Probabilistic Numerical Methods

An Algorithmic Perspective

Jonathan Wenger

MCM 2021

August 19, 2021

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

MAX PLANCK INSTITUTE FOR INTELLIGENT SYSTEMS

imprs-is

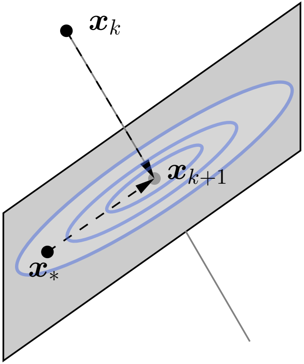
What is a Probabilistic Numerical Method?

…and why should I care?

Numerical methods that return **probability measures** designed to quantify numerical error.

Advantages

- *structural uncertainty* via a probability measure compared to an error estimate
- *customized methods* for specific problems with bespoke priors
- set *parameters of numerical methods* via the Bayesian formalism
- solution of *mutually related problems* of similar type
- incorporate *sources of stochasticity* in the computation



Vision: Propagation of uncertainty through chains of computations.

Problem: Consider the 1D quadratic optimization problem

$$\min_{x \in \mathbb{R}} f(x) = \min_{x \in \mathbb{R}} \frac{1}{2} a x^2 + bx + c$$

where $a > 0$. Since f is strictly convex, the unique minimum is given by $f'(x_*) = 0 \iff x_* = -\frac{b}{a}$.

Introducing Noise: Suppose we only have access to noisy evaluations

$$\hat{y} = \hat{f}(x) = \frac{1}{2}(a + \varepsilon_a)x^2 + (b + \varepsilon_b)x + (c + \varepsilon_c)$$

of the objective function f at x, where $\varepsilon = (\varepsilon_a, \varepsilon_b, \varepsilon_c) \sim \mathcal{N}(0, \Lambda)$ and $\Lambda \in \mathbb{R}^{3 \times 3}$ symmetric positive definite.

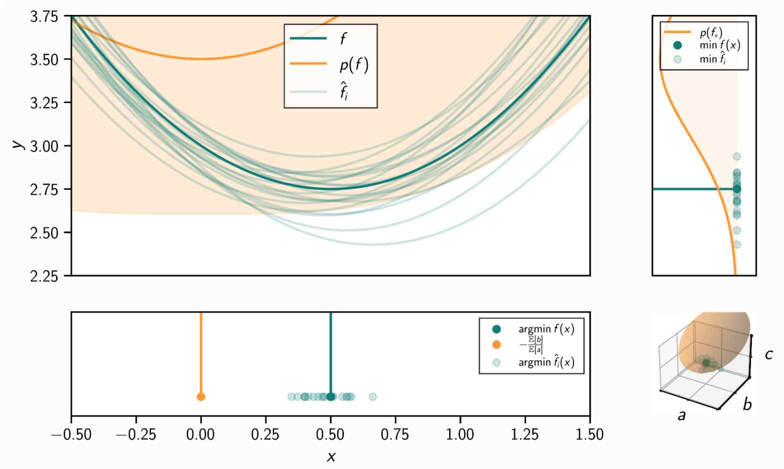
Can we still design an efficient numerical method to find a solution?

Components of a (Bayesian) PN Method

1D Noisy Quadratic Optimization

Components

- Prior Belief
- Policy and Action
- Information Operator
- Belief Update
- Stopping Criterion
- Hyperparameter Optimization



But what *exactly* is a PN Method?

A formal definition and an algorithmic translation.

Definition (Cockayne et al. [1])

Let $(\mathcal{X}, \Sigma_{\mathcal{X}})$, $(\mathcal{A}, \Sigma_{\mathcal{A}})$ and $(\mathcal{Q}, \Sigma_{\mathcal{Q}})$ be measurable spaces and X a random variable on $(\mathcal{X}, \Sigma_{\mathcal{X}})$. Take $\mu(B) = \mathbb{P}(\{w : X(w) \in B\})$ to be the law of X. Further, let $A : \mathcal{X} \to \mathcal{A}$, $Q : \mathcal{X} \to \mathcal{Q}$ and $B : \mathcal{P}_{\mathcal{X}} \times \mathcal{A} \to \mathcal{P}_{\mathcal{Q}}$ where A and Q are measurable functions.

The pair $M = (A, B)$ is called a probabilistic numerical method for estimation of a quantity of interest Q. The map A is called an information operator, and the map B is called a belief update operator.

Goal: Given a problem $F(\boldsymbol{x}) = 0$, return a quantity of interest $q(\boldsymbol{x})$.

Algorithm Probabilistic Numerical Method

procedure ProbNumMethod($F(\cdot)$, $\mathbf{x}$) # problem & prior
  while not HasConverged(F, $\mathbf{x}$, $\mathbf{q}$) **do**
    $\boldsymbol{s} \leftarrow$ Policy(F, $\mathbf{x}$, $\mathbf{q}$) # action via policy
    $\boldsymbol{y} \leftarrow$ InformationOp(F, $\mathbf{x}$, $\mathbf{q}$, $\boldsymbol{s}$)
    $\theta_* \leftarrow$ HyperparamOptim(θ, $\boldsymbol{S}$, $\boldsymbol{Y}$)
    $\mathbf{q} \leftarrow$ BeliefUpdate($(\boldsymbol{s}, \boldsymbol{y})$, F, $\mathbf{x}$, $\mathbf{q}$, θ_*)
  end while
  return $\mathbf{q}$
end procedure

Modular and composable *implementations of PN methods*?

ProbNum

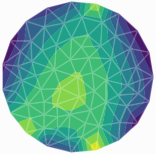
Probabilistic Numerics in Python.

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

ProbNum

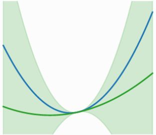
Learn to Approximate. Approximate to Learn.

Get Started 🚀



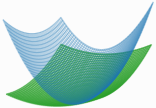
Solve Numerical Problems

Solve problems from linear algebra, optimization, quadrature and differential equations using probabilistic inference.



Quantify Uncertainty in Computation

Quantify and propagate uncertainty from finite resources and stochastic input in computational pipelines.



Compose Custom Numerical Methods

Create problem-specific probabilistic numerical methods from predefined or your own custom components.

ProbNum is a Python library for solving numerical problems.

- Linear solvers: $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$
- ODE solvers: $\dot{\boldsymbol{y}}(t) = f(\boldsymbol{y}(t), t)$
- Integral solvers (quadrature): $F = \int f(x)p(x)d\,x$
- Filters and smoothers
- Random variables and random processes
- Linear operators $\boldsymbol{v} \mapsto \boldsymbol{A}\boldsymbol{v}$

Goals

- expedite the *implementation* of new PN methods
- make PN methods available for *applications*
- demonstrate the *use cases* of PN

ProbNum

Linear Algebra	Quadrature
Differential Equations	Bayesian Optimization

Contributors 13

+ 2 contributors

Design Principles

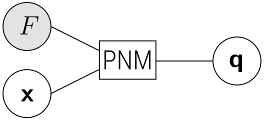
EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

In- and Outputs: Random variables / processes.

Interface: Drop-in replacement for NumPy / SciPy

```
# Solve using NumPy
x = np.linalg.solve(A, b)

# Solve using ProbNum
x_rv, _, _, info = pn.linalg.problinsolve(A, b)
```

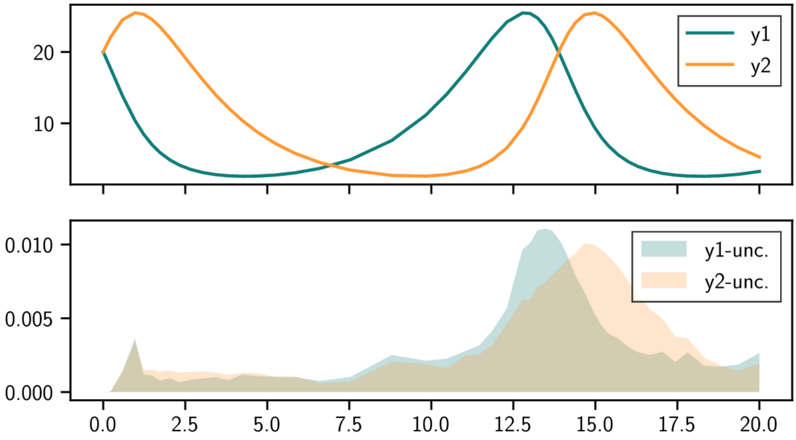


Implementation: Modularity enables custom PN Methods

```
# Construct custom solver
pls = pn.linalg.ProbabilisticLinearSolver(
  policy = ConjugateDirections(),
  information_op = MatrixVectorProduct(),
  stopping_criterion = PosteriorContraction(tol=e-5))

# Solve problem
pls.solve(problem = (A, b))
```

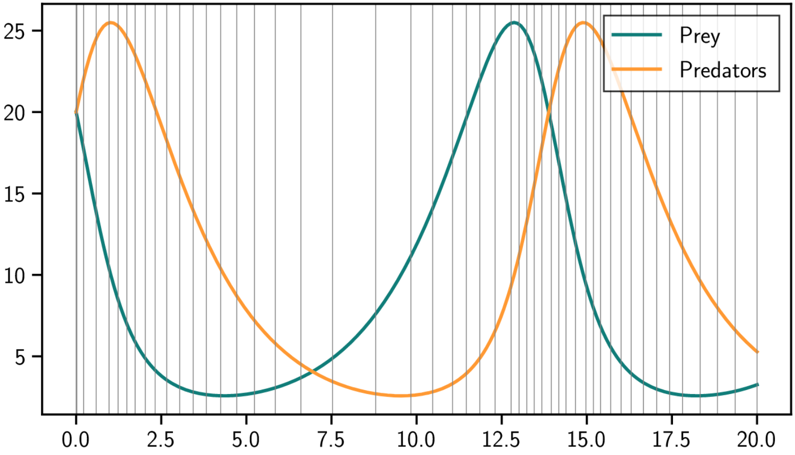
Problem: Lotka-Volterra equations (first-order, non-linear ODE)



Case Study: ODE Filters

Adaptive step-size selection [2, 3].

Problem: Lotka-Volterra equations (first-order, non-linear ODE)



Summary

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

ProbNum

- *Solve* numerical problems.
- *Quantify* uncertainty in computation.
- *Compose* custom probabilistic numerical methods.

PyPI	`pip install probnum`
GitHub	https://github.com/probabilistic-numerics/probnum
Documentation	https://probnum.readthedocs.io
Tutorials	https://probnum.readthedocs.io/en/latest/tutorials.html

Questions or Feedback?

References I

[1] Jon Cockayne, Chris Oates, TJ Sullivan, and Mark Girolami. Bayesian probabilistic numerical methods. SIAM Review, 61(4):756–789, 2019.

[2] Michael Schober, Simo Särkkä, and Philipp Hennig. A probabilistic model for the numerical solution of initial value problems. Statistics and Computing, 29(1):99–122, 2019.

[3] Nathanael Bosch, Philipp Hennig, and Filip Tronarp. Calibrated adaptive probabilistic ode solvers. In Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS), volume 130 of Proceedings of Machine Learning Research, pages 3466–3474. PMLR, 2021. URL `http://proceedings.mlr.press/v130/bosch21a.html`.