cases.

TemporalGPs.jl Scaling

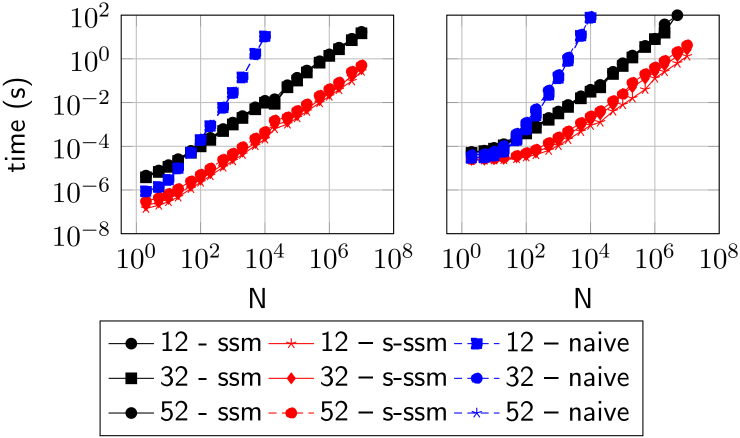


Figure: Time to compute log marginal likelihood (left) and log marginal likelihood + gradient (right). Naive uses Stheno.jl.

GPs as SDEs in a Nutshell

Use TemporalGPs.jl

```julia
using Stheno, TemporalGPs

# Specify a Stheno.jl GP as usual
f_naive = GP(Matern32(), GPC())

# Wrap it in an object that TemporalGPs knows how to handle.
f = to_sde(f_naive)

# Project onto finite-dimensional distribution as usual.
x = range(-5.0, 5.0; length=10_000_000)
fx = f(x, 0.1)

# Sample from the prior as usual.
y = rand(fx)

# Compute the log marginal likelihood of the data as usual.
logpdf(fx, y)
```

Features of TemporalGPs.jl

- Accelerate inference and learning in GPs from Stheno.jl
- Reverse-mode AD
- Checkpointing for memory-intensive problems (e.g. AD)
- Utilise StaticArrays.jl when D is small
- Spatio-temporal problems (small-medium space)

The Future

- Tidy up some of the API / types for prediction
- Integration with AbstractGPs.jl
- Further integration with Stheno.jl
- Non-Gaussian prediction problems

Acknowledgements

In no particular order...

- Rich Turner (my PhD supervisor)
- Arno Solin for teaching me a lot about these methods
- Lyndon White for reviewing the slides
- Various people in the MLG for helpful discussions
- Invenia Labs for its on-going support

Summary

- GPs scale poorly to large data
- TemporalGPs.jl makes them scale well for time-series.

Bibliography I

Hartikainen, J. et al. (2013).
Sequential inference for latent temporal gaussian process models.

Särkkä, S. and Solin, A. (2019).
Applied stochastic differential equations, volume 10.
Cambridge University Press.

Solin, A. et al. (2016).
Stochastic differential equation methods for spatio-temporal gaussian process regression.

Bibliographic Notes

GPs as Linear SDEs

- Final chapter of [Särkkä and Solin, 2019]
- Arno's these [Solin et al., 2016]
- Jouni Hartikainen's thesis: [Hartikainen et al., 2013]