

# INTRO TO GAUSSIAN PROCESSES

+ A LITTLE BIT OF JAX

# **WARNING**

*I don't really know anything about  
Machine Learning™.*

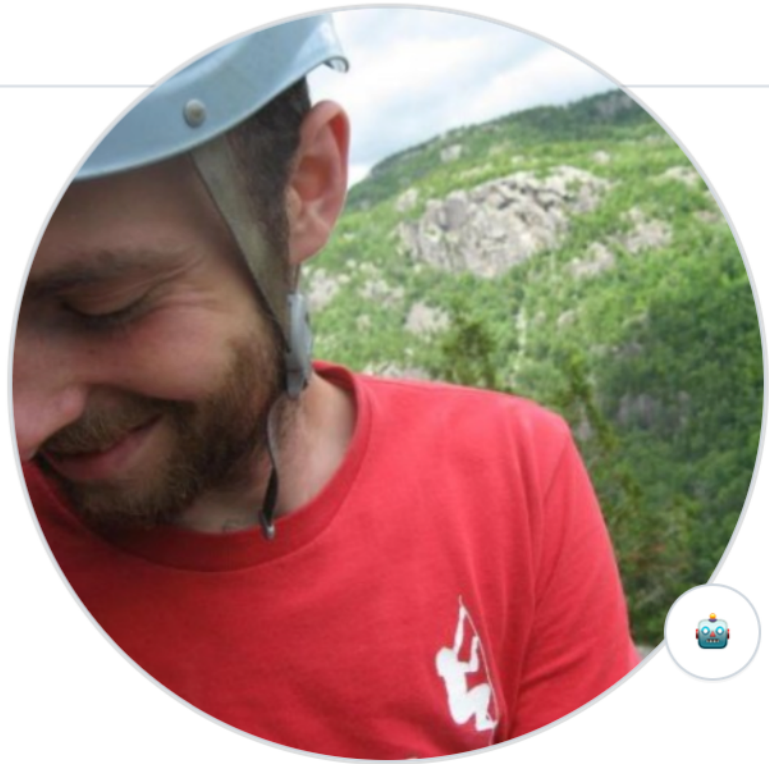
dfm (Dan Foreman-Mackey) · GitHub

https://github.com/dfm/

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search Sign in Sign up

Overview Repositories 408 Projects Packages 3 Stars 160



**Dan Foreman-Mackey**  
dfm


Follow

1.2k followers · 42 following

NYC

https://dfm.io

**Achievements**



**Highlights**

Developer Program Member

**Organizations**

dfm / README.md

### Hi, I'm Dan

I'm an Associate Research Scientist at the Flatiron Institute's Center for Computational Astrophysics in New York City. I'm interested in scientific software development, open and reproducible scientific practices, and many other things. I create too many GitHub repositories and I try to keep up with maintaining them.

Here are some of my coordinates online:

- dfm.io
- @exoplaneteer
- dfm@dfm.io

**Pinned**

**emcee** Public

The Python ensemble sampling toolkit for affine-invariant MCMC

Python 1.2k 411

**corner.py** Public

Make some beautiful corner plots

Python 378 198

**tinygp** Public

The tiniest of Gaussian Process libraries

Python 91 3

**daft-dev/daft** Public

Render probabilistic graphical models using matplotlib

Python 618 115

**exoplanet-dev/exoplanet** Public

Fast & scalable MCMC for all your exoplanet needs!

Python 156 44

**exoplanet-dev/celerite2** Public

Fast & scalable Gaussian Processes in one dimension

C++ 42 5

**get in touch!**

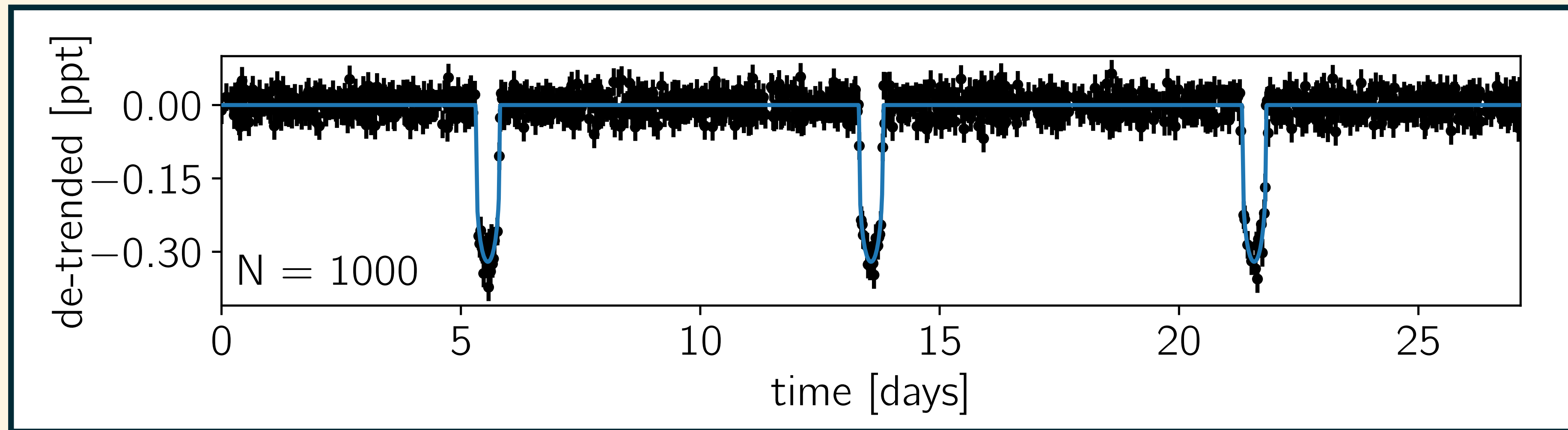
`dfm.io`

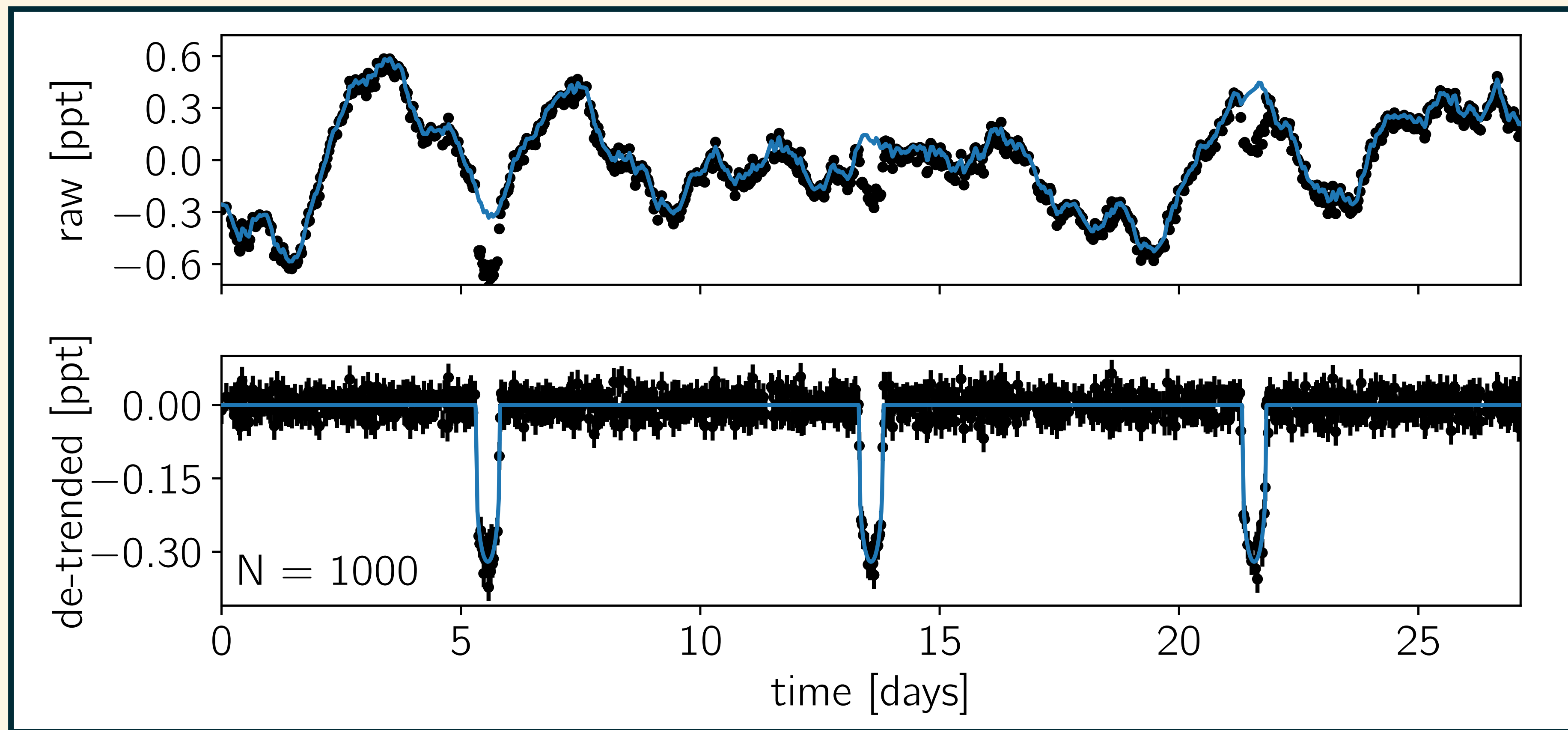
`github.com/dfm`

`twitter.com/exoplaneteer`

1

a **motivating** example



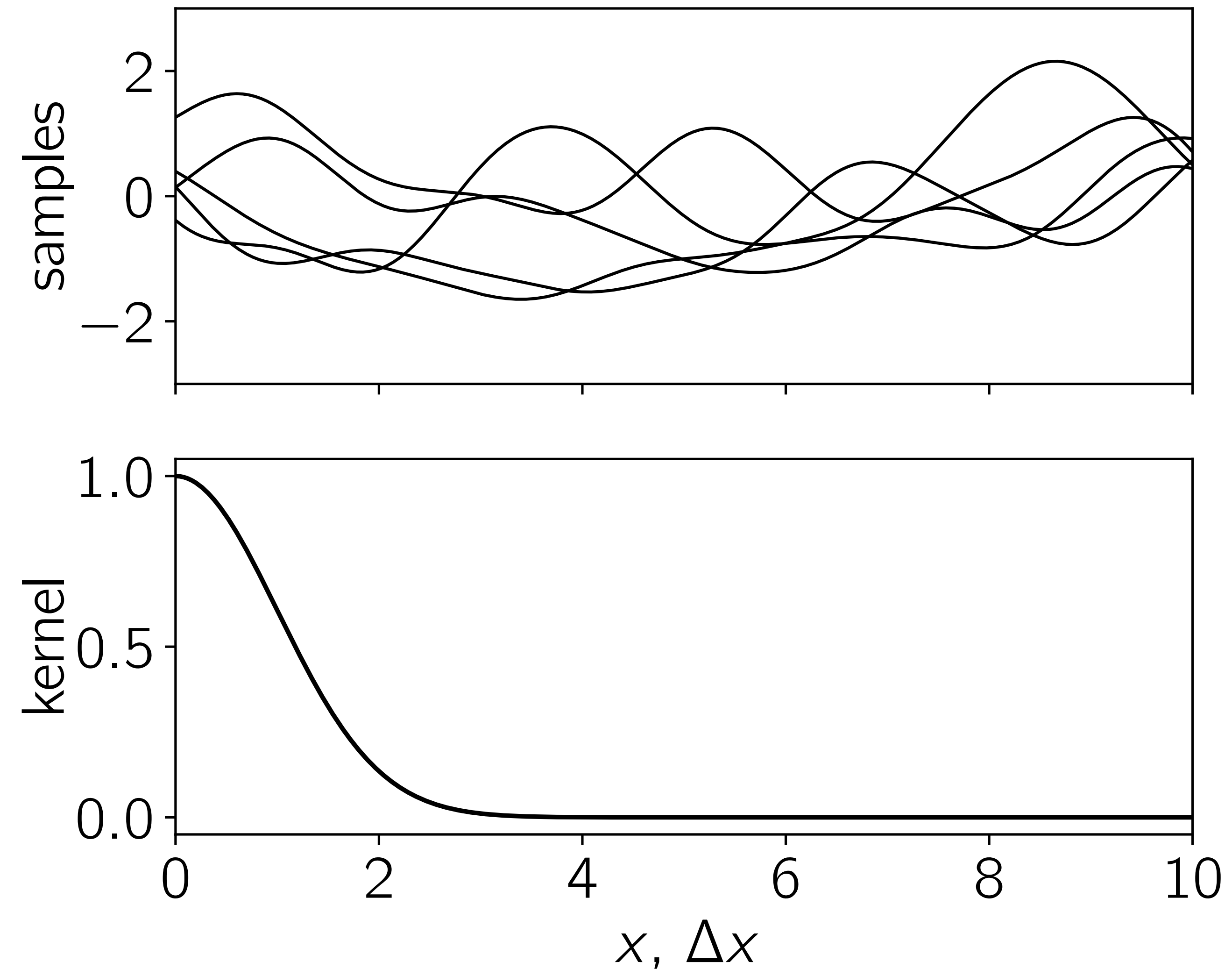


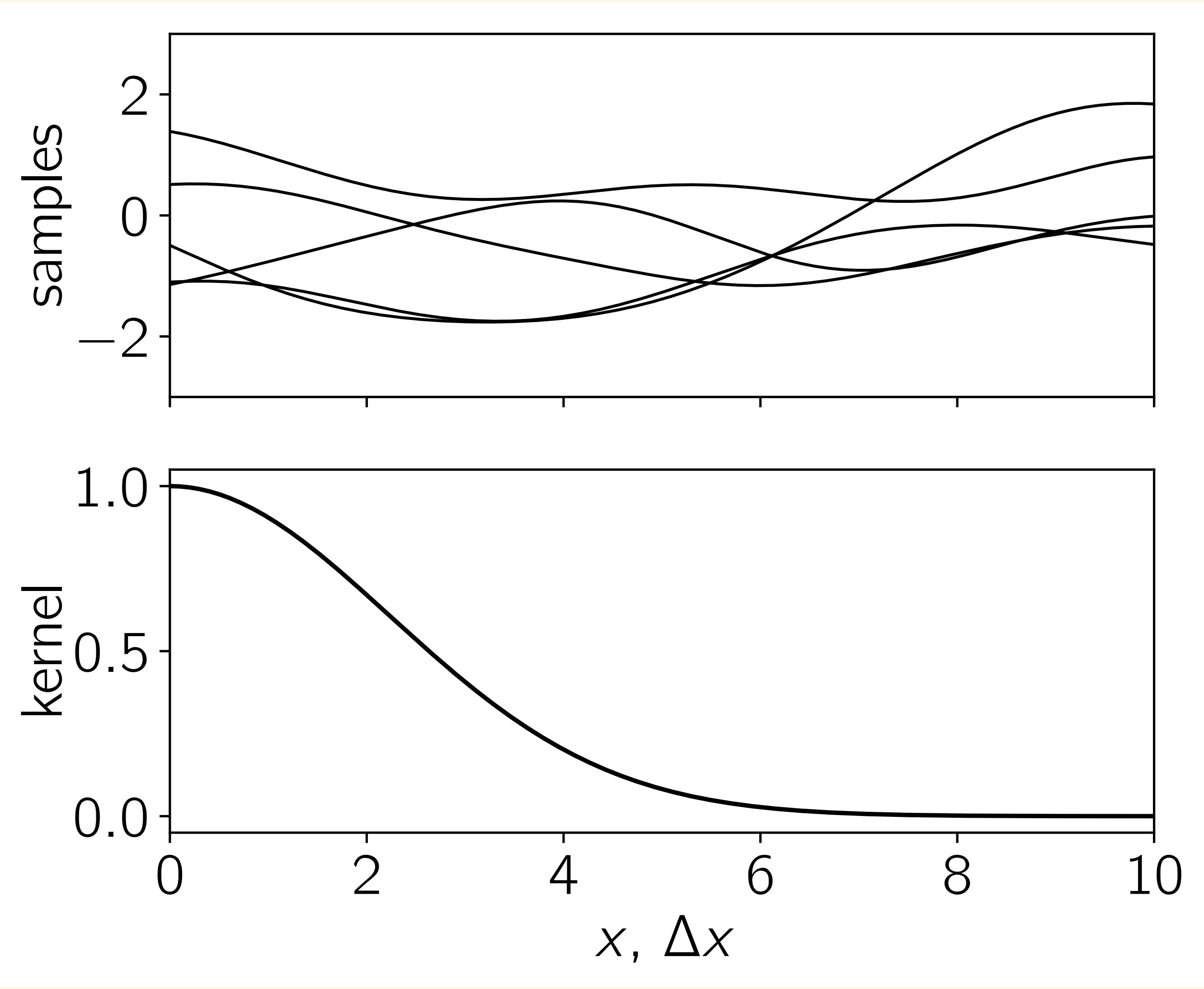
2

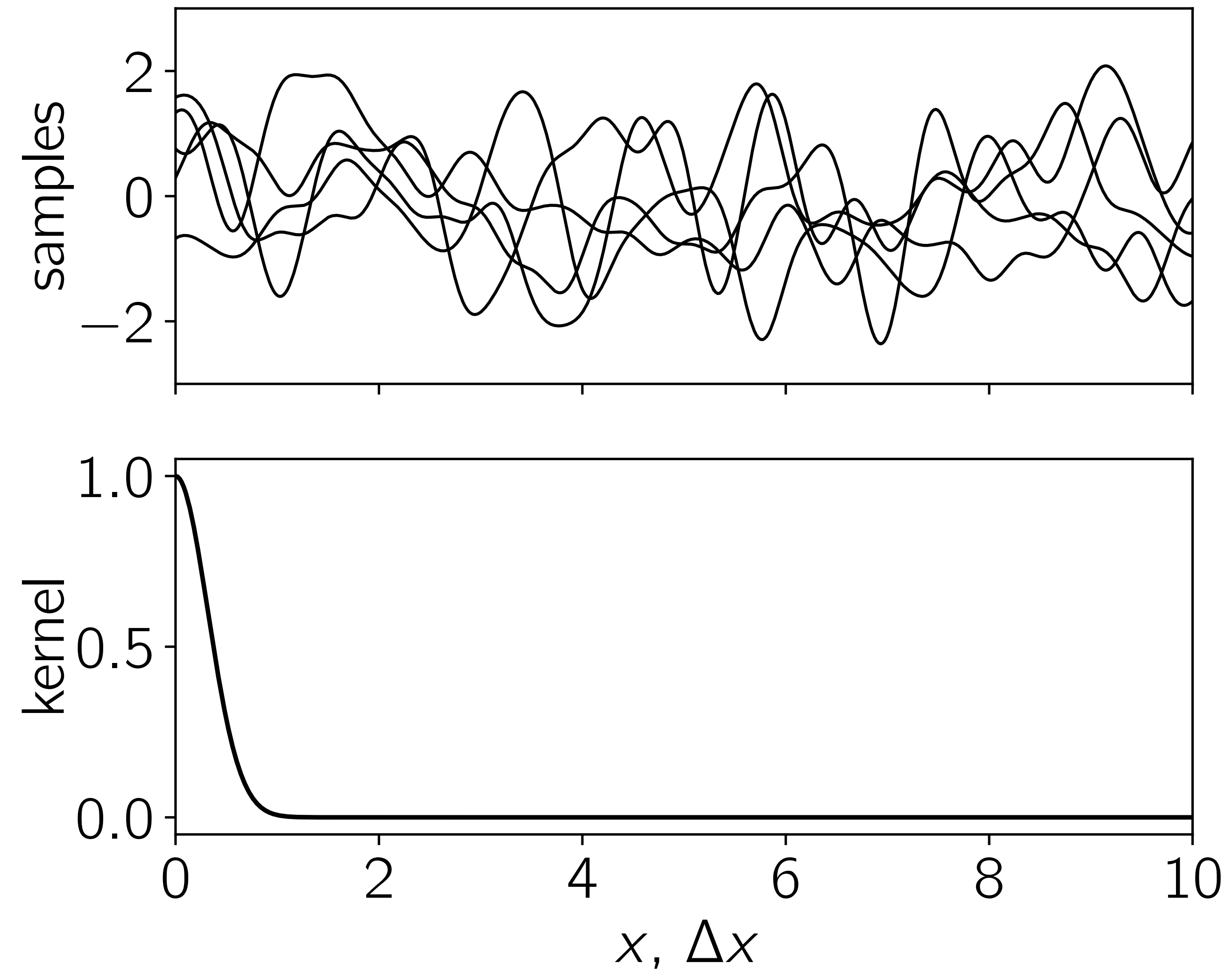
**theory** (on the board)

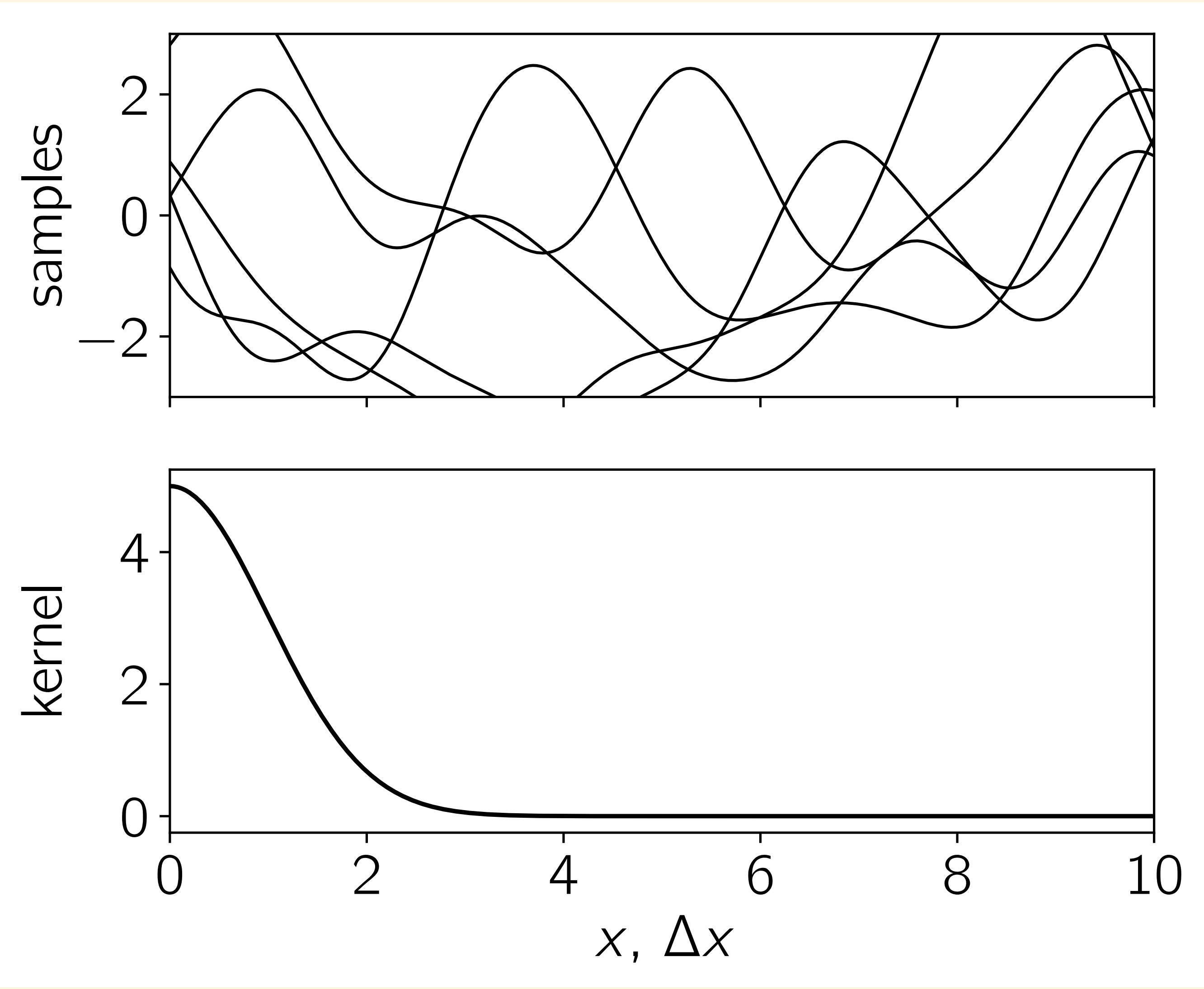














3

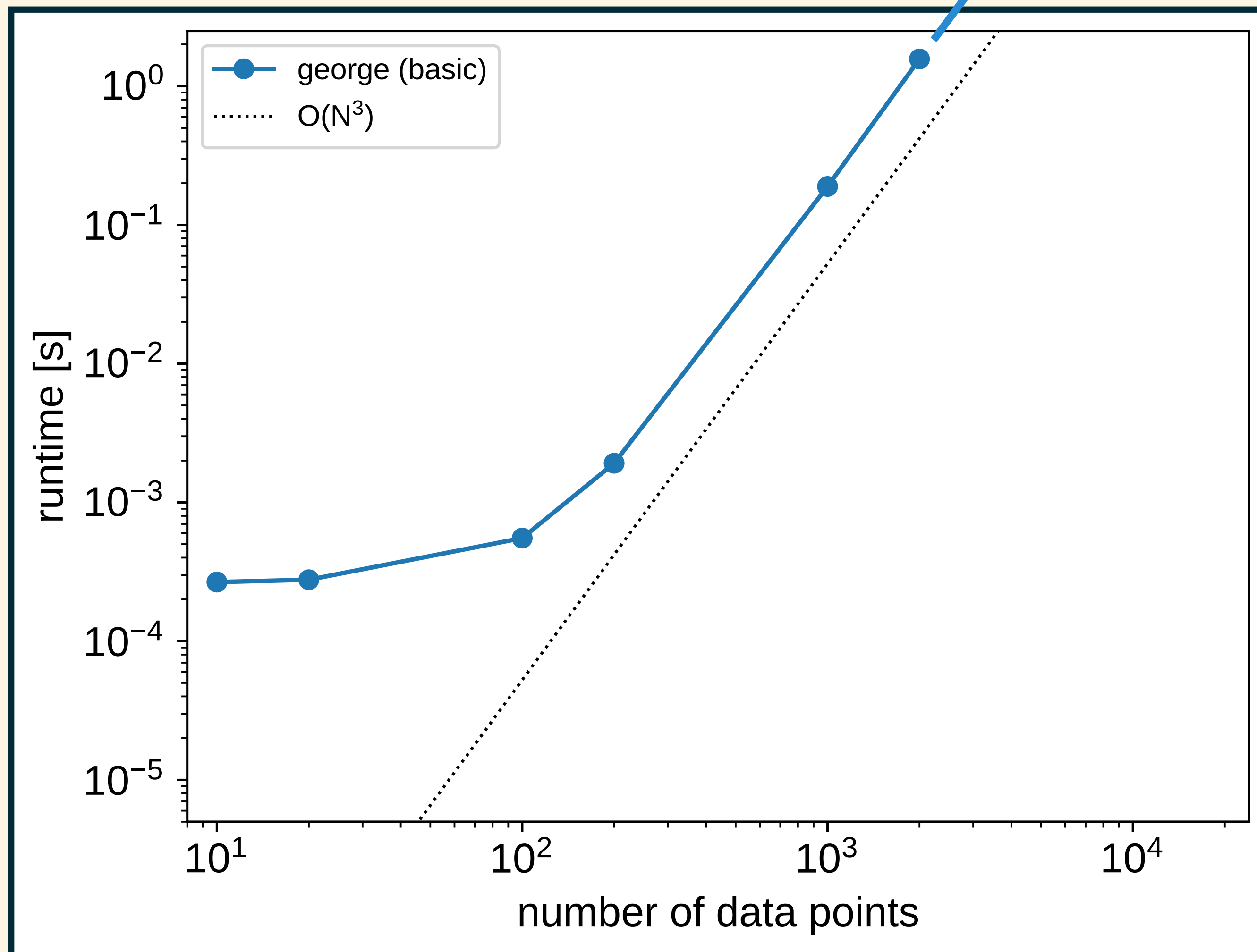
the **problem(s)**

[1] **kernel** choices

[2] **computational** cost



$$\mathbf{log\_like} = -0.5 * (\mathbf{r.T} @ \mathbf{K^{-1}} @ \mathbf{r} + \log(\mathbf{det(K)}))$$



$$\mathbf{log\_like} = -0.5 * (\mathbf{r.T} @ \mathbf{K^{*-1}} @ \mathbf{r} + \log(\mathbf{det(K)}))$$

$$\mathbf{mu\_pred} = \mathbf{Ks} @ \mathbf{K^{*-1}} @ \mathbf{r}$$

$$\mathbf{cov\_pred} = \mathbf{Kss} - \mathbf{Ks} @ \mathbf{K^{*-1}} @ \mathbf{Ks.T}$$



some **solutions**

- [1] **bigger/better** computers
- [2] exploit matrix **structure**
- [3] **approximate** linear algebra
- [4] **etc.**

- [1] **bigger/better** computers
  - [a] multicore **CPUs**
  - [b] **GPU** acceleration
  - [c] **iterative** methods

- [2] exploit matrix **structure**
  - [a] **uniformly** sampled data
  - [b] **Kronecker** products
  - [c] **sparsity**
  - [d] **quasiseparable** matrices

- [3] **approximate** linear algebra
  - [a] **subsample** data
  - [b] **randomized** methods
  - [c] approximate **structure**



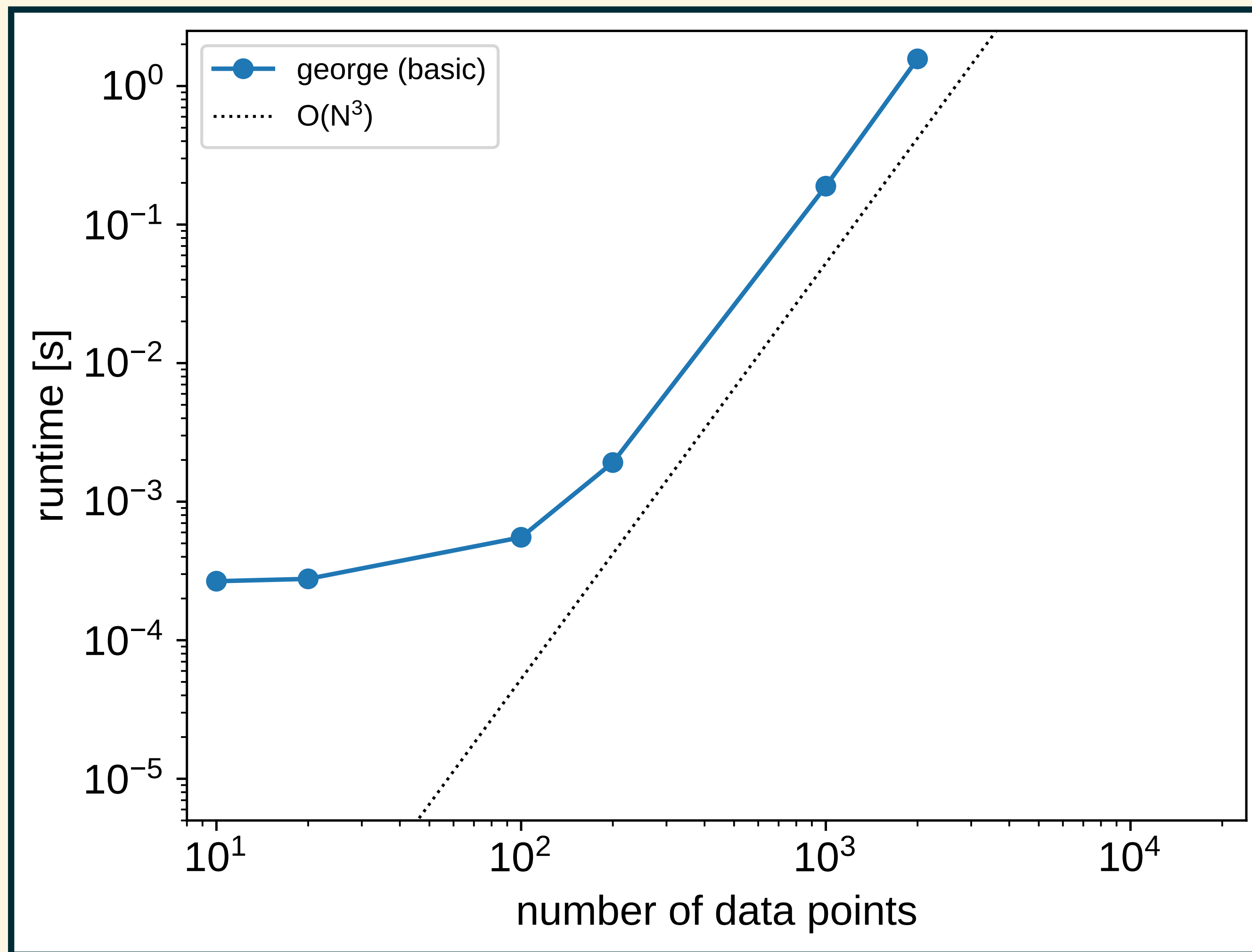
[4] **etc.**

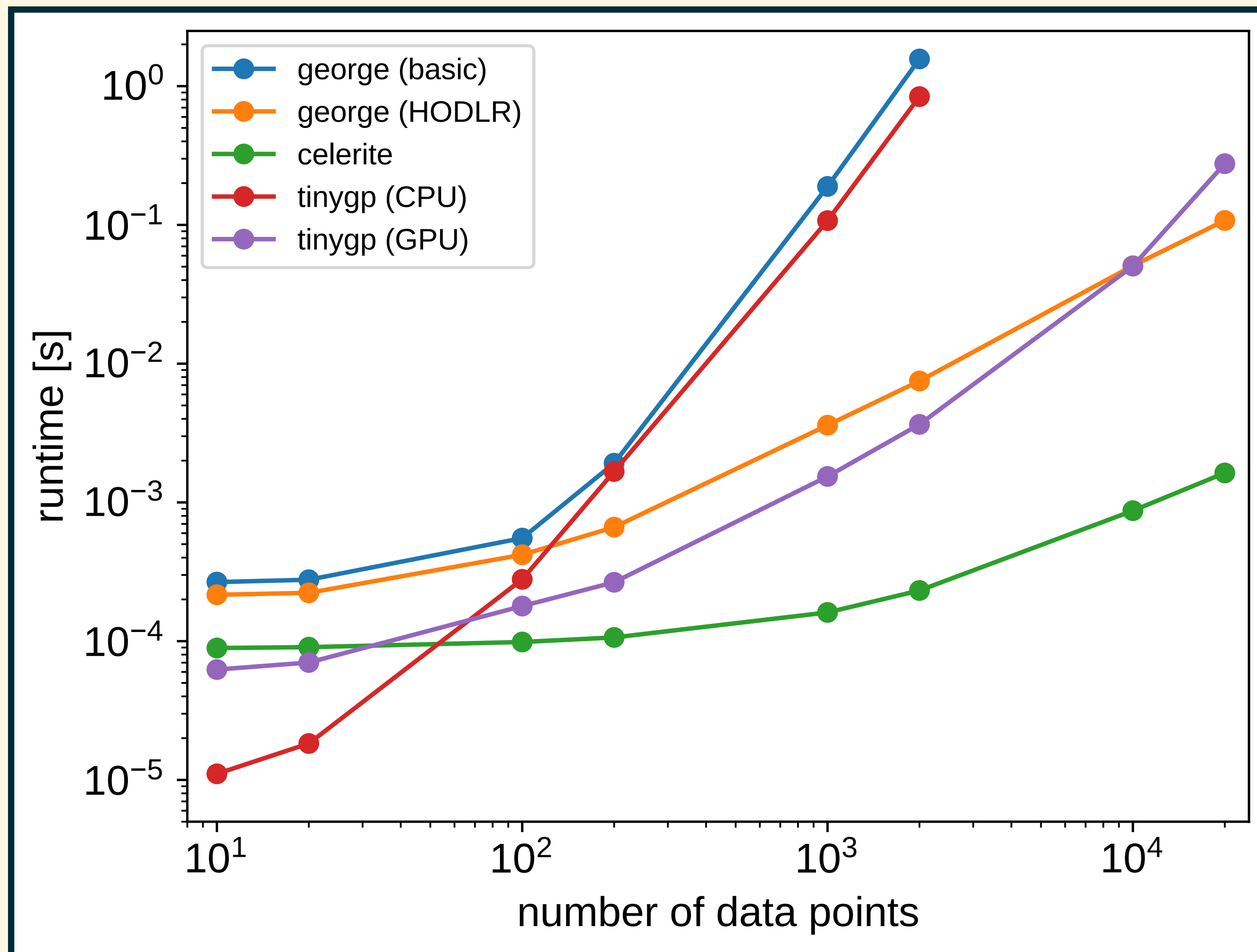
[a] **state-space** models

[b] stochastic **ODEs**

[c] **autoregressive** models

[d] ...





1

3

2

5

**implementation**

```
import numpy as np
```

```
def log_likelihood(params, x, diag, r):  
    K = build_kernel_matrix(params, x, diag)  
  
    gof = r.T @ np.linalg.solve(K, r)  
    norm = np.linalg.slogdet(K)[1]  
    return -0.5 * (gof + norm)
```

```
import numpy as np
from scipy.linalg import cho_factor, cho_solve

def log_likelihood(params, x, diag, r):
    K = build_kernel_matrix(params, x, diag)
    factor = cho_factor(K)
    gof = r.T @ cho_solve(factor, r)
    norm = 2 * np.sum(np.log(np.diag(factor[0])))
    return -0.5 * (gof + norm)
```

## what about...

- [1] different **kernels**?
- [2] **inference** methods?
- [3] **scalable** computations?
- [4] ...

some **examples**:

scikit-learn, GPy, PyMC, **GPyTorch**,  
george, celerite, **tinygp**,

...

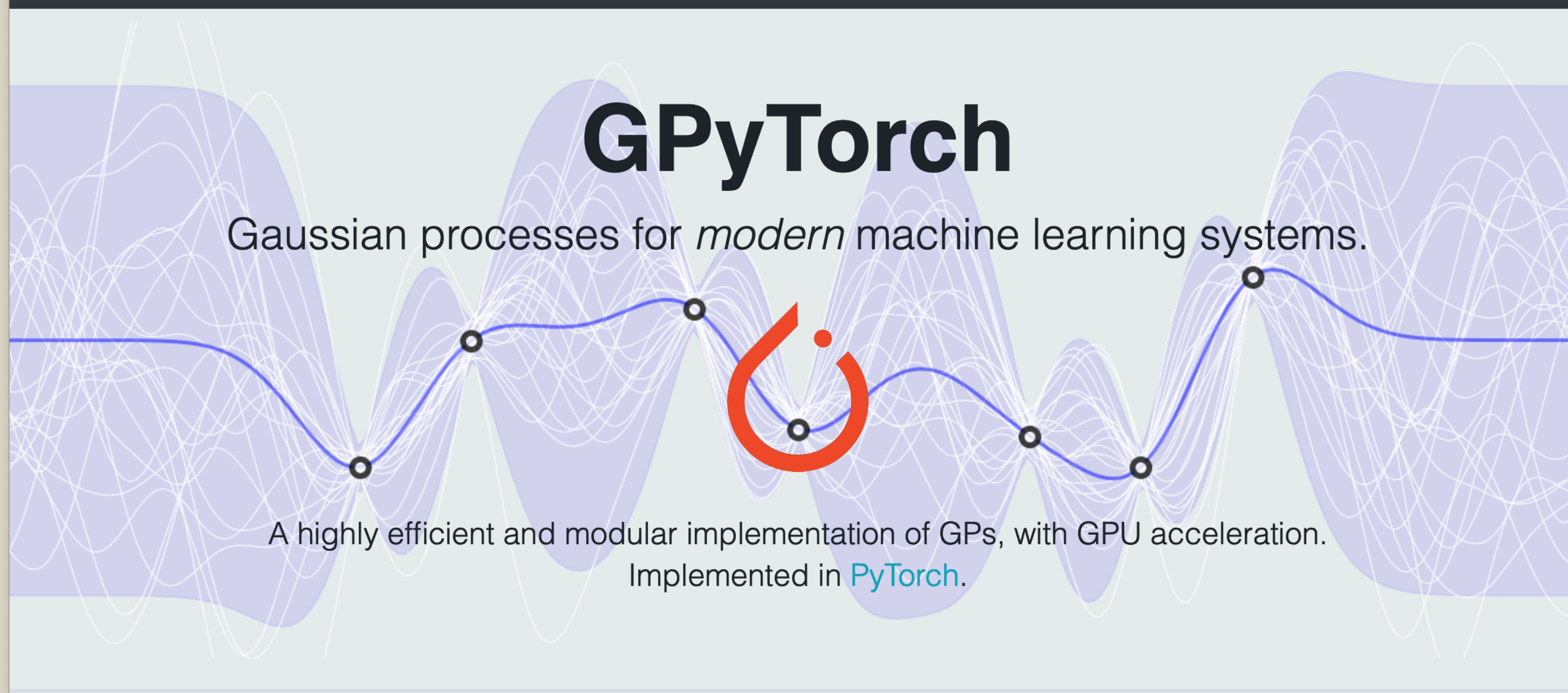


CORNELLIUS GP




[Install](#) [Docs](#) [Examples](#) [Github](#)

# GPyTorch

Gaussian processes for *modern* machine learning systems.



A highly efficient and modular implementation of GPs, with GPU acceleration.  
Implemented in [PyTorch](#).



Exact GPs with Scalable (GPU) | X

https://docs.gpytorch.ai/en/stable/examples/02\_Scalable\_Exact\_GPs/index.html

Exact GPs with Scalable (GPU) Inference

- BlackBox Matrix-Matrix Inference (BBMM)
- LancZos Variance Estimates (LOVE)
- Exact GPs with GPU Acceleration
- Scalable Posterior Sampling with CIQ
- Scalable Kernel Approximations
- Structure-Exploiting Kernels

Multitask/Multioutput GPs with Exact Inference

- Gaussian Process Latent Variable Models (GPLVM) with SVI
- Variational and Approximate GPs
- Deep GP and Deep Sigma Point Processes
- PyTorch NN Integration (Deep Kernel Learning)
- Pyro Integration
- Advanced Usage

PACKAGE REFERENCE

- gpytorch.models
- gpytorch.likelihoods

Read the Docs v: stable

Docs » Exact GPs with Scalable (GPU) Inference [Edit on GitHub](#)

## Exact GPs with Scalable (GPU) Inference

In GPyTorch, Exact GP inference is still our preferred approach to large regression datasets. By coupling GPU acceleration with [BlackBox Matrix-Matrix Inference](#) and [LancZos Variance Estimates \(LOVE\)](#), GPyTorch can perform inference on datasets with over 1,000,000 data points while making very few approximations.

### BlackBox Matrix-Matrix Inference (BBMM)

[BlackBox Matrix-Matrix Inference](#) (introduced by Gardner et al., 2018) computes the GP marginal log likelihood using only matrix multiplication. It is stochastic, but can scale exact GPs to millions of data points.

### LancZos Variance Estimates (LOVE)

[LancZos Variance Estimates \(LOVE\)](#) (introduced by Pleiss et al., 2019) is a technique to rapidly speed up predictive variances and posterior sampling. Check out the [GP Regression with Fast Variances and Sampling \(LOVE\)](#) notebook to see how to use LOVE in GPyTorch, and how it compares to standard variance computations.

### Exact GPs with GPU Acceleration

Here are examples of Exact GPs using GPU acceleration.

tinygp – tinygp

https://tinygp.readthedocs.io/en/latest/

tinygp

Search the docs ...

- User Guide
- Tutorials
- Contributor Guide
- Public API
- GitHub Repository

Theme by the Executable Book Project

tinygp

The tiniest of Gaussian Process libraries.

tinygp is an extremely lightweight library for building Gaussian Process (GP) models in Python, built on top of [jax](#). It has a [nice interface](#), and it's pretty fast (see [Benchmarks](#)). Thanks to [jax](#), [tinygp](#) supports things like GPU acceleration and automatic differentiation.

**How to find your way around?**

- A good place to get started is with the [Installation Guide](#) and then the [Tutorials](#). You might also be interested in the [Why tinygp?](#) page.
- For all the details, check out the [User Guide](#), including the [full API documentation](#).
- If you're running into getting [tinygp](#) to do what you want, first check out the [Troubleshooting](#) page, for some general tips and tricks.
- If [Troubleshooting](#) doesn't solve your problems, or if you find bugs, check out the [Contributor Guide](#) and then head on over to the [GitHub issues page](#).
- Check out the sidebar to find the full table of contents.

Contents

- Table of contents
- Authors & license

v: latest



Benchmarks — tinygp

https://tinygp.readthedocs.io/en/latest/benchmarks.html

tinygp

Search the docs ...

User Guide

- Why tinygp?
- Installation Guide
- Troubleshooting
- Benchmarks**
- Release Notes

Tutorials

Contributor Guide

Public API

GitHub Repository

Theme by the Executable Book Project

```
plt.xlabel("number of data points")
plt.ylabel("runtime [s]");
```

number of data points	george (exact)	george (approx)	celerite2 (struct)	tinygp (exact; CPU)	tinygp (exact; GPU)	tinygp (struct; CPU)
10 <sup>1</sup>	~10 <sup>-4</sup>	~10 <sup>-4</sup>	~10 <sup>-4</sup>	~10 <sup>-5</sup>	~10 <sup>-4</sup>	~10 <sup>-5</sup>
10 <sup>2</sup>	~10 <sup>-2</sup>	~10 <sup>-2</sup>	~10 <sup>-4</sup>	~10 <sup>-3</sup>	~10 <sup>-3</sup>	~10 <sup>-4</sup>
10 <sup>3</sup>	~10 <sup>0</sup>	~10 <sup>-1</sup>	~10 <sup>-3</sup>	~10 <sup>-1</sup>	~10 <sup>-2</sup>	~10 <sup>-4</sup>
10 <sup>4</sup>	-	~10 <sup>-1</sup>	~10 <sup>-2</sup>	-	~10 <sup>-1</sup>	~10 <sup>-3</sup>
10 <sup>5</sup>	-	-	-	-	-	~10 <sup>-2</sup>

Previous [Troubleshooting](#) Next [Release Notes](#)

v: latest

© Copyright 2021, 2022 Simons Foundation, Inc.

6

**try** it out!

**get in touch!**

`dfm.io`

`github.com/dfm`

`twitter.com/exoplaneteer`