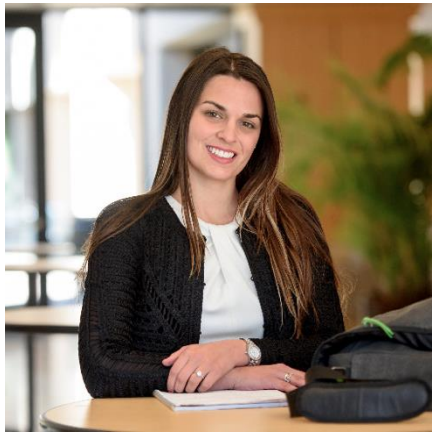
End-to-end Probabilistic Framework for Learning with Hard Constraints

14th International Conference on Learning Representations

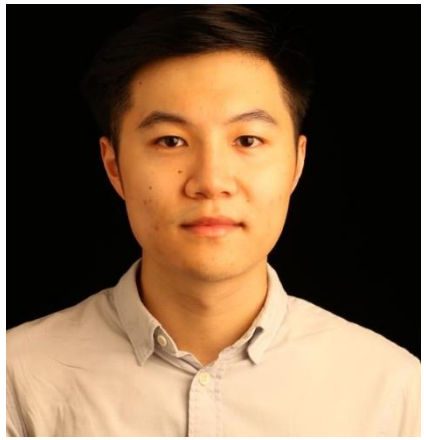
Rio De Janeiro, Brazil

Collaborators

Utkarsh



Danielle C. Maddix



Ruijun Ma

Michael W. Mahoney

Bernie Wang

The Problem in Hand

Learn this map:

$$\hat{f}_\theta : \Phi \to \mathcal{Y}$$

$$\mathbf{Y}_\theta(\phi^{(i)}) \in \mathbb{R}^n$$

$$\hat{u}(\phi^{(i)}) \sim \mathbf{Y}_\theta(\phi^{(i)})$$

Inputs maps to "spatial/temporal probabilistic predictions"

Predicted random variable

Seems simple and straightforward?

What if I want some constraints on my predictions?

$$\hat{f}_\theta : \Phi \to \mathcal{Y}$$

$$\mathbf{Y}_\theta(\phi^{(i)}) \in \mathbb{R}^n$$

$$\hat{u}(\phi^{(i)}) \sim \mathbf{Y}_\theta(\phi^{(i)})$$

$$h(\hat{u}(\phi^{(i)})) = 0$$

$$g(\hat{u}(\phi^{(i)})) \leq 0$$

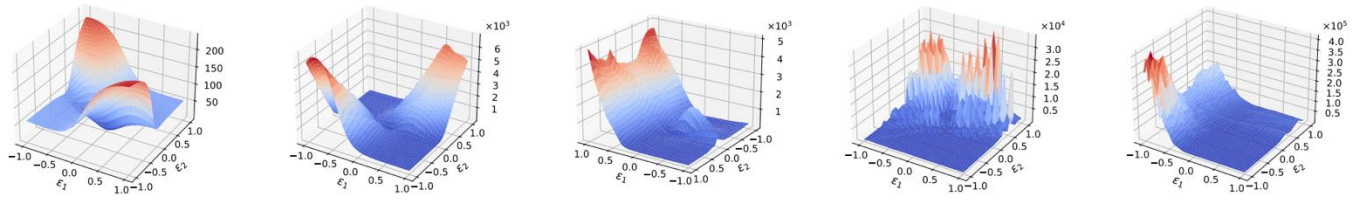
Inputs maps to "spatial/temporal probabilistic predictions"

Predicted random variable

Is it straightforward enough?

Regularization: Standard Trick

- **Soft constraints or Regularization only "encourage" the solution to satisfy the constraints**
- **Requires hyper-parameter tuning**
- **Impose conflicting training objectives**
- **A core problem in Physics-based ML**



(a) $\beta = 1.0$ (b) $\beta = 10.0$ (c) $\beta = 20.0$ (d) $\beta = 30.0$ (e) $\beta = 40.0$

β	1	10	20	30	40
Relative error	7.84×10^{-3}	1.08×10^{-2}	7.50×10^{-1}	8.97×10^{-1}	9.61×10^{-1}
Absolute error	3.17×10^{-3}	6.03×10^{-3}	4.32×10^{-1}	5.42×10^{-1}	5.82×10^{-1}

Figure 3: ***Loss landscapes for varying values of β, for the 1D convection example in §3.1.*** *The loss landscape is more smooth at low β, and it becomes increasingly more complex as β increases, which can make the optimization problem more difficult. In particular, at higher β, the optimizer gets stuck in a certain regime. These results support that adding the PDE soft regularization term results in a more complex optimization loss landscape.*

Krishnapriyan, A., Gholami, A., Zhe, S., Kirby, R., & Mahoney, M. W. (2021). Characterizing possible failure modes in physics-informed neural networks. *Advances in neural information processing systems*, *34*, 26548-26560.

What we need is Hard Constraints

Given a map

$$\hat{f}_\theta : \Phi \to \mathcal{Y}$$

$$\mathbf{Y}_\theta(\phi^{(i)}) \in \mathbb{R}^n$$

$$\hat{u}(\phi^{(i)}) \sim \mathbf{Y}_\theta(\phi^{(i)})$$

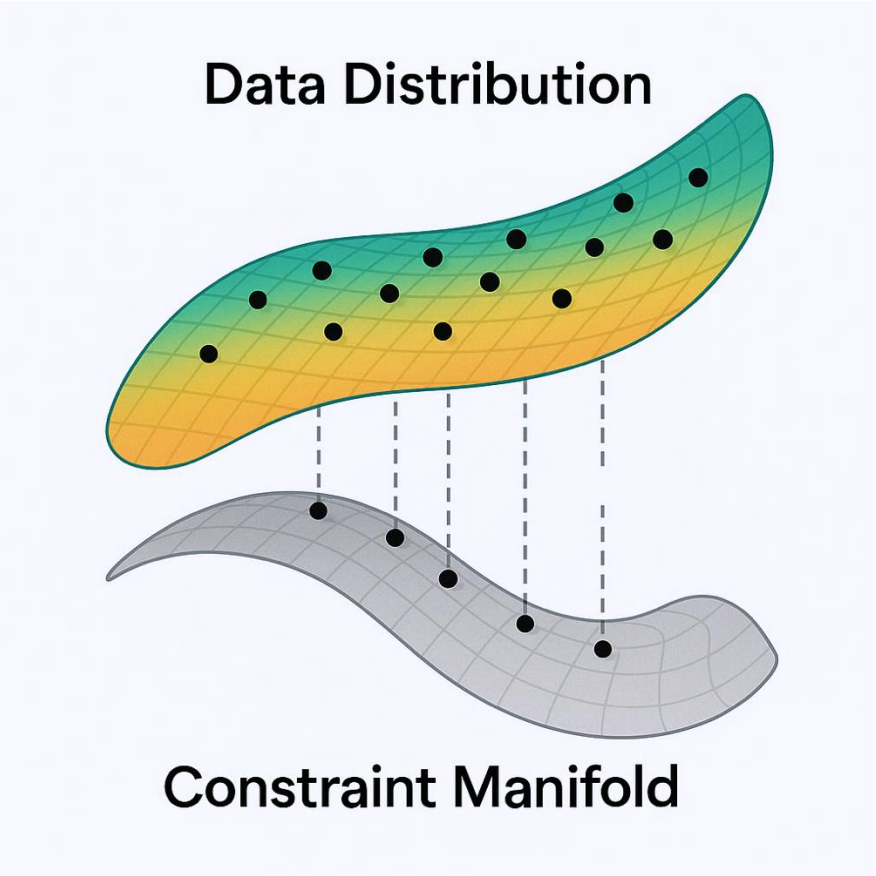
$$h(\hat{u}(\phi^{(i)})) = 0$$

$$g(\hat{u}(\phi^{(i)})) \leq 0$$

Inputs maps to "probabilistic predictions"

Predicted random variable

The distribution should strictly obey some known constraints



Why are Hard constraints “hard”?

Previous methods known to have **limitations.**

Domain Specific

Hinders generalizability and adoption

Simple Constraints

Reality generally has nonlinear constraints

Point Estimates

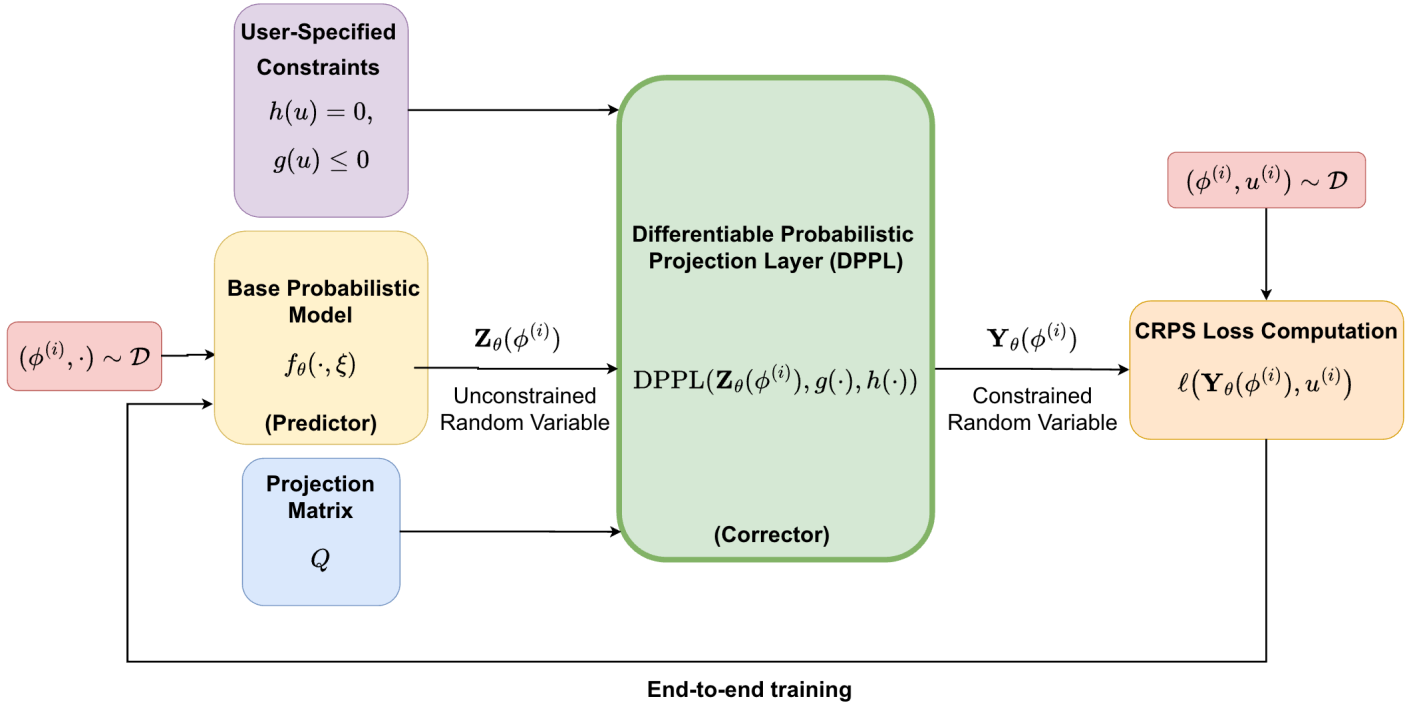
What if I want probabilistic estimates/UQ?

Post-hoc correctio

End-to-end training procedure?

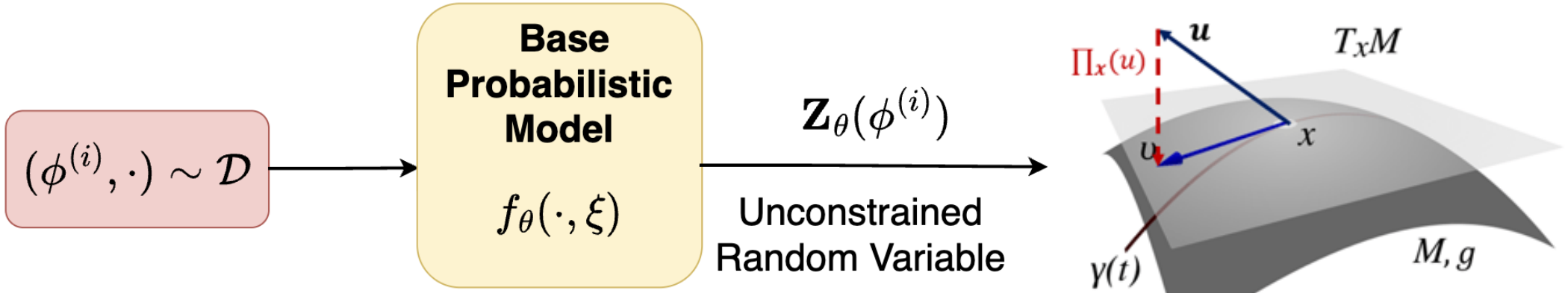
Introducing ProbHardE2E

A Novel **Probabilistic Framework** to impose **Hard Constraints**



- ✔ **Probabilistic by Design**
- ✔ **Nonlinear constraints**
- ✔ **Domain Agnostic**
- ✔ **Efficient Sampling Free Training**

Key Idea: Predict the distribution, Project the distribution



Data sample

A "backbone" probabilistic model

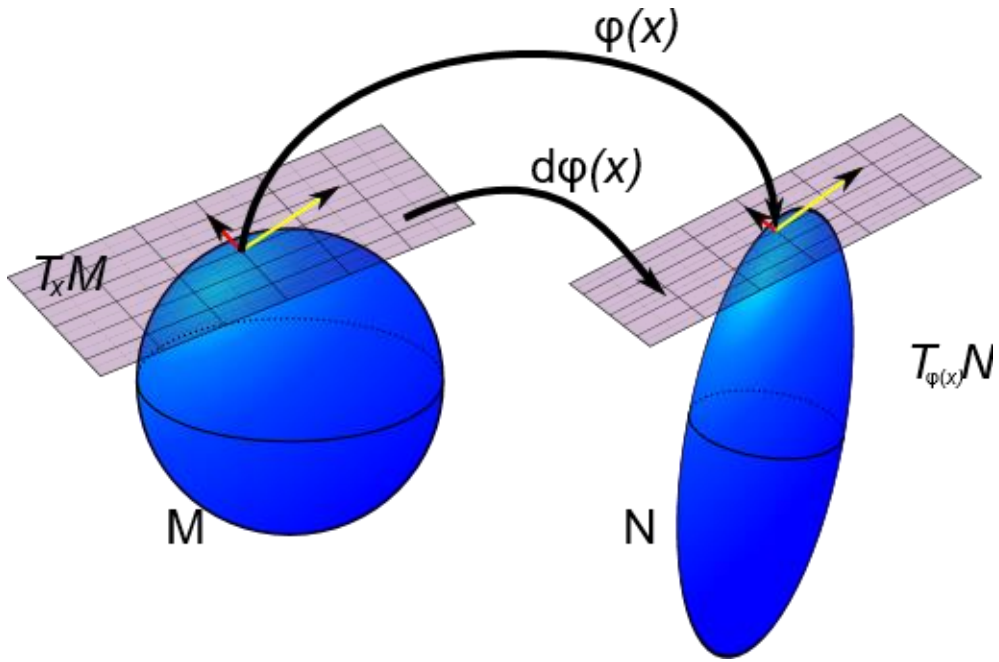
E.g.: Gaussian Processes, Ensemble NNs, DeepAR

Prediction: **Mean and variance**

Project the distribution on the constrained manifold

DPPL: Differentiable Probabilistic Projection Layer

- DPPL generalizes the idea of projection layers to projecting distributions
- DPPL considers projection as a pushforward measure & during training it directly obtains **mean** and **variance** of the projected distribution
- Scalable backpropagation via implicit differentiation



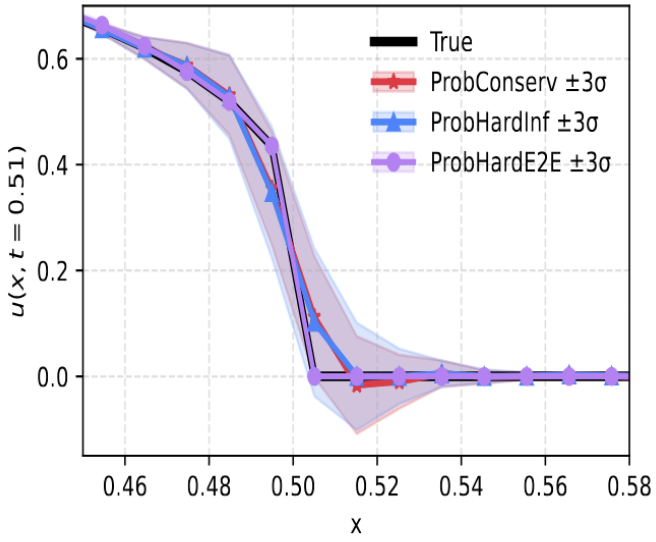
Theorem 3.1. *Let* $\mathbf{Z} \sim \mathcal{F}(\mu, \Sigma)$ *be a random variable, where the underlying distribution* $\mathcal{F}$ *belongs to a multivariate location-scale family of distributions, with mean* μ *and covariance* Σ*; and let* $\mathcal{T}$ *be a function with continuous first derivatives, such that* $J_{\mathcal{T}}(\mu)\Sigma J_{\mathcal{T}}(\mu)^\top$ *is symmetric positive semi-definite. Then, the transformed distribution* $\mathbf{Y} = \mathcal{T}(\mathbf{Z})$ *converges in distribution with first-order accuracy to* $\mathcal{F}(\hat{\mu}, \hat{\Sigma})$ *with mean* $\hat{\mu} = \mathcal{T}(\mu)$ *and covariance* $\hat{\Sigma} = J_{\mathcal{T}}(\mu)\Sigma J_{\mathcal{T}}(\mu)^\top$*, where* $J_{\mathcal{T}}(\mu) = \nabla\mathcal{T}(\mu)^\top$ *denotes the Jacobian of* $\mathcal{T}$ *with respect to* z *evaluated at* μ*.*

DPPL with Different Constraint Types

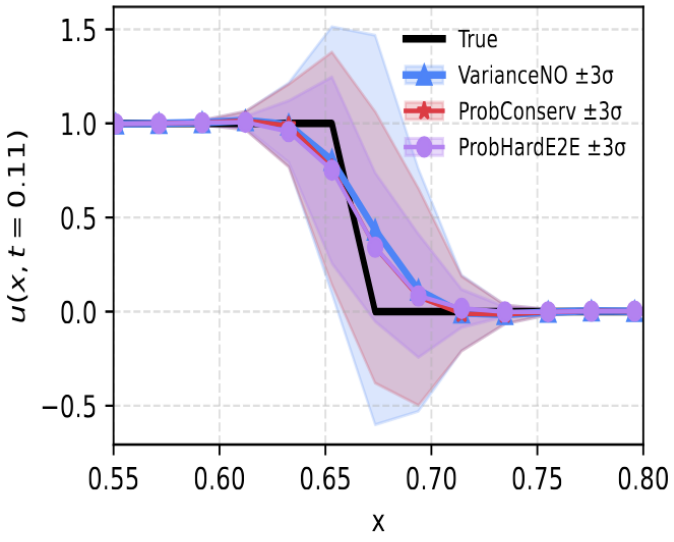
Constraint Type	Solution $u^*(z)$	Solver Type	Jacobian $J_{\mathcal{T}}$
Linear Equality	$P_{Q^{-1}}z + (I - P_{Q^{-1}})A^\dagger b$	closed-form	$P_{Q^{-1}}$
Nonlinear Equality	(u^*, λ^*) s.t. $R(u^*, \lambda^*; z) = 0\}$	nonlinear	implicit differentiation
Convex Inequality	$\underset{h(\hat{u})=0,\, g(\hat{u})\leq 0}{\text{argmin}} \quad \|\hat{u} - z\|_Q^2$	convex opt.	sensitivity analysis; argmin differentiation

DPPL: Probabilistic learning + Differentiable optimization

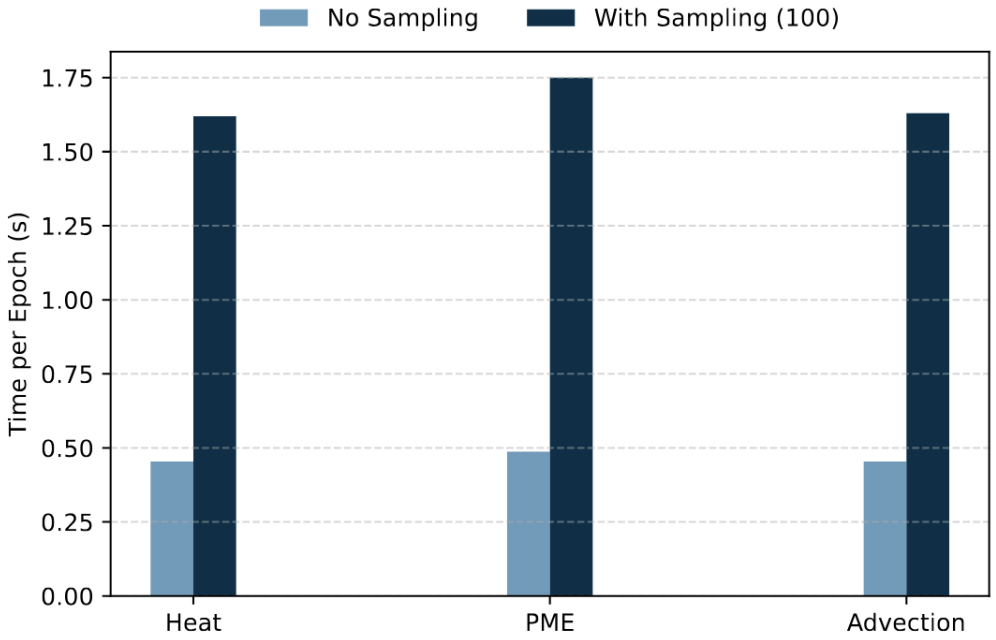
Evaluation on PDEs with conservation laws



(b) Nonlinear Equality: PME



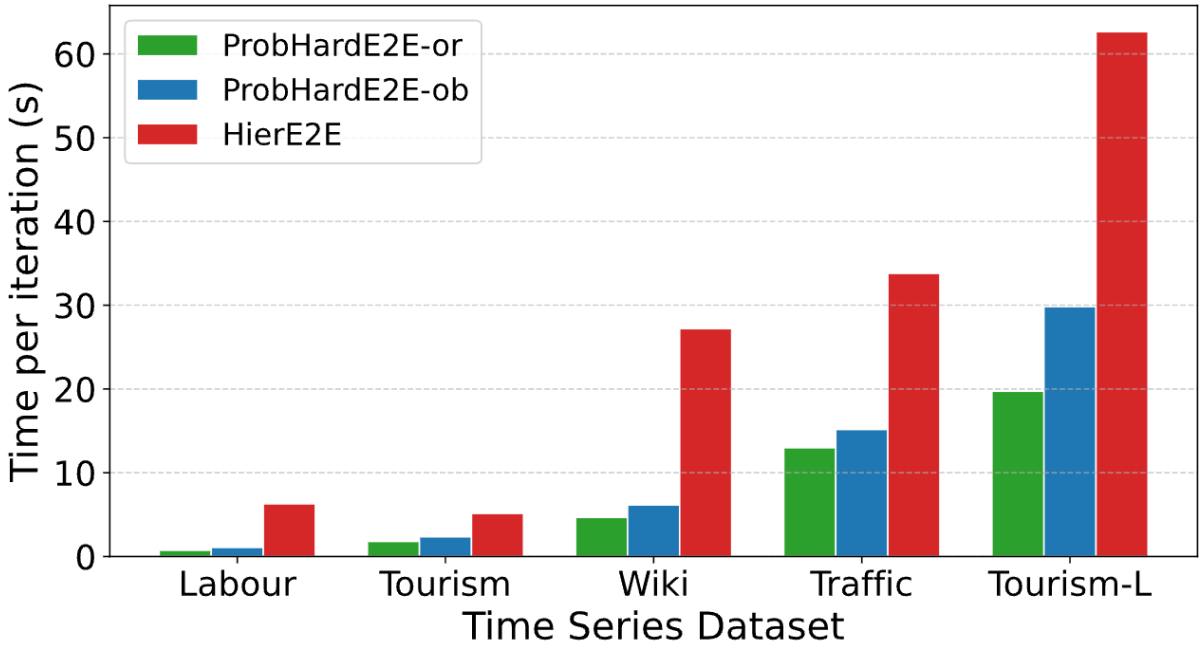
(c) Convex Inequality: TVD



Empirically showcase the value of optimizing a proper-scoring rule instead of likelihood in PDE datasets!

Evaluation on Hierarchical Time-Series Forecasting

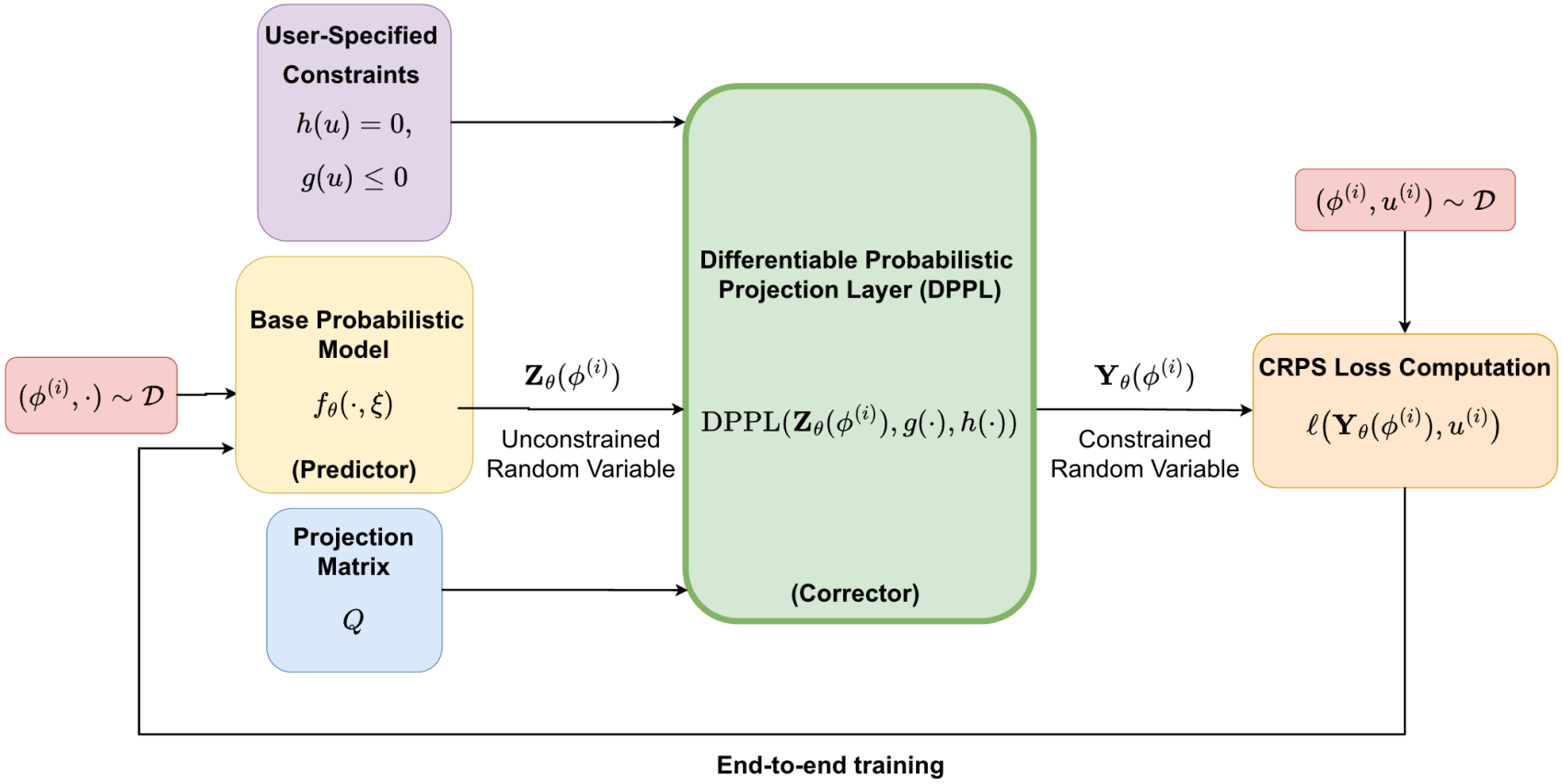
ProbHardE2E not only achieves **superior performance** on hierarchical datasets but is **faster to train** as well!



Dataset	ProbHardE2E-Ob	ProbHardE2E-Or	ProbConserv	HierE2E	ARIMA-NaiveBU	ETS-NaiveBU	PERMBU-MINT	DeepVAR (base)
LABOUR	36.1± 2.7 (0)	**28.6**±6.5 (0)	45.8±6.5 (0)	50.5±20.6 (0)	45.3 (0)	43.2 (0)	39.3 (0)	38.2±4.5 (0.215)
TOURISM	98.9±13.0 (0)	82.4±6.6 (0)	100.7±7.7 (0)	103.1±16.3 (0)	113.8 (0)	100.8 (0)	**77.1** (0)	92.5±2.2 (2818.01)
TOURISM-L	**155.2**±3.6 (0)	156.4±9.4 (0)	176.9±21.5 (0)	161.3±10.9 (0)	174.1 (0)	169.0 (0)	–	158.1±10.2 (70000)
TRAFFIC	55.0±10.6 (0)	60.6±7.8 (0)	71.0±3.9 (0)	**41.8±7.8 (0)**	80.8 (0)	66.5 (0)	67.7 (0)	40.0±2.6 (0.192)
WIKI	**212.1**±29.4 (0)	215.8±16.9 (0)	264.7± 30.7 (0)	216.5±26.7 (0)	377.2 (0)	467.3 (0)	281.2 (0)	229.4±15.8 (8398.6)

Thank You!

Our Algorithm in a Nutshell

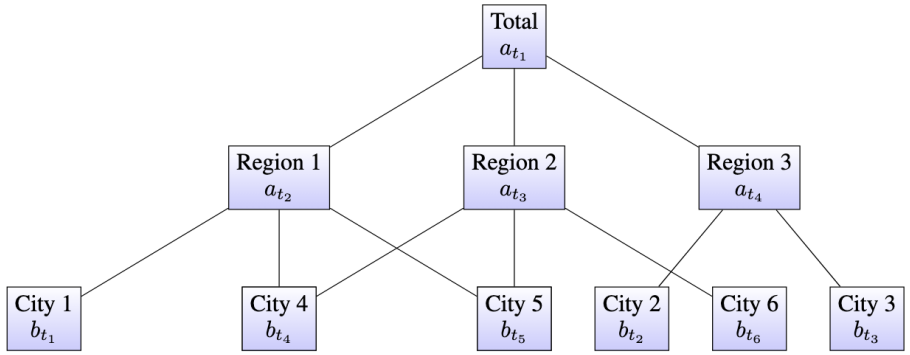


Landscape of ML with Hard Constraints

Method	Domain	Constraint Type	End-to-End	Prob. Model w/ Variance Estimate	Sampling-free Training	Constraint on Distribution
HardNet [Min et.al]	General	Convex	✅	❌	✅	✅
DC3 [Donti et. al]	General	Nonlinear	✅	❌	✅	✅
Hier-E2E [Rangapuram et. al]	Forecasting	Linear	✅	✅	❌	❌
CLOVER [Olivares et al]	Forecasting	Linear	✅	✅	❌	✅
PDE-CL [Négiar et al.]	PDEs	Nonlinear	✅	❌	✅	✅-
ProbConserv [Hansen et al.]	PDEs	Linear	❌	✅	✅	✅
HardC [Hansen et al.]	PDEs	Linear	❌	✅	✅	✅
ProbHardE2E	General	Nonlinear	✅	✅	✅	✅

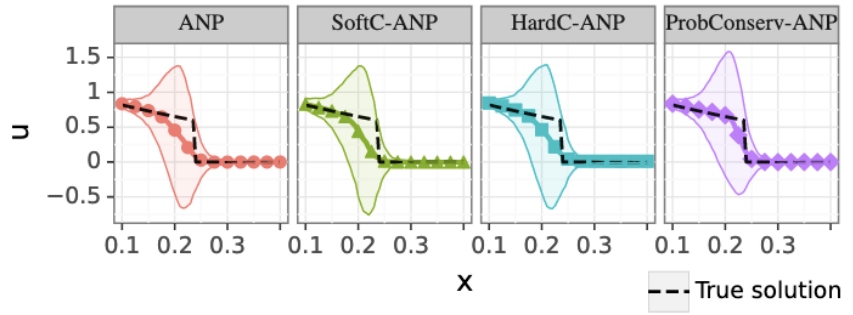
Extra Slides

Domain specific developments for hard constraints

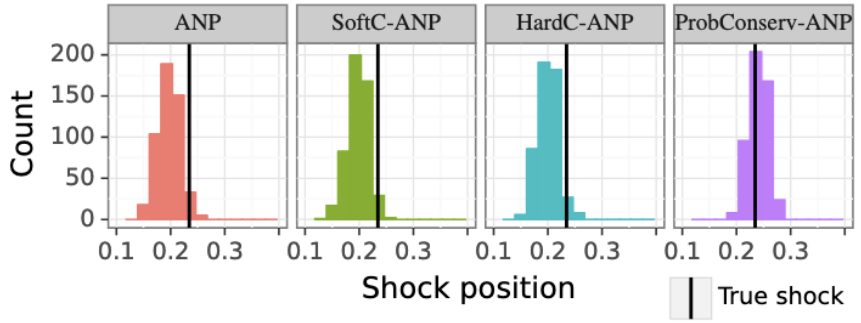


Hierarchical time-series

Ensuring strict adherence to invariances to conservation laws



(a) Solution profile.



(b) Posterior of the shock position.

PDEs with global conservation laws

Rangapuram, Syama Sundar, et al. "End-to-end learning of coherent probabilistic forecasts for hierarchical time series." *International Conference on Machine Learning*. PMLR, 2021

Hansen, Derek, et al. "Learning physical models that can respect conservation laws." International Conference on Machine Learning. PMLR, 2023.

Our Three “Prongs”

Modeling Framework

General abstractions on modeling framework.

Theoretical similarities between framework, although domain specific treatment of hard constraints and UQ.

End-to-End

Replication of ideas of E2E in time-series work to PDEs.

Model agnostic adaption in the training pipeline.

Hard Constraints

Shocks in the PDE (Advection equations) is empirically similar to spikes in time-series (Prime day sale forecasts)

Regularization: Standard Trick

$$\underset{\theta\in\Theta,\ g(\mathbf{Y}_\theta(\phi^{(i)}))\leq 0,\ h(\mathbf{Y}_\theta(\phi^{(i)}))=0}{\arg\min} \mathbb{E}_{(\phi^{(i)},u^{(i)})\sim\mathcal{D}}\ \ell\big(\mathbf{Y}_\theta(\phi^{(i)}), u^{(i)}\big)$$

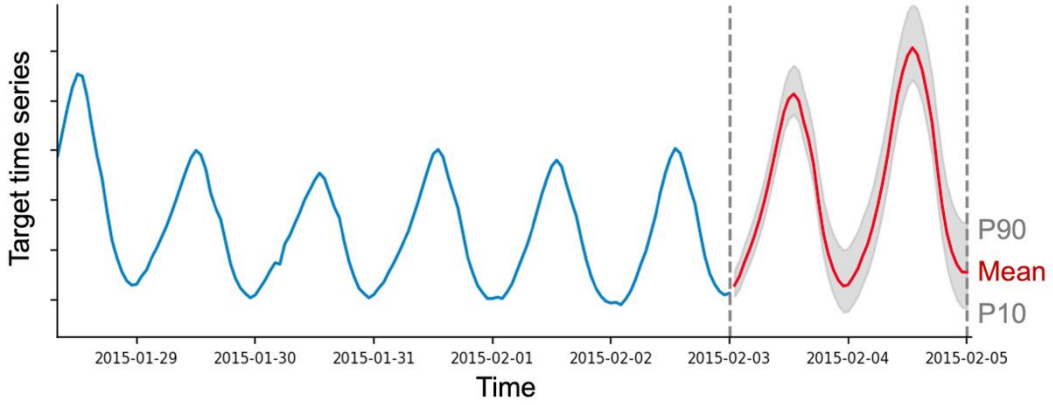
Training Objective

$$\arg\min_{\theta\in\Theta} \mathbb{E}_{(\phi^{(i)},u^{(i)})\sim\mathcal{D}}\ \ell\big(\mathbf{Y}_\theta(\phi^{(i)}), u^{(i)}\big) + \lambda_g \left\|\mathrm{ReLU}(g(\mathbf{Y}_\theta(\phi^{(i)}))\right\|_2^2 + \lambda_h \left\|h(\mathbf{Y}_\theta(\phi^{(i)}))\right\|_2^2$$

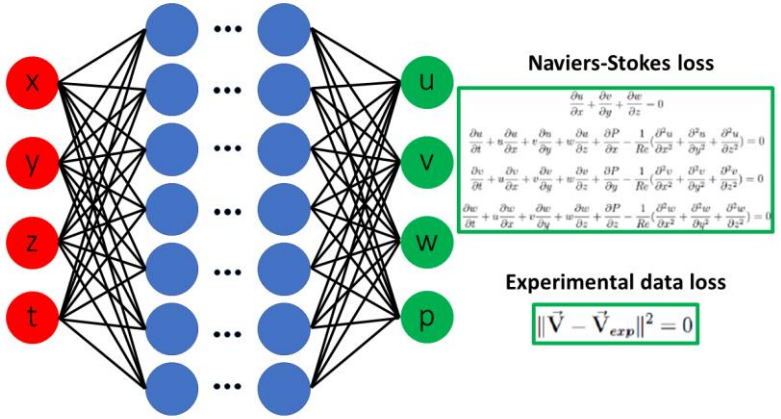
Regularized Training Objective

Is this sufficient to enforce strict constraint satisfaction?

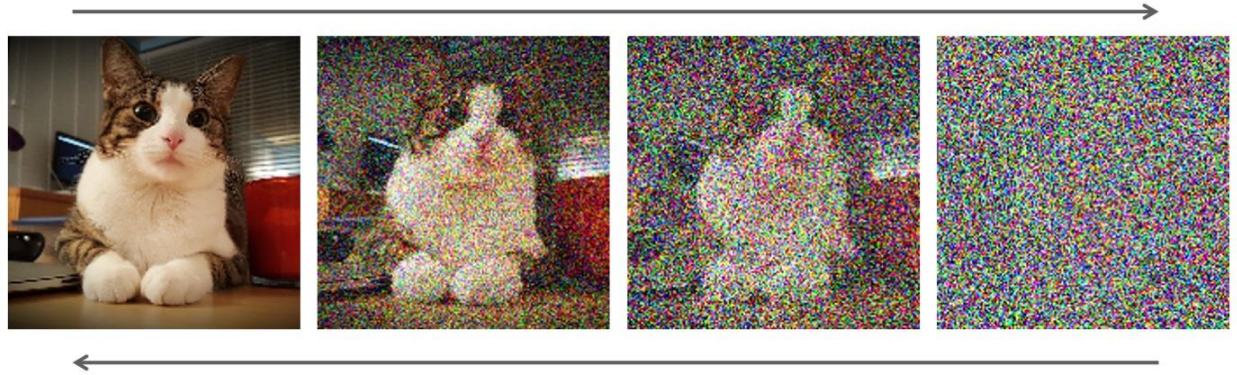
ML has come too far to provide domain-specific solutions



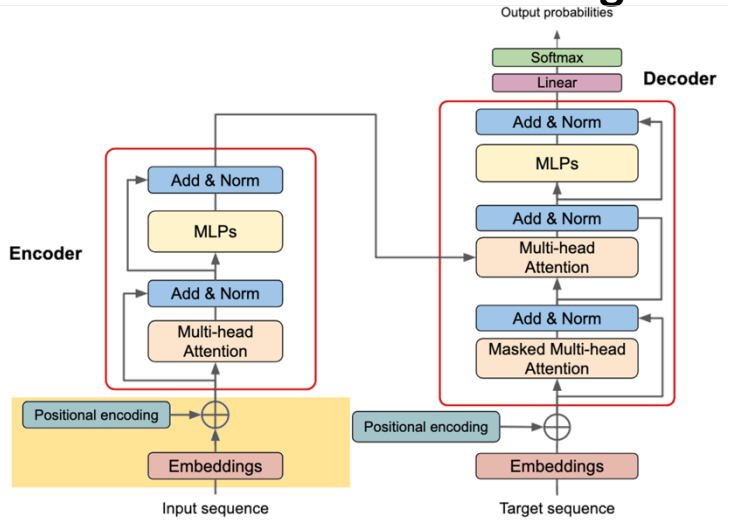
Probabilistic Time Series Forecasting



Physics informed Machine Learning

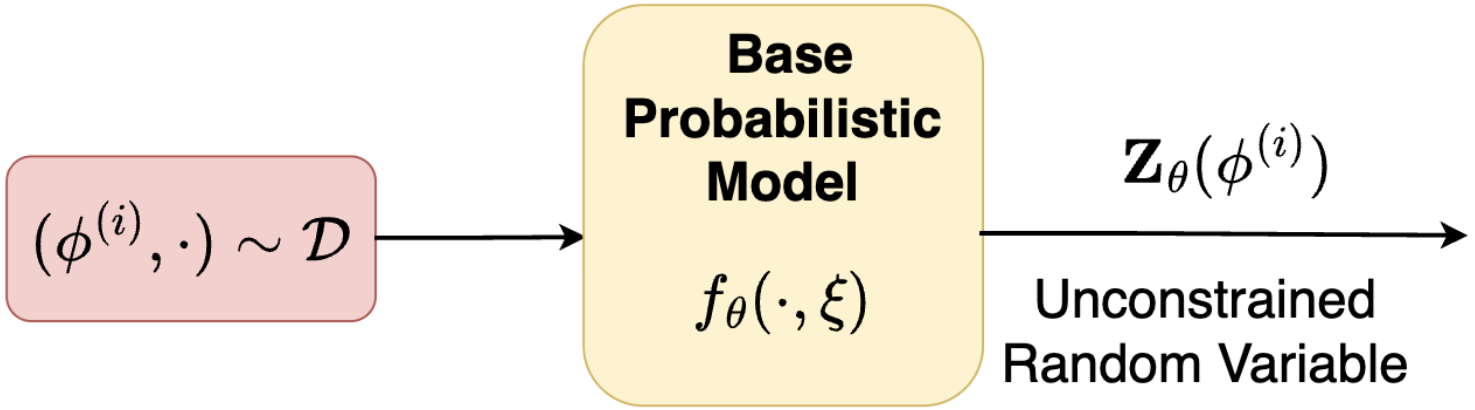


Generative Modeling



Large Language Models

Key Idea: Predict the distribution, Project the distribution



Data sample

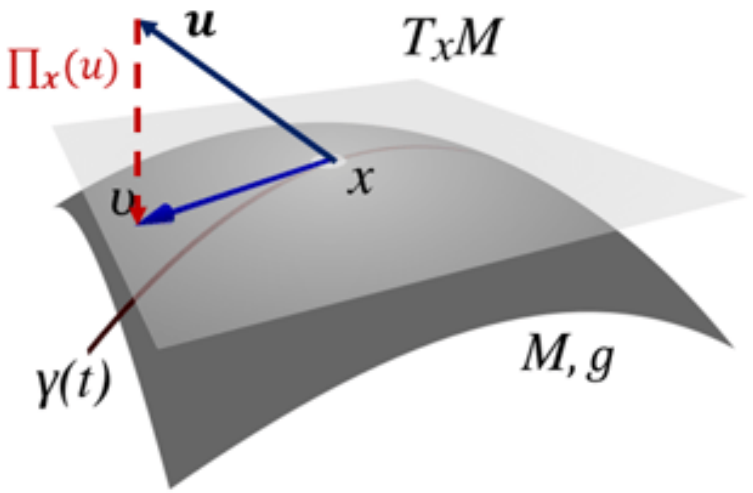
A "backbone" probabilistic model

E.g.: Gaussian Processes, Ensemble NNs, DeepAR

Prediction: **Mean and variance**

Unconstrained

Generalizing Projecting Distribution on a Manifold



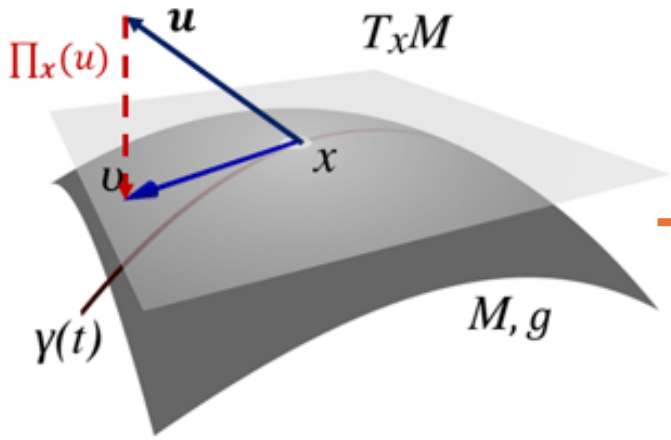
$$u^*(z_\theta(\phi^{(i)})) := \underset{\hat{u}_\theta(\phi^{(i)})\in\mathbb{R}^n, g(\hat{u}_\theta(\phi^{(i)}))\leq 0, h(\hat{u}_\theta(\phi^{(i)}))=0}{\arg\min} \|\hat{u}_\theta(\phi^{(i)}) - z_\theta(\phi^{(i)})\|_Q^2,$$

Essentially, projecting every drawn sample via constrained optimization formulation

This can be prohibitively expensive during training and scales poorly with no. of samples!

DPPL for Location Scale Distributions

$$\mathbf{Z} \sim \mathcal{F}(\mu, \Sigma)$$



$$\mathbf{Y} \sim \mathcal{F}(\hat{\mu}, \hat{\Sigma})$$

$\mathcal{F}$ **needs to be a location-scale distribution (Gaussian, logistic, student's t, etc.)**

$$h(\cdot) = 0$$

$$g(\cdot) \leq 0$$

$$\hat{\mu} = \mathcal{T}(\mu),$$
$$\hat{\Sigma} = J_{\mathcal{T}}(\mu)\Sigma J_{\mathcal{T}}(\mu)^{\top}.$$

Complete sampling free procedure!