Probabilistic Linear Solvers for Machine Learning

Jonathan Wenger Philipp Hennig

NeurIPS 2020

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Max Planck Institute for Intelligent Systems

imprs-is

erc

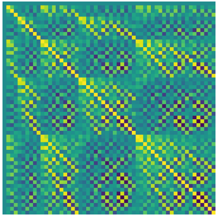
some of the presented work is supported by the European Research Council.

$$\boldsymbol{A}\boldsymbol{x}_* = \boldsymbol{b}$$

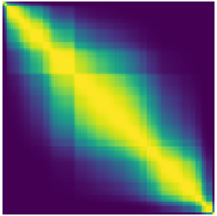
where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ symmetric positive definite.

ML-specific Challenges

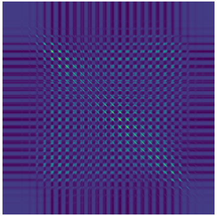
- large-scale systems
- characteristic structure
- generative information
- subject to noise



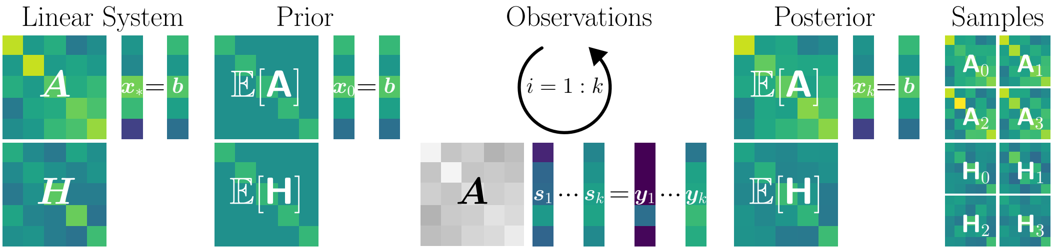
general linear model $\boldsymbol{X}^\top \boldsymbol{X}$



kernel matrix $\boldsymbol{K} = k_{\text{RBF}}(\boldsymbol{X}, \boldsymbol{X})$



Hessian matrix $\boldsymbol{H} = \nabla^2 f(\boldsymbol{\theta})$



Algorithm Components

1. Prior $p(\mathbf{A})$ or $p(\mathbf{H})$
2. Policy $\boldsymbol{s}_i = -\mathbb{E}[\mathbf{H}](\boldsymbol{A}\boldsymbol{x}_{i-1} - \boldsymbol{b})$
3. Observation $\boldsymbol{y}_i = \boldsymbol{A}\boldsymbol{s}_i$
4. Belief Update for **A** and **H**
5. (Uncertainty Calibration)

Theoretical Properties

- Conjugate Directions Method ($\leq n$ iters)
- Recovers CG given certain priors
- Time complexity $\mathcal{O}(kn^2)$
- Space complexity $\mathcal{O}(kn)$

Covariance Class and Uncertainty Calibration

Prior Information Improves Uncertainty Estimation

Desiderata

No.	Property	
(1)	distribution over matrices	✓
(2)	symmetry	✓
(3)	positive definiteness	~
(4)	positive linear combination in same distr. family	✓
(5)	corresponding priors on matrix and inverse	✓
⋮	⋮	⋮

Covariance Class

$$p(\mathsf{A}) = \mathcal{N}(\mathsf{A}; \boldsymbol{A}_0, \boldsymbol{W}_0^{\mathsf{A}} \otimes \boldsymbol{W}_0^{\mathsf{A}})$$

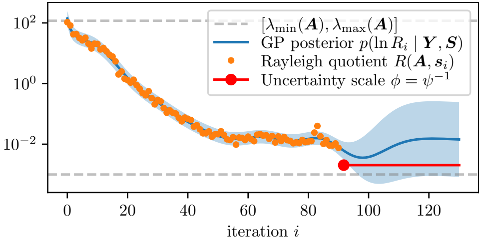
$$p(\mathsf{H}) = \mathcal{N}(\mathsf{H}; \boldsymbol{A}_0^{-1}, \boldsymbol{W}_0^{\mathsf{H}} \otimes \boldsymbol{W}_0^{\mathsf{H}})$$

where $\boldsymbol{W}_0^{\mathsf{A}}$ and $\boldsymbol{W}_0^{\mathsf{H}}$ admit degrees of freedom for *uncertainty calibration*.

Prior Spectral Knowledge

Spectrum $\lambda(\boldsymbol{A})$

Rayleigh Quotient $R(\boldsymbol{A}, \boldsymbol{s}) = \frac{\boldsymbol{s}^\top \boldsymbol{A} \boldsymbol{s}}{\boldsymbol{s}^\top \boldsymbol{s}}$



Software and Applications

ProbNum: Probabilistic Numerics in Python

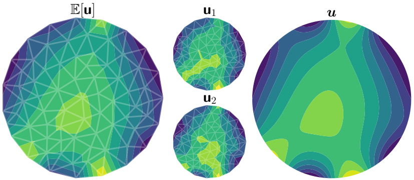
EBERHARD KARLS UNIVERSITÄT TÜBINGEN

ProbNum implements probabilistic numerical methods in Python.

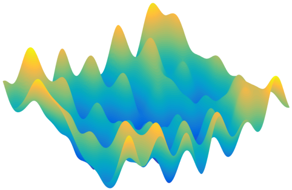
`https://github.com/probabilistic-numerics/probnum`

or alternatively `pip install probnum`.

Future Applications



Galerkin Methods $\boldsymbol{A}\boldsymbol{u} = \boldsymbol{f}$



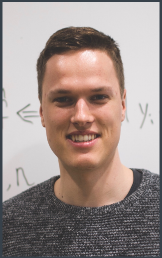
Empirical Risk Minimization $\mathbf{H}\boldsymbol{d} = \mathbf{g}$

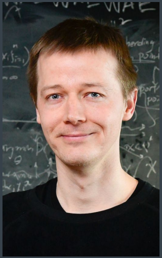
Summary

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Probabilistic Linear Solvers for Machine Learning

Jonathan Wenger and Philipp Hennig





- Linear systems in ML exhibit characteristic structure.
- Probabilistic linear solvers make use of prior generative information.
- Limited computational resources induce *numerical uncertainty*.

Preprint	arXiv	2010.09691
Implementation	$f(\Lambda)$	https://github.com/probabilistic-numerics/probnum
Experiments		https://github.com/JonathanWenger/probabilistic-linear-solvers-for-ml