Foundations of Differentiable Programming in Probabilistic Models

Jiaxin Shi

Microsoft Research New England

Differentiable Programming

Differentiable Programming

- a new paradigm that has become popular in machine learning, especially in deep learning

Differentiable Programming

- a new paradigm that has become popular in machine learning, especially in deep learning
- (over-)parameterized models trained in an end-to-end fashion to minimize a loss function

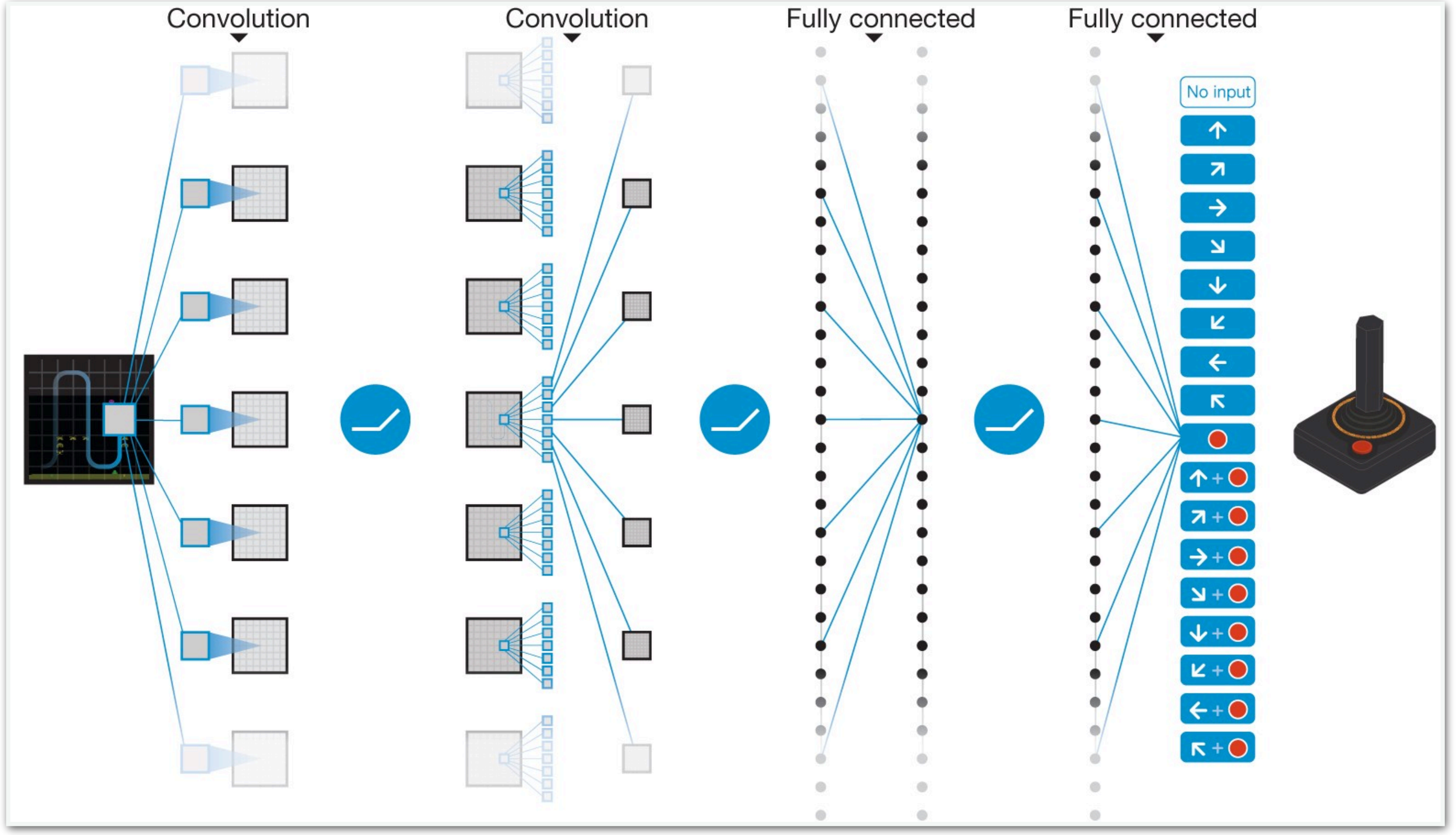
Differentiable Programming

- a new paradigm that has become popular in machine learning, especially in deep learning
- (over-)parameterized models trained in an end-to-end fashion to minimize a loss function
- Models are differentiable. Training is through gradient-based optimization.

Why Differentiable Programming?

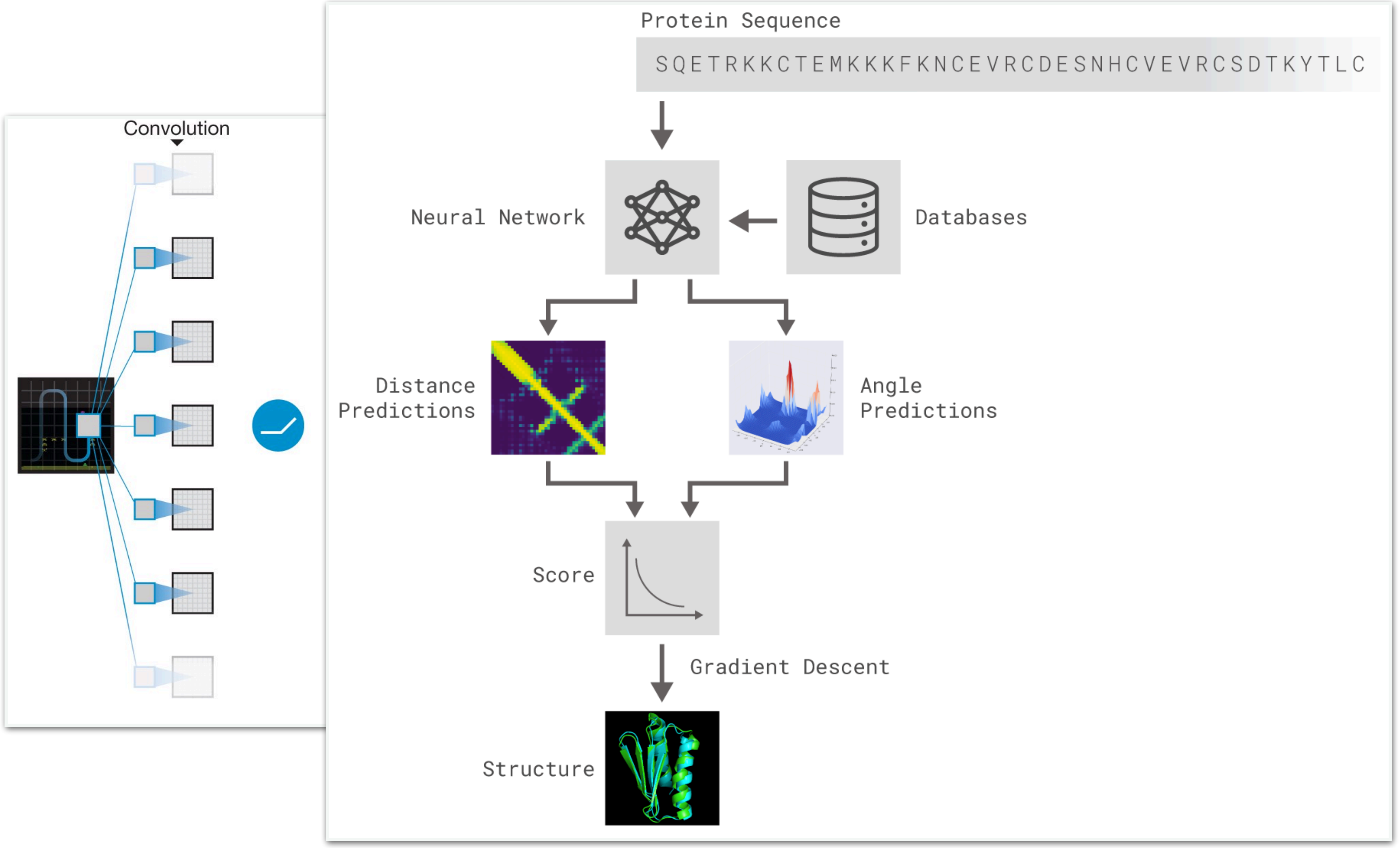
Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?



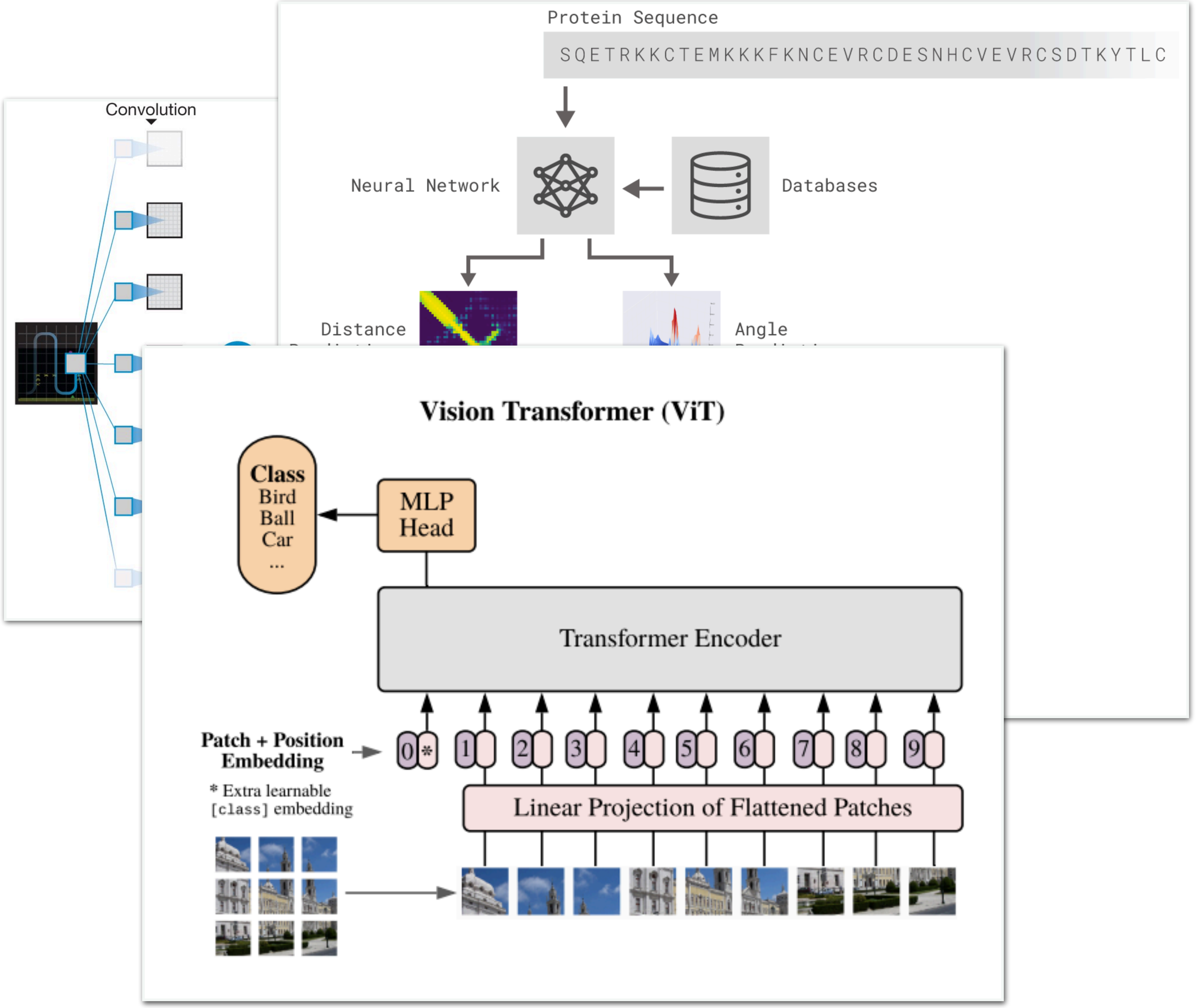
Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?



Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

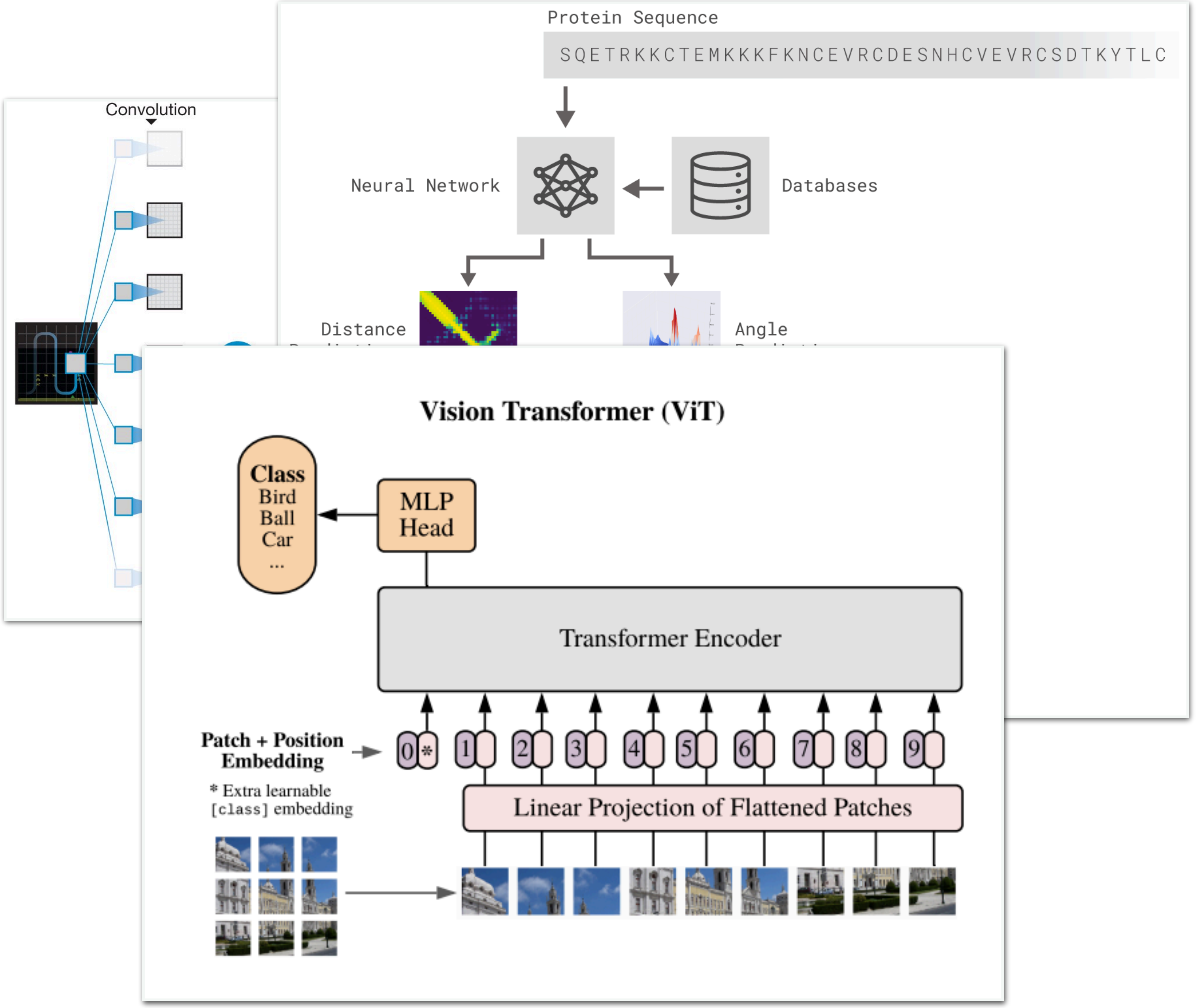
Why Differentiable Programming?



Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?

- Expressiveness

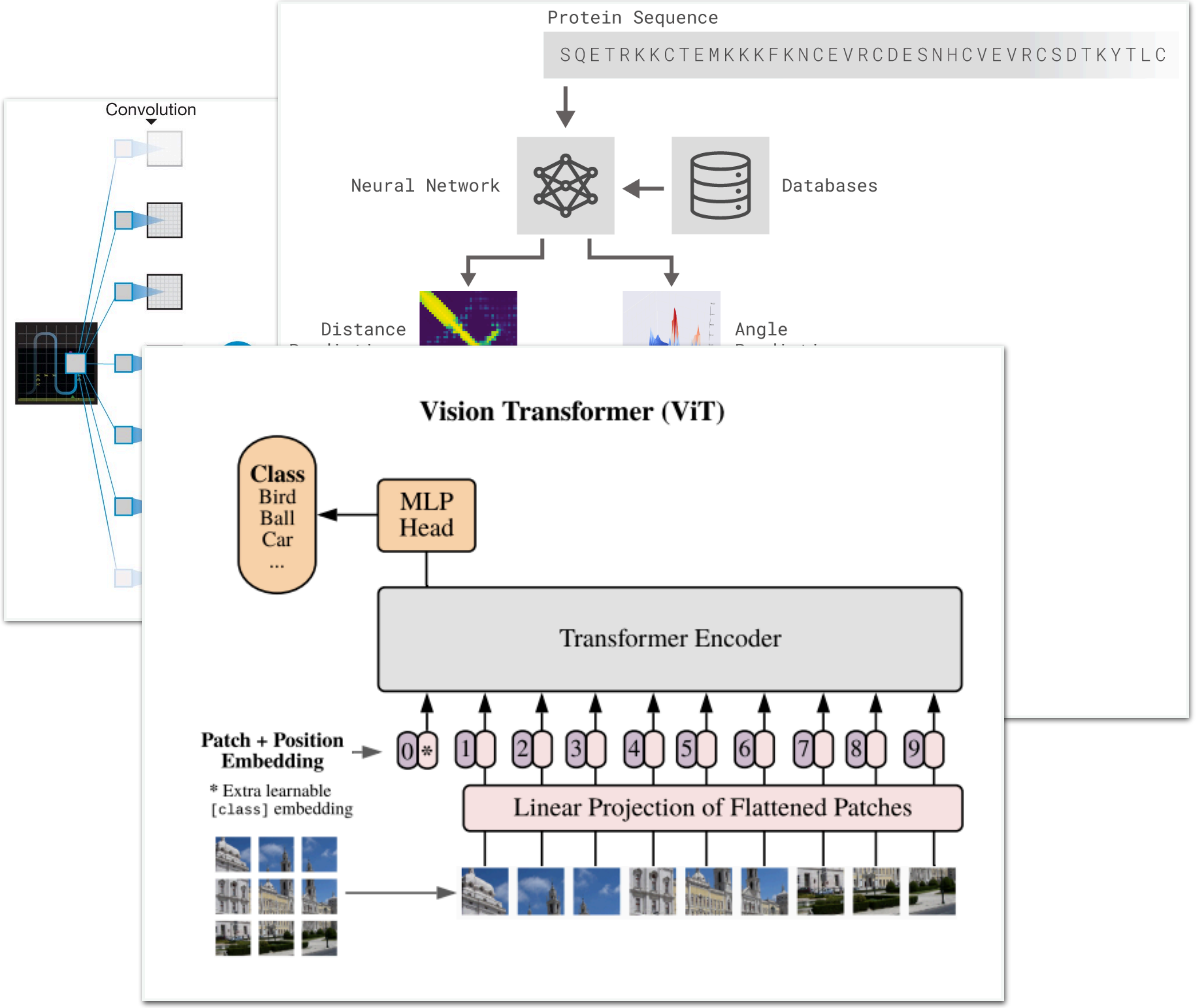


Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?

- Expressiveness

rich enough to express complex mechanisms



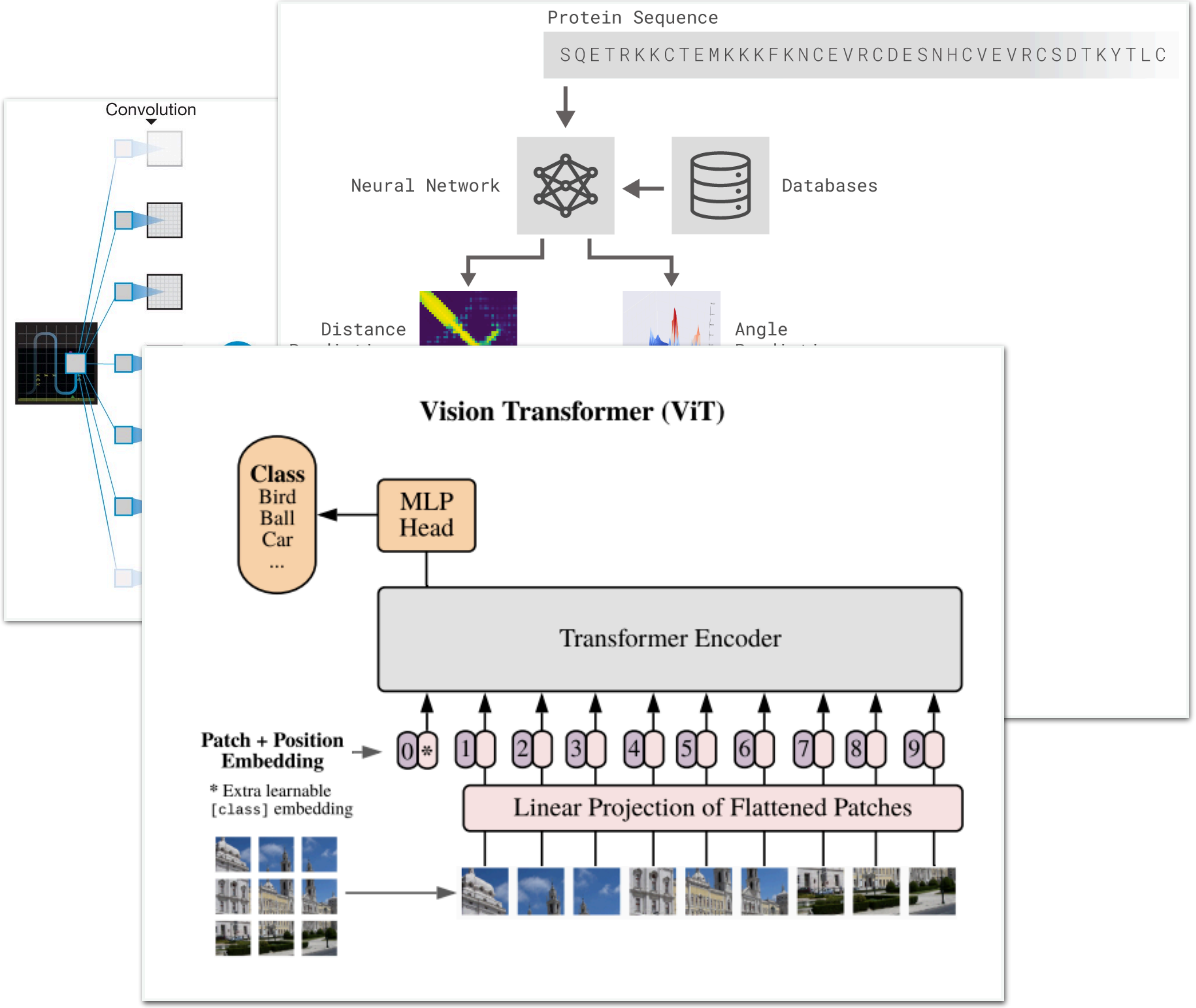
Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?

- Expressiveness

rich enough to express complex mechanisms

- Compositionality



Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

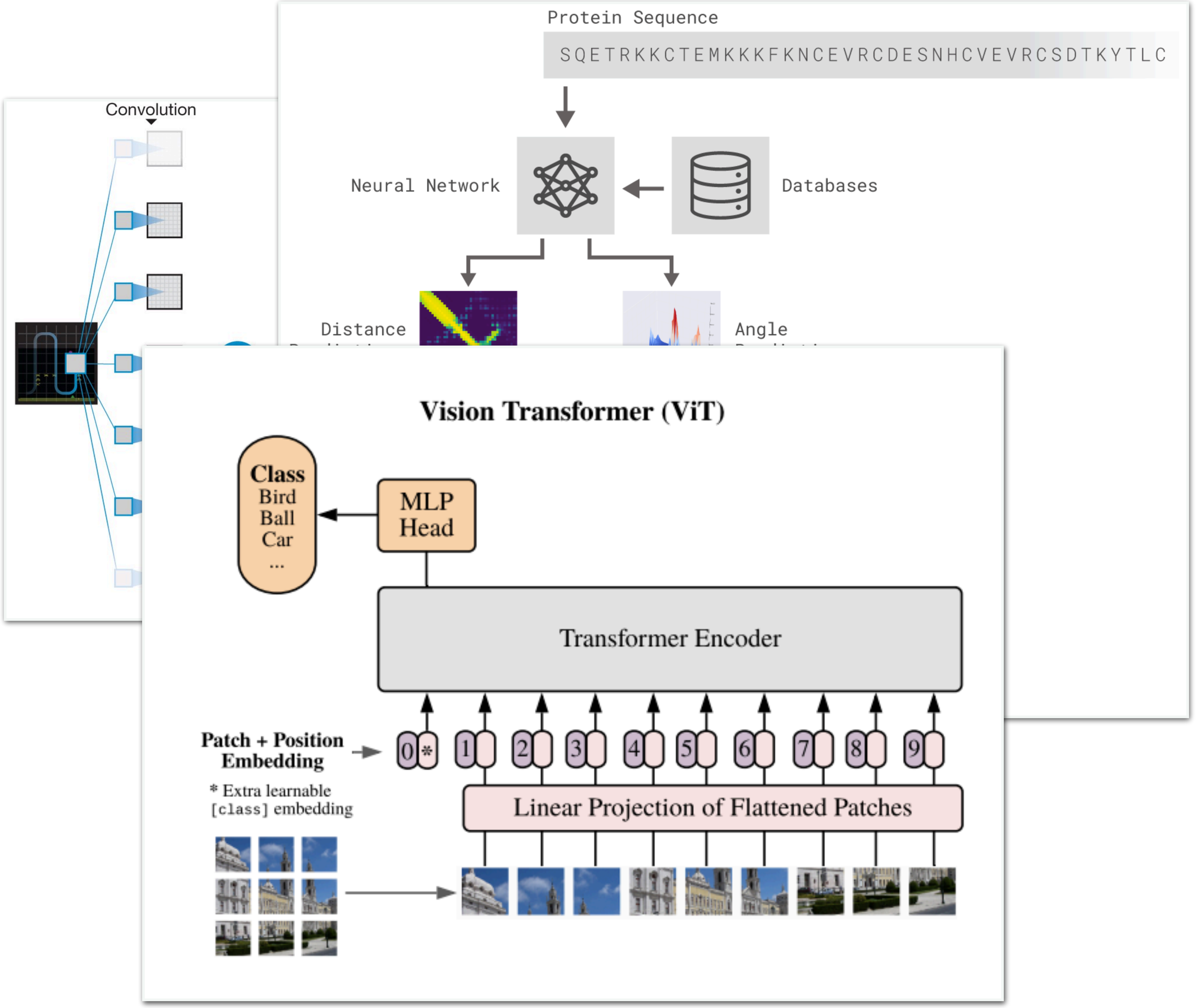
Why Differentiable Programming?

- Expressiveness

rich enough to express complex mechanisms

- Compositionality

Models are easily composable to allow end-to-end training



Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?

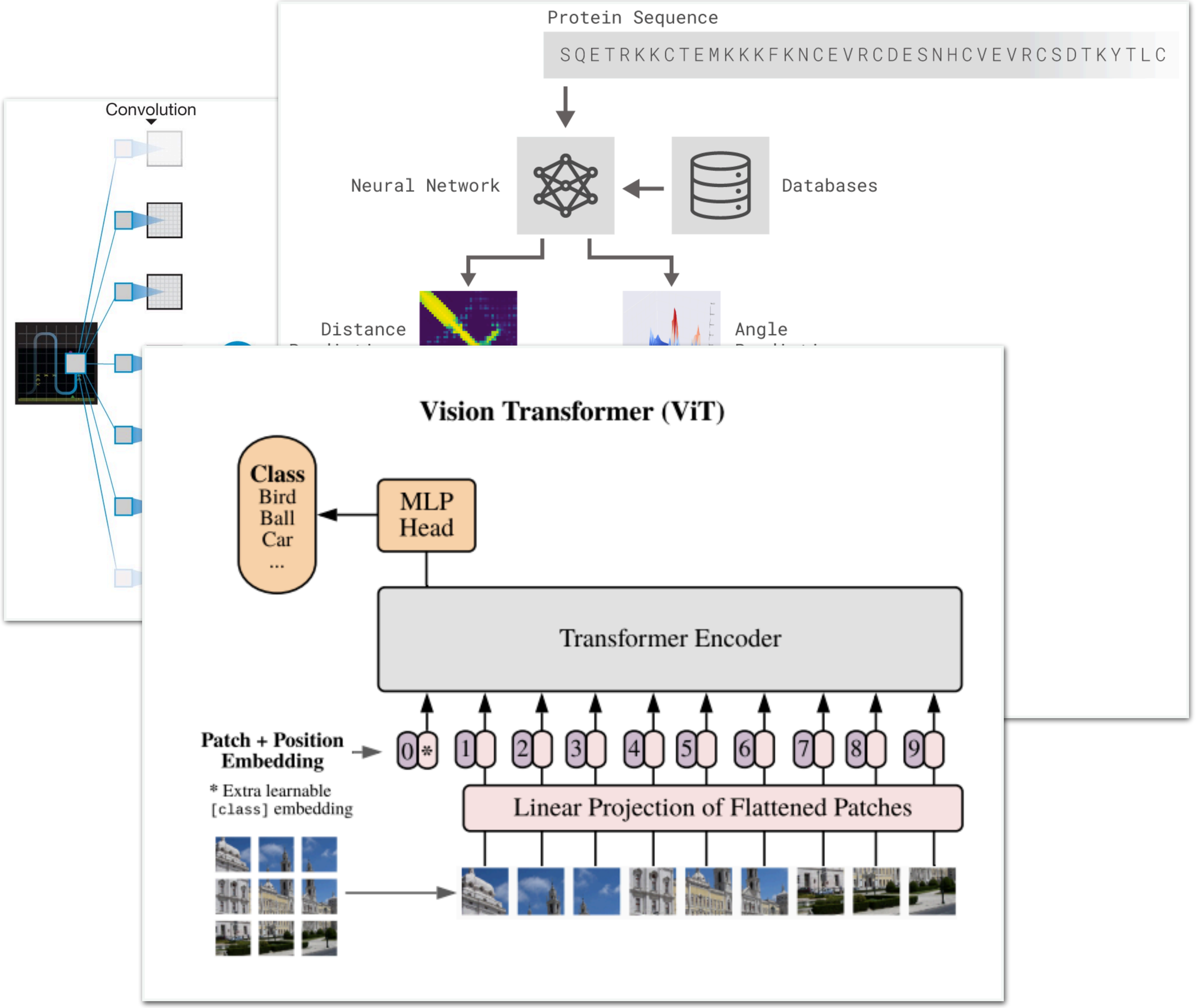
- Expressiveness

rich enough to express complex mechanisms

- Compositionality

Models are easily composable to allow end-to-end training

- Scalability



Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Why Differentiable Programming?

- Expressiveness

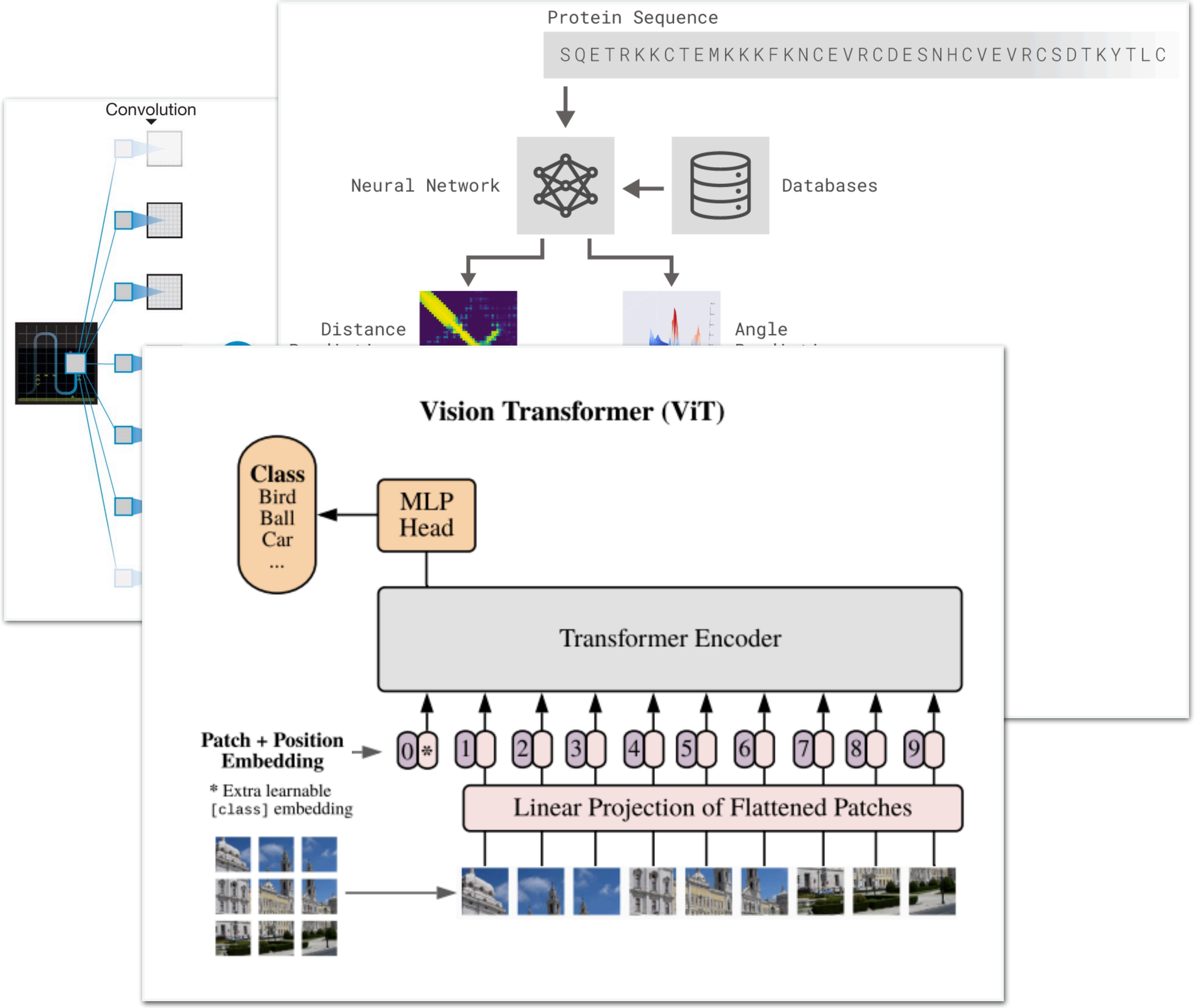
rich enough to express complex mechanisms

- Compositionality

Models are easily composable to allow end-to-end training

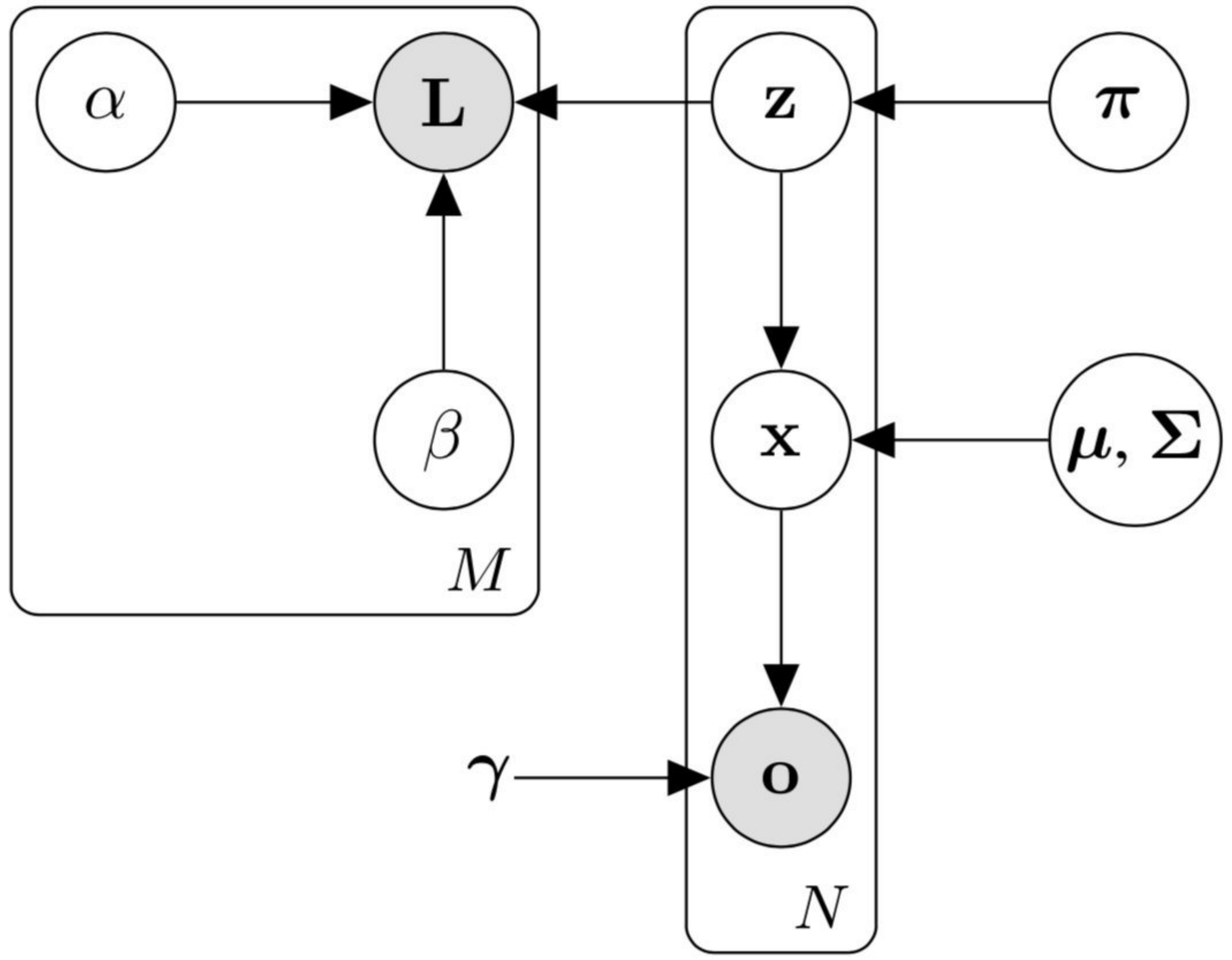
- Scalability

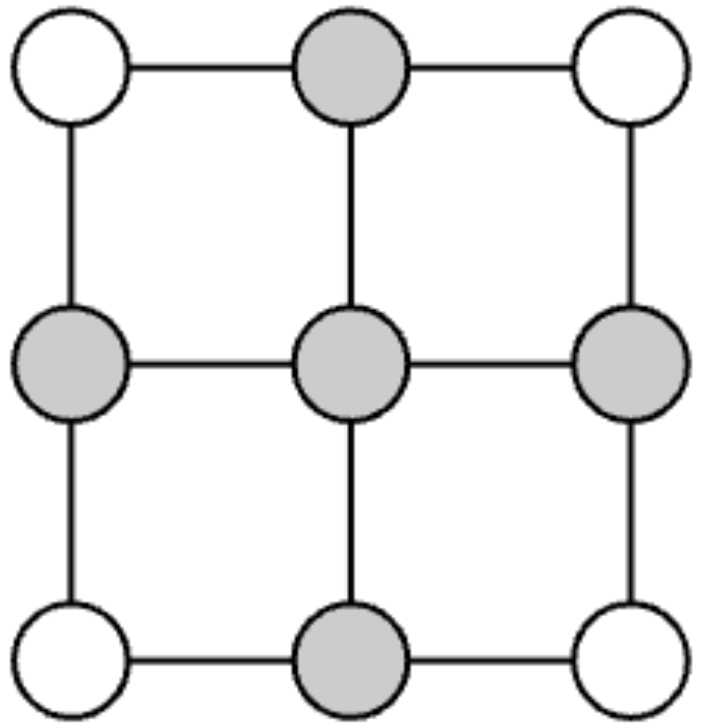
scales to high-dimensional inputs and huge datasets on modern hardware

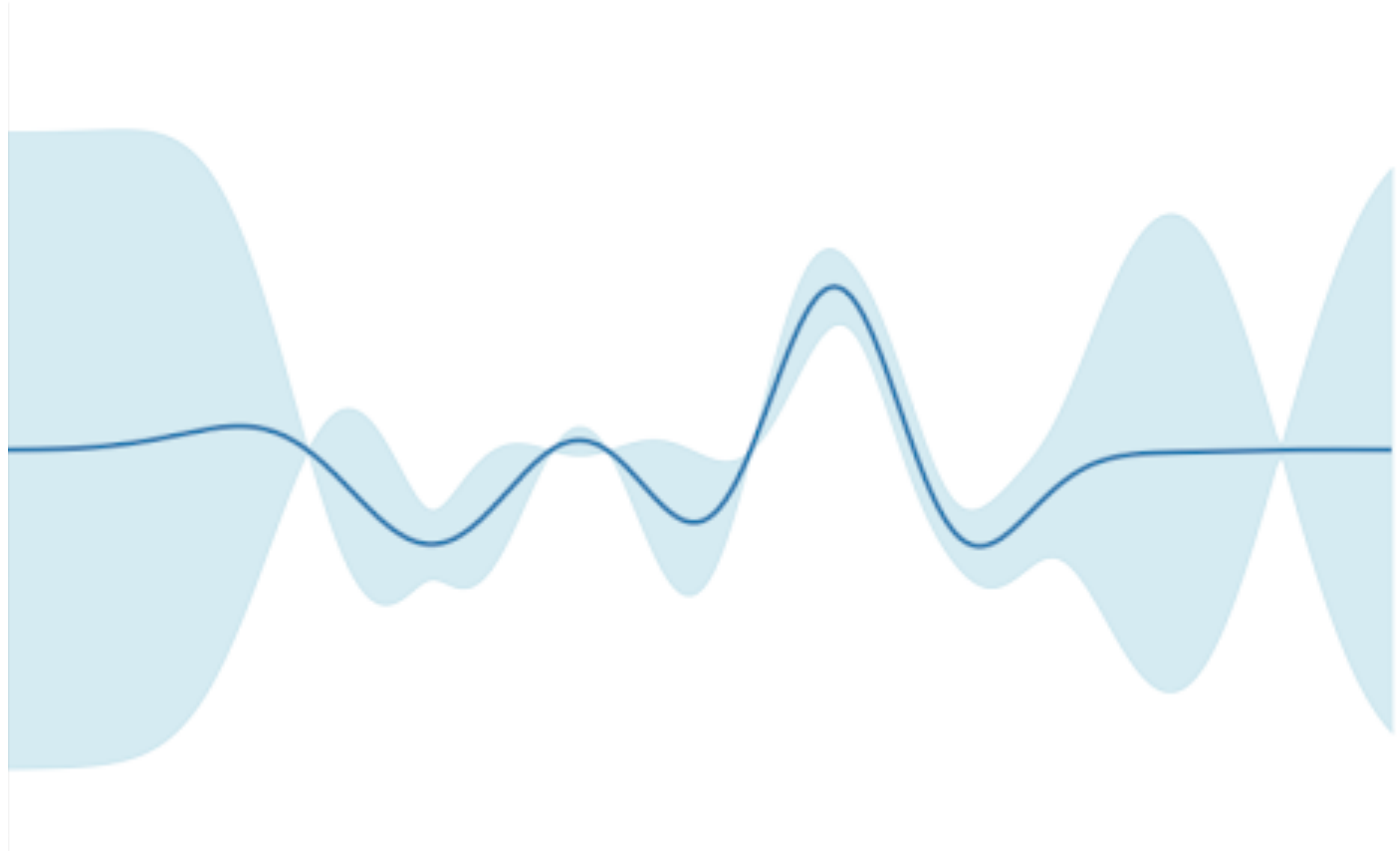


Mnih et al. 13; Jumper et al. 21; Dosovitskiy et al. 20

Probabilistic Modeling







Luo, Tian, **Shi,** Zhu & Zhang (NeurIPS'18)
Zhuo, Liu, **Shi**, Chen, Zhu & Zhang (ICML'18)
Shi, Titsias & Mnih (AISTATS'20)

Why Differentiable Programming in Probabilistic Models?

Why Differentiable Programming in Probabilistic Models?

- Classical probabilistic models in our toolbox (e.g., linear regression, conjugate graphical models) can be significantly misspecified.

Why Differentiable Programming in Probabilistic Models?

- Classical probabilistic models in our toolbox (e.g., linear regression, conjugate graphical models) can be significantly misspecified.

(the commitment to such models) "has led to irrelevant theory, questionable conclusions, and has kept statisticians from working on a large range of interesting current problems"

Statistical Science
2001, Vol. 16, No. 3, 199–231

Statistical Modeling: The Two Cultures

Leo Breiman

Abstract. There are two cultures in the use of statistical modeling to reach conclusions from data. One assumes that the data are generated by a given stochastic data model. The other uses algorithmic models and treats the data mechanism as unknown. The statistical community has

Why Differentiable Programming in Probabilistic Models?

- Classical probabilistic models in our toolbox (e.g., linear regression, conjugate graphical models) can be significantly misspecified.

(the commitment to such models) "has led to irrelevant theory, questionable conclusions, and has kept statisticians from working on a large range of interesting current problems"

(algorithmic model like neural networks) "can produce more reliable information about the structure of the relationship between inputs and outputs than data models"

Statistical Science
2001, Vol. 16, No. 3, 199–231

Statistical Modeling: The Two Cultures

Leo Breiman

Abstract. There are two cultures in the use of statistical modeling to reach conclusions from data. One assumes that the data are generated by a given stochastic data model. The other uses algorithmic models and treats the data mechanism as unknown. The statistical community has

Why Differentiable Programming in Probabilistic Models?

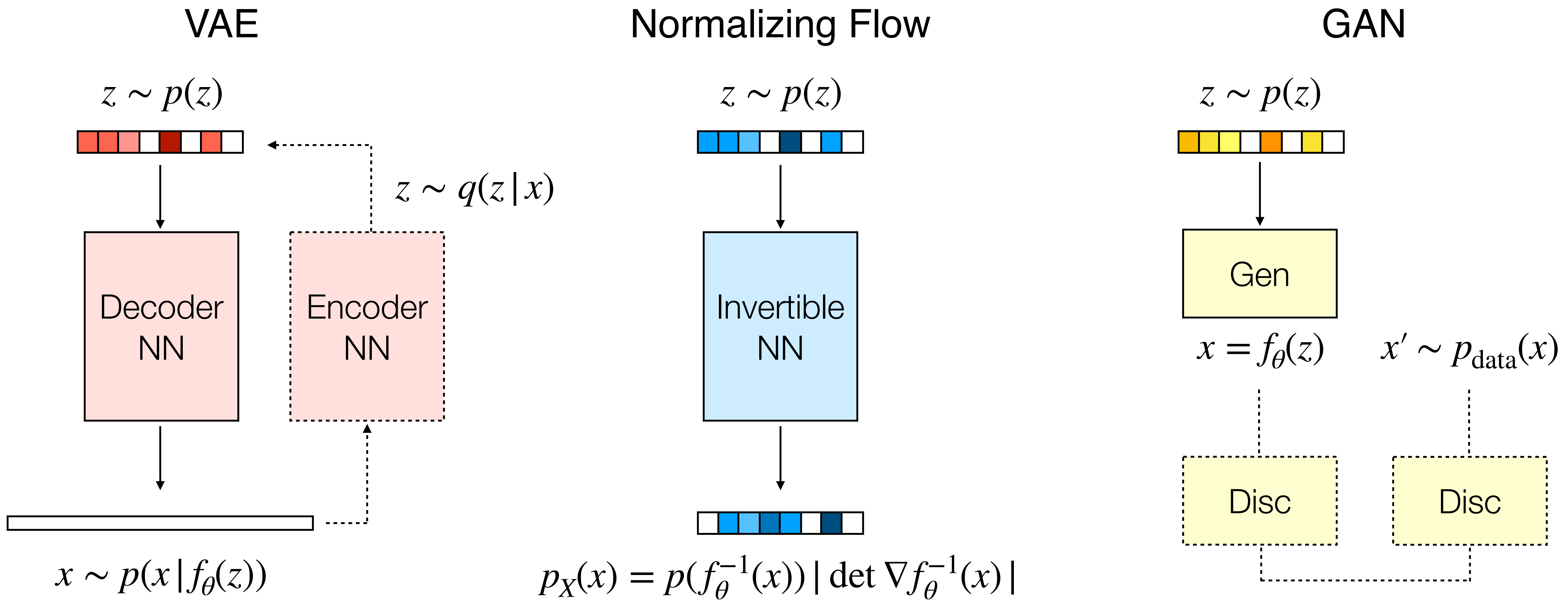
- Classical probabilistic models in our toolbox (e.g., linear regression, conjugate graphical models) can be significantly misspecified.

(the commitment to such models) "has led to irrelevant theory, questionable conclusions, and has kept statisticians from working on a large range of interesting current problems"

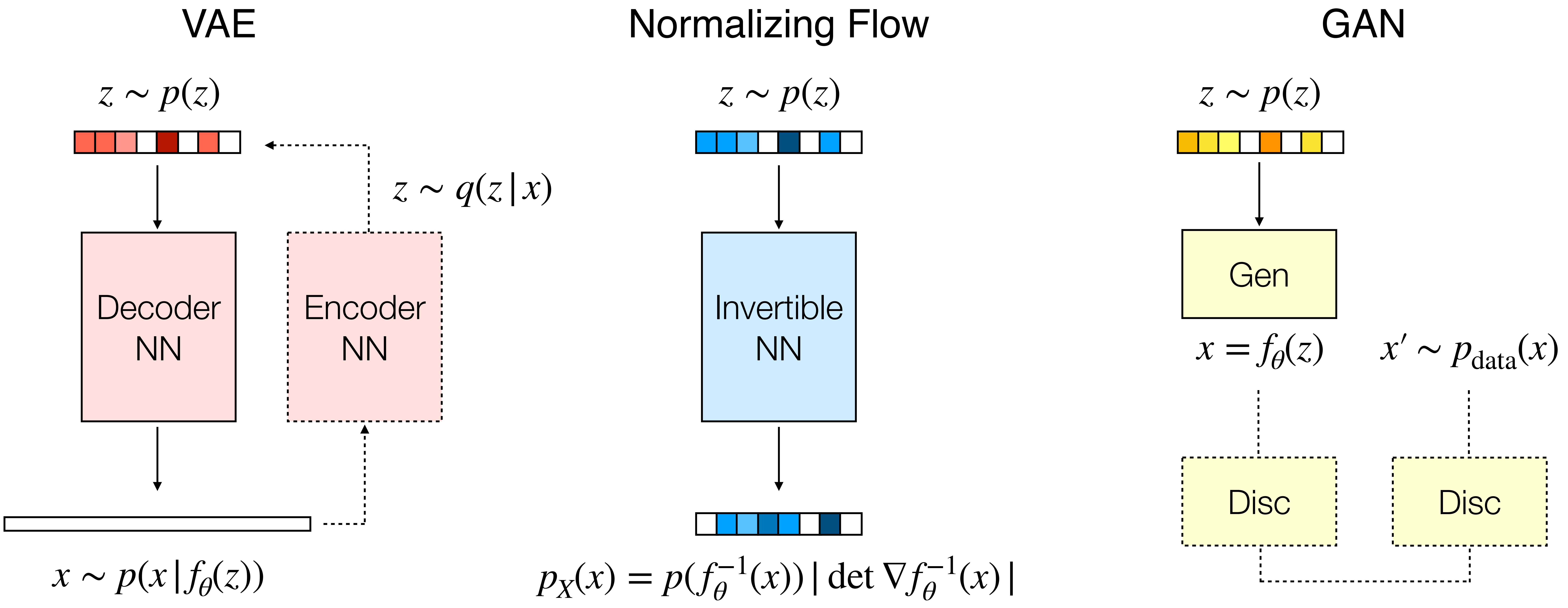
(algorithmic model like neural networks) "can produce more reliable information about the structure of the relationship between inputs and outputs than data models"

- Use differentiable programming ideas to improve the situation?

Existing Attempts are Highly Model-Specific

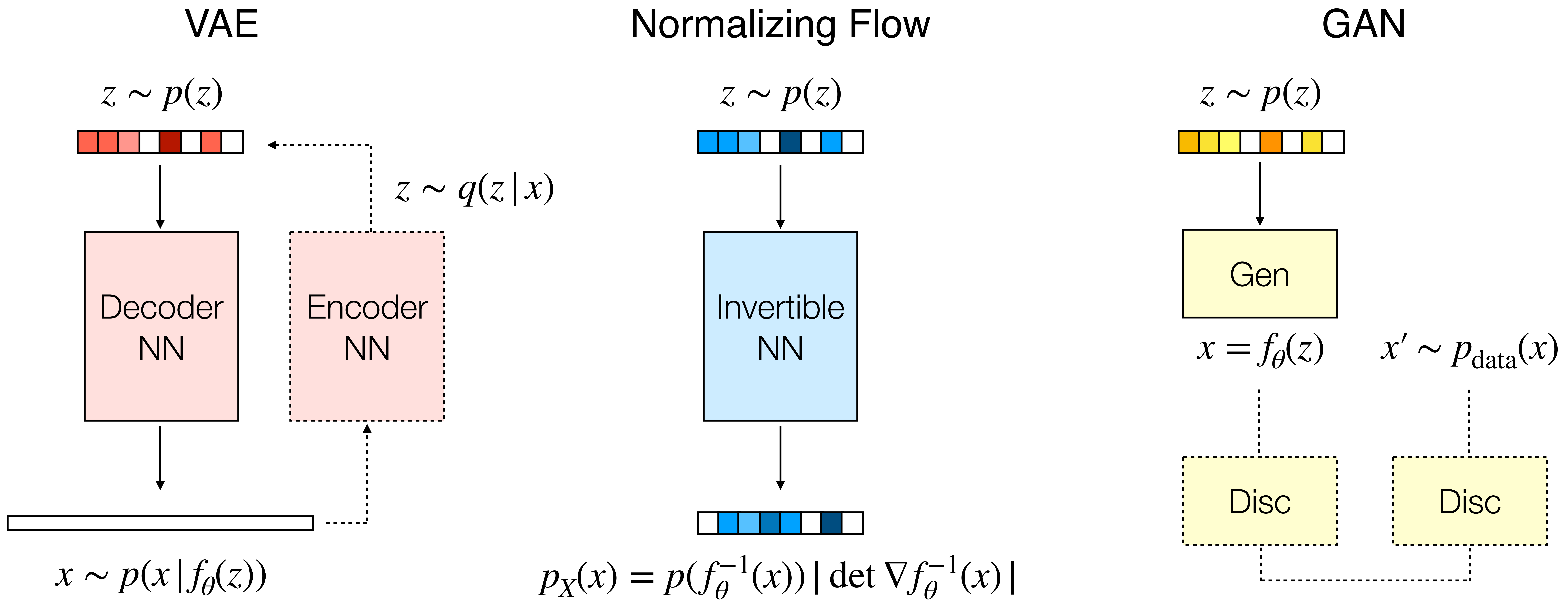


Existing Attempts are Highly Model-Specific



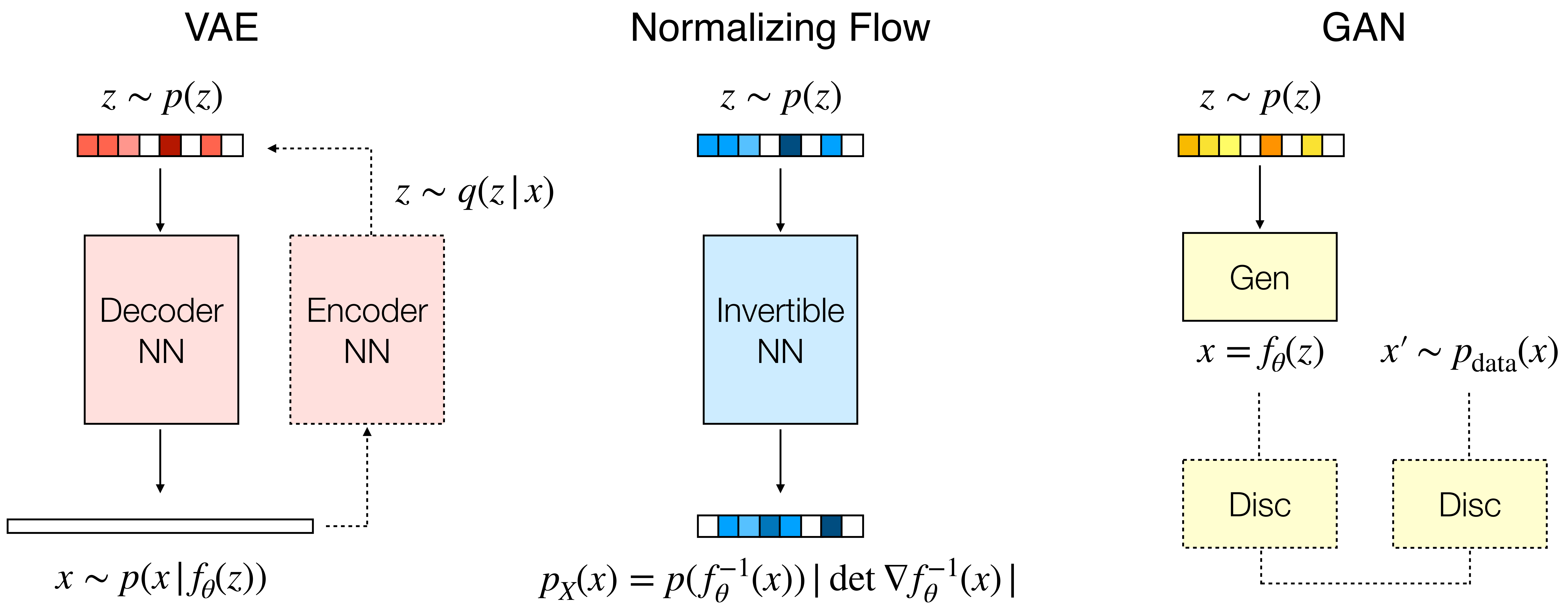
Lack of a unifying framework—algorithms are tailored to certain model class and configurations.

Existing Attempts are Highly Model-Specific



Lack of a unifying framework—algorithms are tailored to certain model class and configurations. **Ideal: One algorithm for all models**

Existing Attempts are Highly Model-Specific



Lack of a unifying framework—algorithms are tailored to certain model class and configurations. **Ideal: One algorithm for all models** **Real: One model for all tasks**

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

a differentiable program

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

a differentiable program

It appears everywhere:

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

a differentiable program

It appears everywhere:

- fitting models to data by minimizing expected loss

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

a differentiable program

It appears everywhere:

- fitting models to data by minimizing expected loss
- optimizing variational objectives

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

a differentiable program

It appears everywhere:

- fitting models to data by minimizing expected loss
- optimizing variational objectives
- computing policy gradients for model-based reinforcement learning

Why is Differentiable Programming in Probabilistic Models Challenging?

Stochastic Gradient Estimation is Difficult

loss function

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))]$$

a differentiable program

It appears everywhere:

- fitting models to data by minimizing expected loss
- optimizing variational objectives
- computing policy gradients for model-based reinforcement learning
- …

Two Levels of Intractability

$$Y \longleftarrow \boxed{\quad f_\theta \quad} \longleftarrow X \sim P_\phi$$

Two Levels of Intractability

$$Y \longleftarrow \boxed{f_\theta} \longleftarrow X \sim P_\phi$$

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))] = \nabla_{\phi,\theta}\mathbb{E}_{P_Y}[L(Y)]$$

Two Levels of Intractability

$$Y \longleftarrow \boxed{f_\theta} \longleftarrow X \sim P_\phi$$

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))] = \nabla_{\phi,\theta}\mathbb{E}_{P_Y}[L(Y)]$$

- Intractable expectation (sum/integration)

Two Levels of Intractability

$$Y \longleftarrow \boxed{f_\theta} \longleftarrow X \sim P_\phi$$

$$\nabla_{\phi,\theta}\mathbb{E}_{X\sim P_\phi}[L(f_\theta(X))] = \nabla_{\phi,\theta}\mathbb{E}_{P_Y}[L(Y)]$$

- Intractable expectation (sum/integration)
- Intractable density functions: L could depend on $p_Y(y)$.

Today's Talk

Gradient estimation for differentiable programming in probabilistic models

- Gradient Estimation for Discrete Expectations
  - Titsias & **Shi**. (AISTATS'22)
  - **Shi**, Zhou, Hwang, Titsias & Mackey. (In Submission)
- Gradient Estimation for Intractable Densities
  - **Shi**, Sun & Zhu. (ICML'18)
  - Zhou, **Shi** & Zhu. (ICML'20)

Today's Talk

Gradient estimation for differentiable programming in probabilistic models

- Gradient Estimation for Discrete Expectations
  - Titsias & **Shi**. (AISTATS'22)
  - **Shi**, Zhou, Hwang, Titsias & Mackey. (In Submission)
- Gradient Estimation for Intractable Densities
  - **Shi**, Sun & Zhu. (ICML'18)
  - Zhou, **Shi** & Zhu. (ICML'20)

Why are Discrete Expectations Challenging?

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X \sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X \sim P_\phi}[\nabla_\theta f_\theta(X)]$

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options**:**

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X \sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X \sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X \sim P_\phi}[f(X)]$—three options**:**

- **Exact expectation + autodiff**

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X \sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X \sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X \sim P_\phi}[f(X)]$—three options:

X : d-dim binary vector => 2^d terms to sum

- **Exact expectation + autodiff**

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options**:**

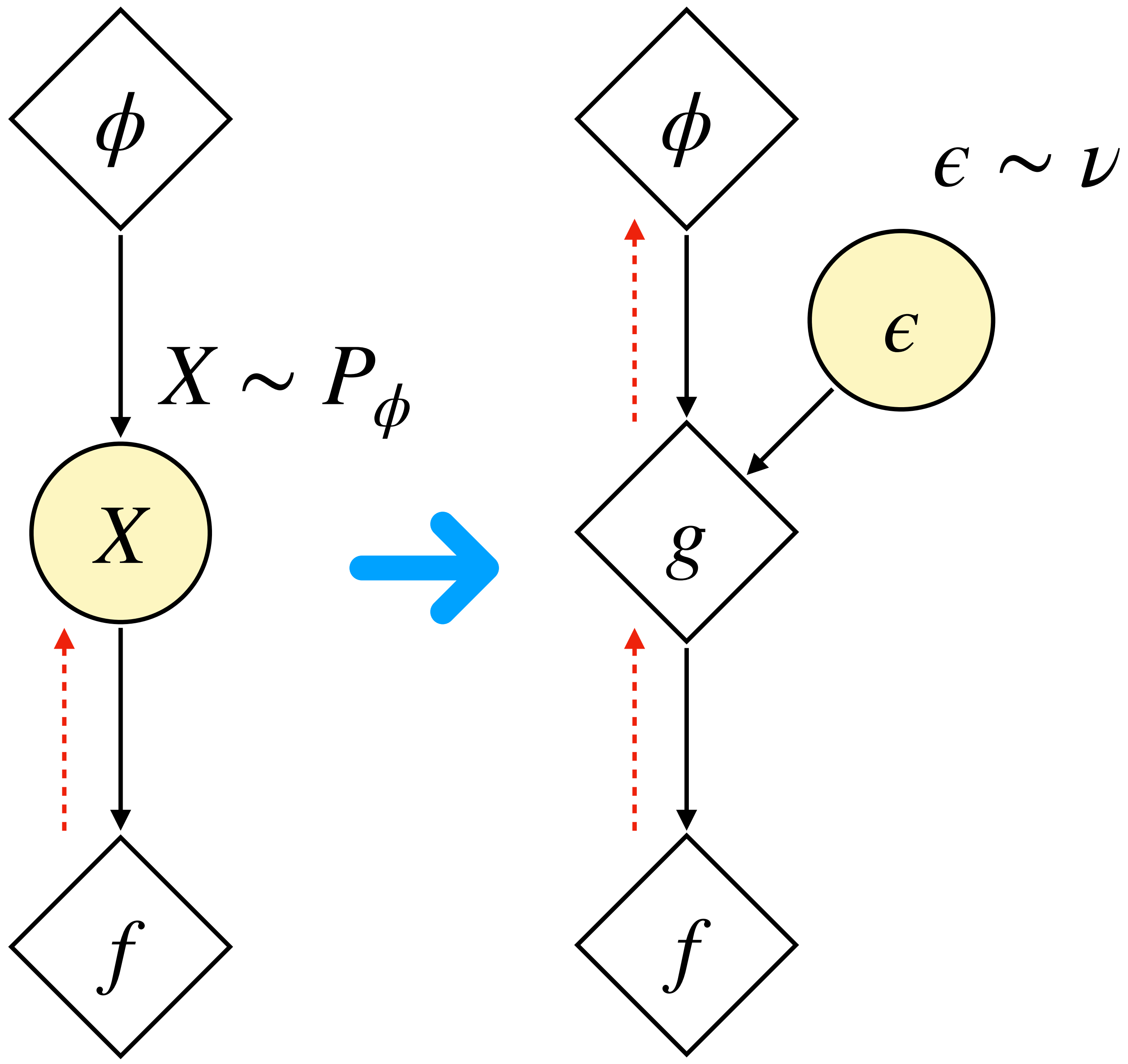
❌ • **Exact expectation + autodiff**

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options:

- ❌ **Exact expectation + autodiff**
- **Pathwise gradients**: reparameterize $X \sim P_\phi$:



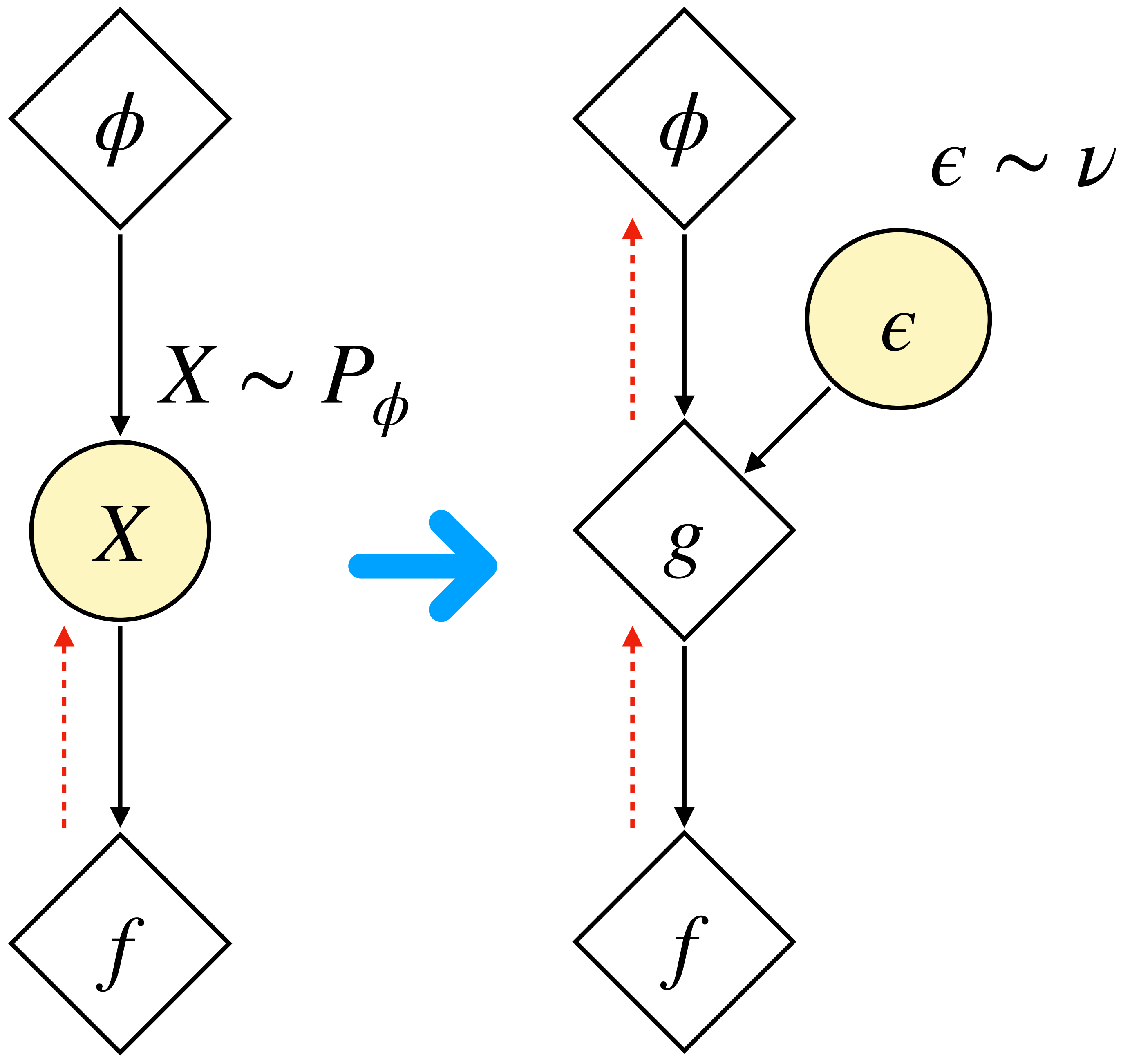
Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options**:**

❌ • **Exact expectation + autodiff**

• **Pathwise gradients**: reparameterize $X \sim P_\phi$:



only works for continuous distributions

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options**:**

❌ • **Exact expectation + autodiff**

❌ • **Pathwise gradients**: reparameterize $X \sim P_\phi$:

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options**:**

❌ • **Exact expectation + autodiff**

❌ • **Pathwise gradients**: reparameterize $X \sim P_\phi$:

• **REINFORCE**:
$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)] = \mathbb{E}_{X\sim P_\phi}[f(X)\nabla_\phi \log p_\phi(X)]$

Why are Discrete Expectations Challenging?

The easy part: $\nabla_\theta \mathbb{E}_{X\sim P_\phi}[f_\theta(X)] = \mathbb{E}_{X\sim P_\phi}[\nabla_\theta f_\theta(X)]$

$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)]$—three options:

❌ • **Exact expectation + autodiff**

❌ • **Pathwise gradients**: reparameterize $X \sim P_\phi$:

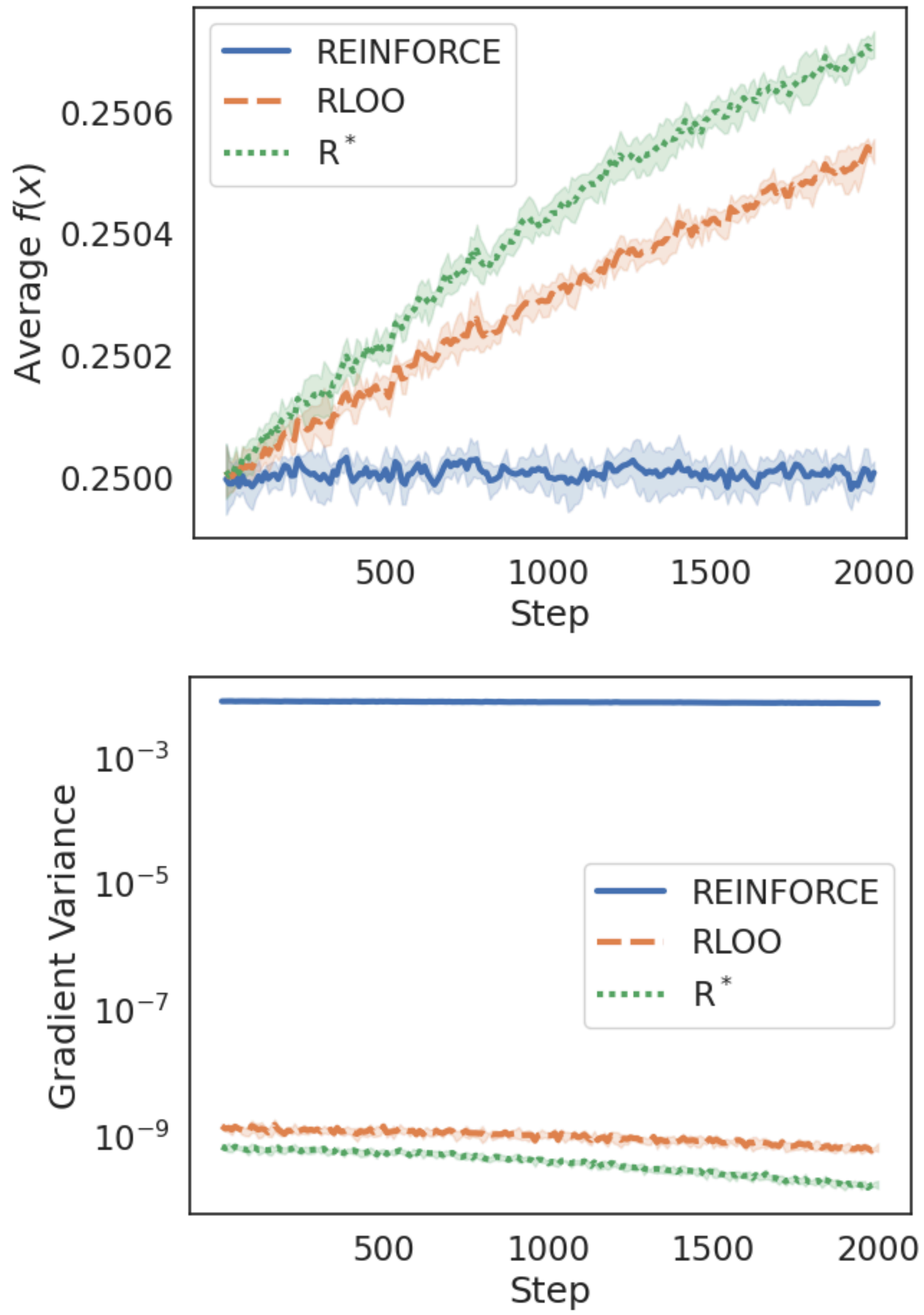
• **REINFORCE**:

$$\nabla_\phi \mathbb{E}_{X\sim P_\phi}[f(X)] = \mathbb{E}_{X\sim P_\phi}[f(X)\nabla_\phi \log p_\phi(X)]$$

very high variance

The REINFORCE Estimator

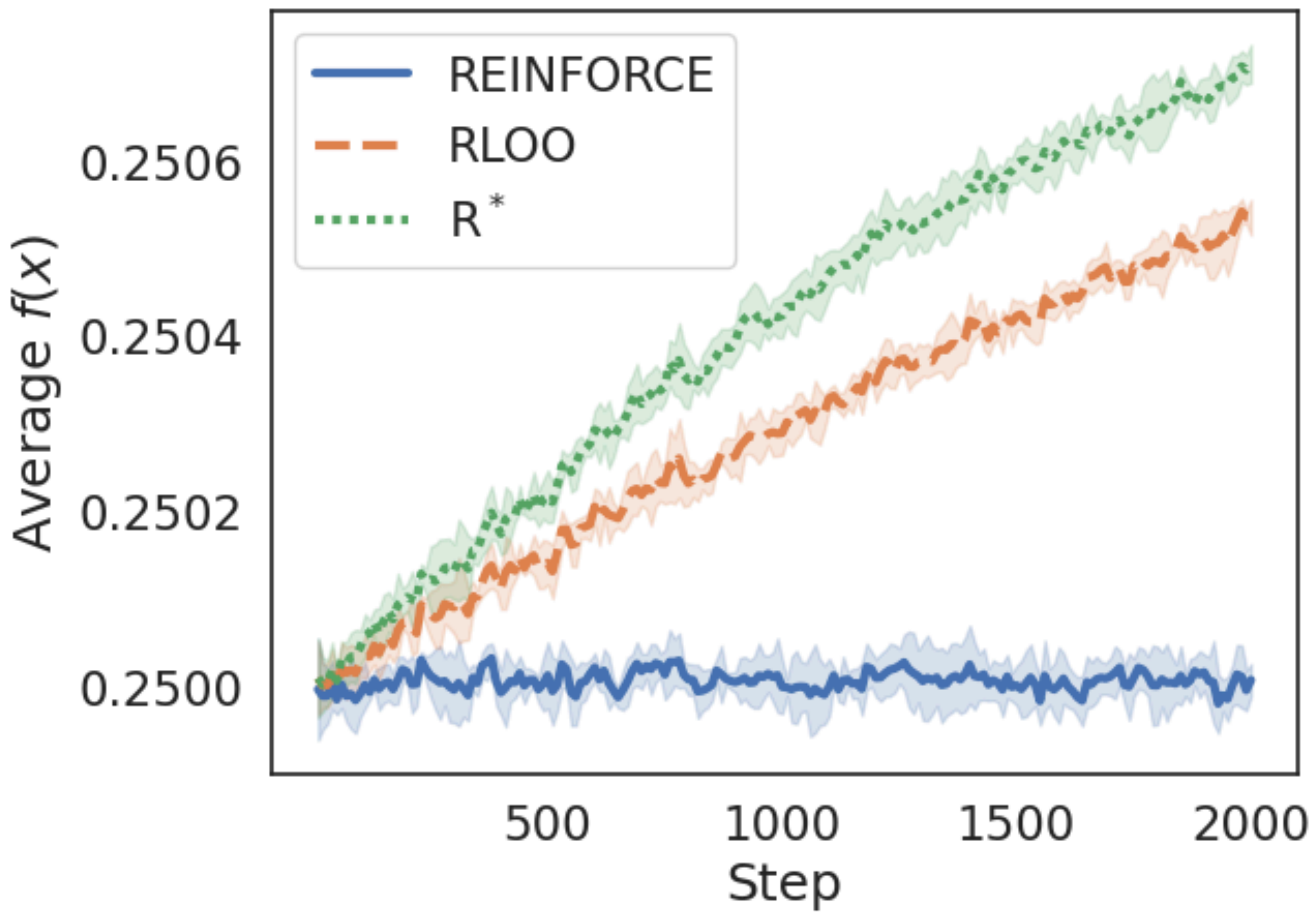
$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

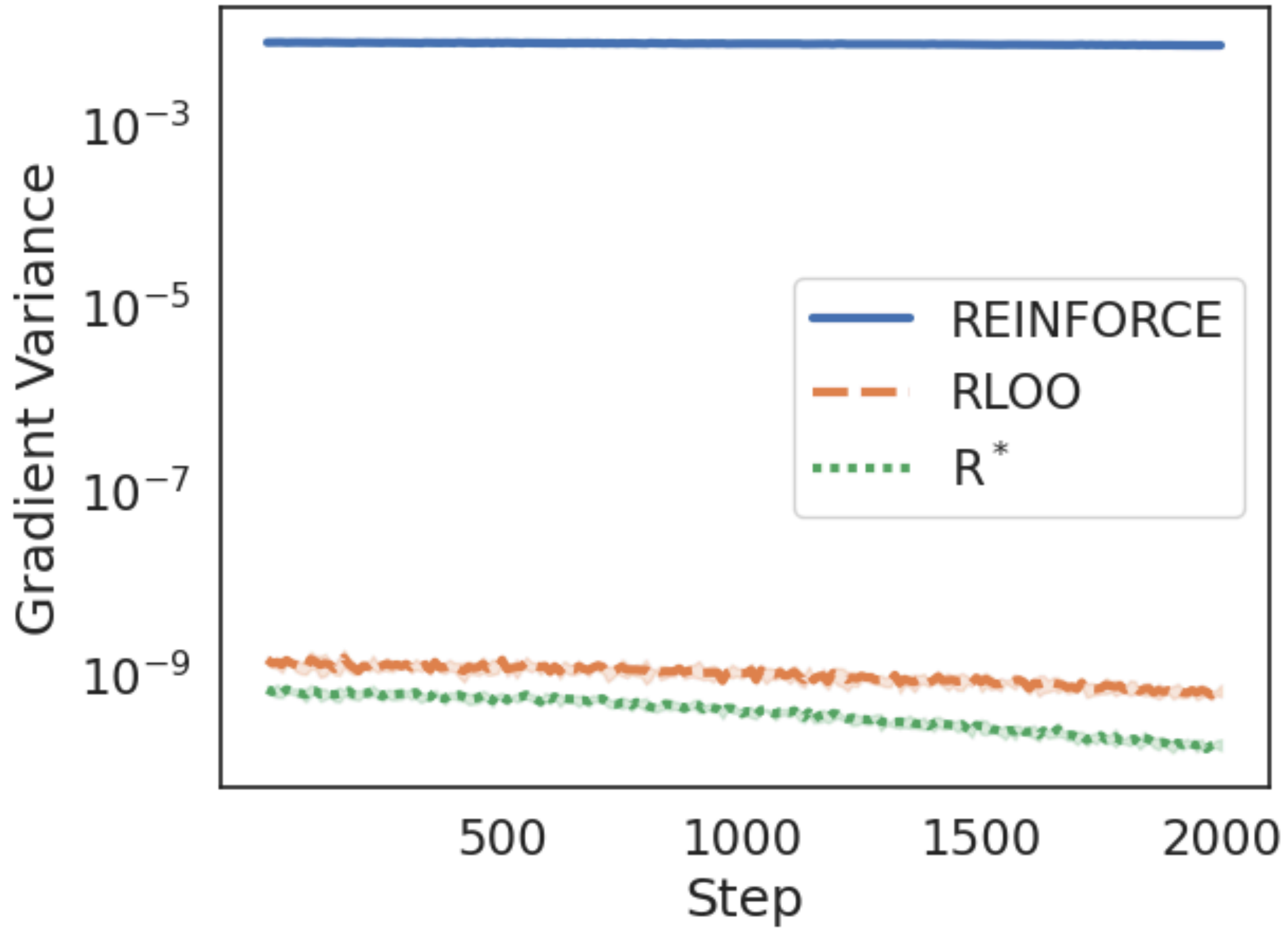


The REINFORCE Estimator

$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

REINFORCE with "baseline"



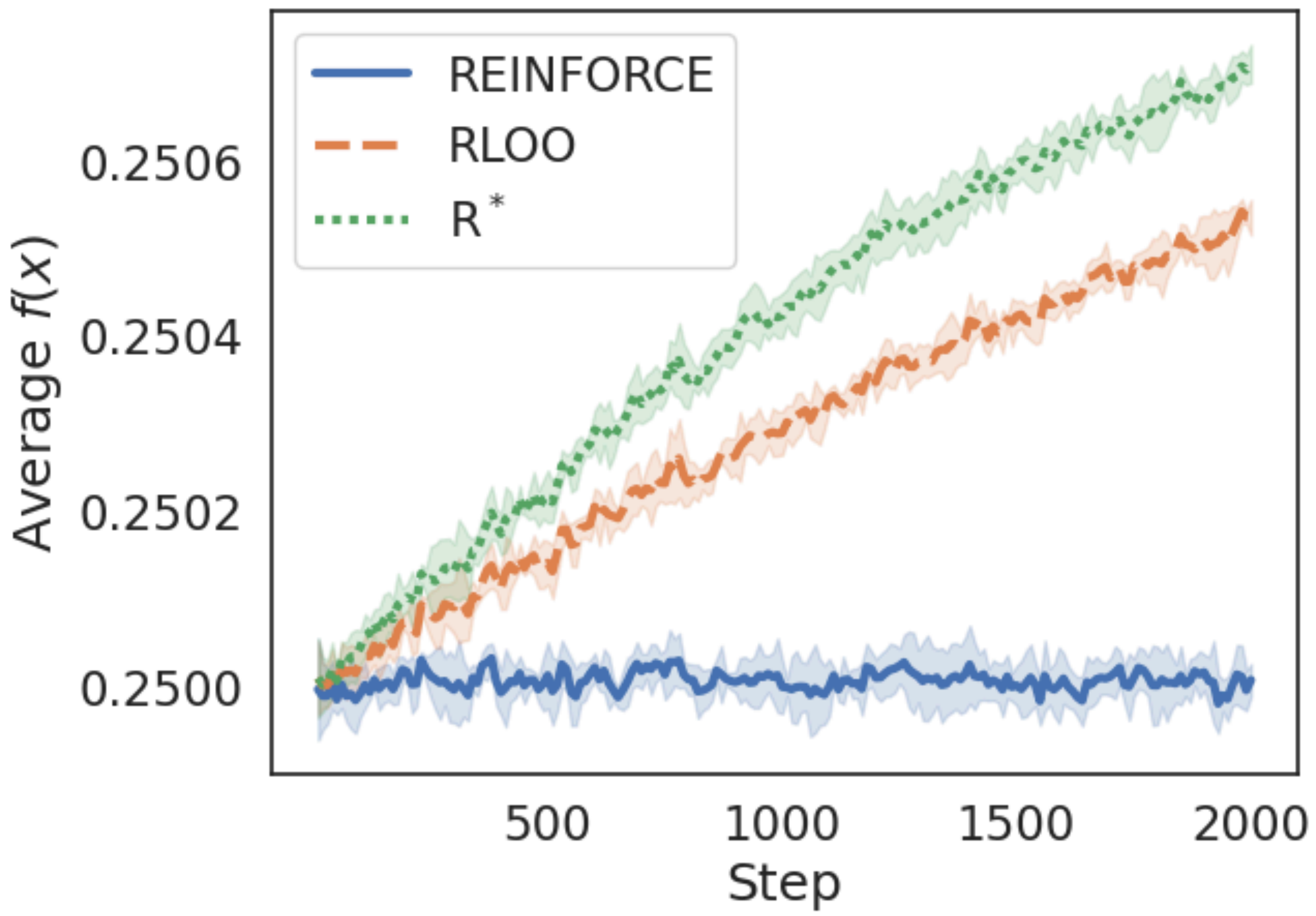


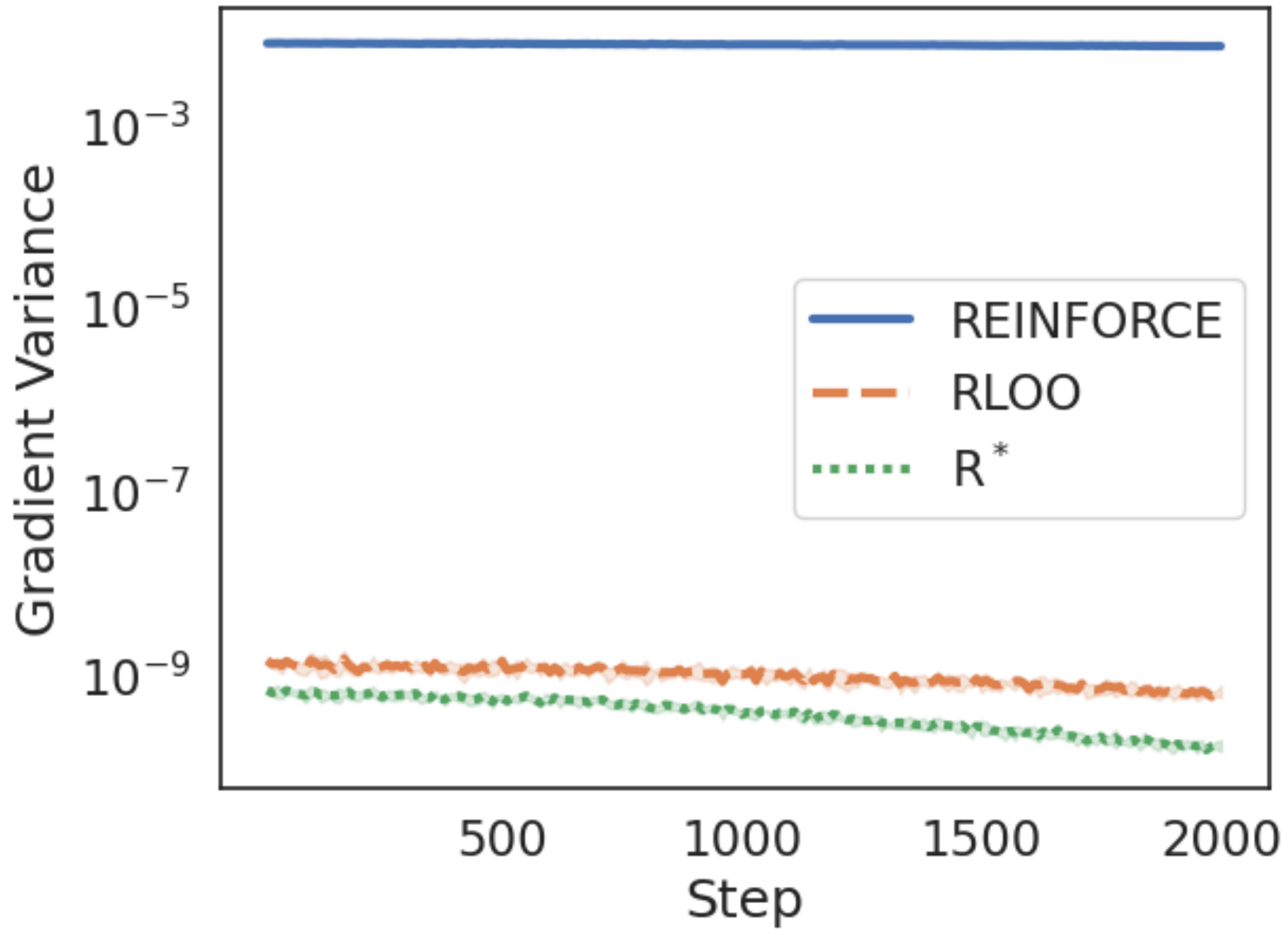
The REINFORCE Estimator

$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

REINFORCE with "baseline"

$$\frac{1}{K}\sum_{k=1}^{K} (f(x_k) - b)\,\nabla_\phi \log p_\phi(x_k)$$





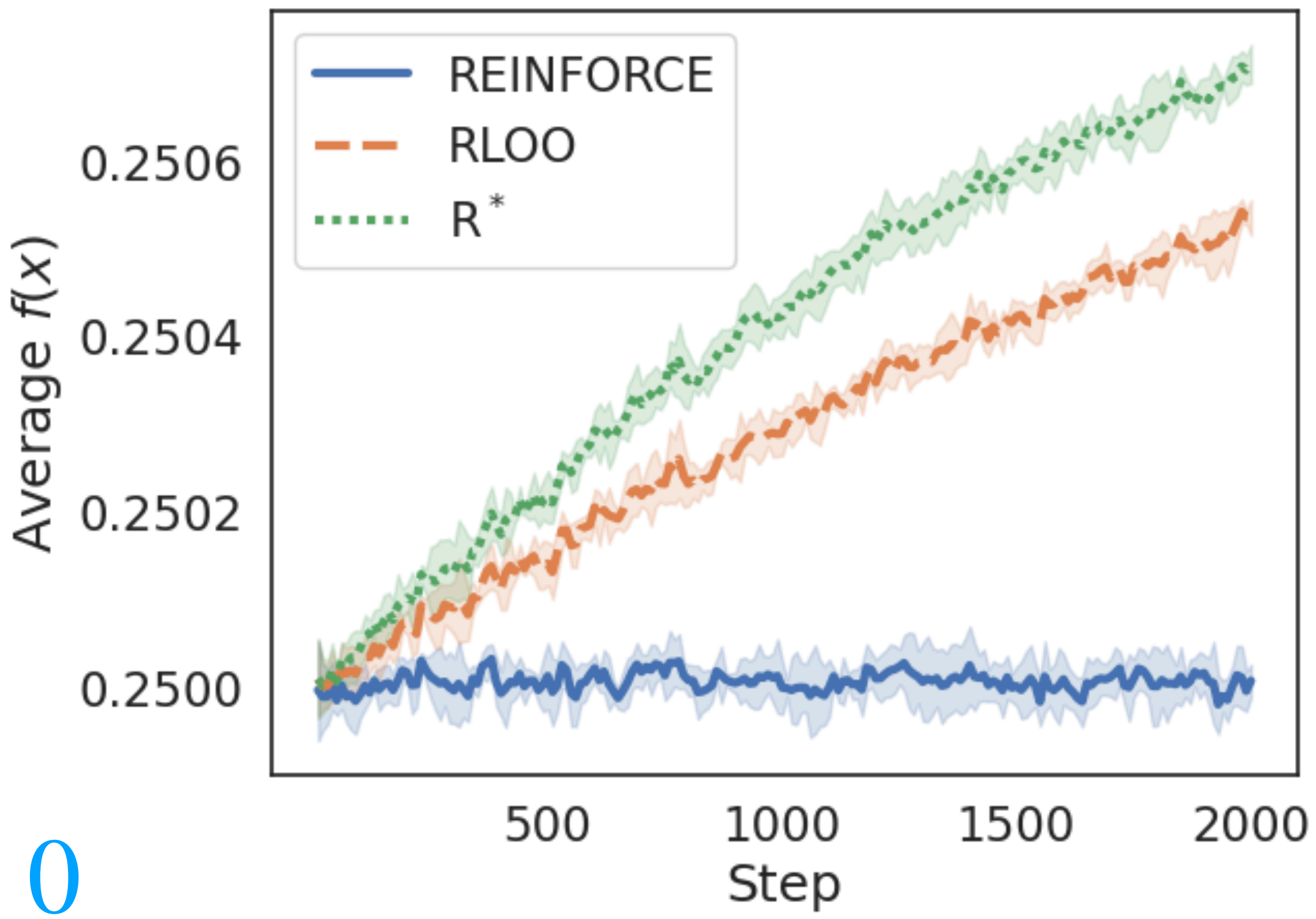
The REINFORCE Estimator

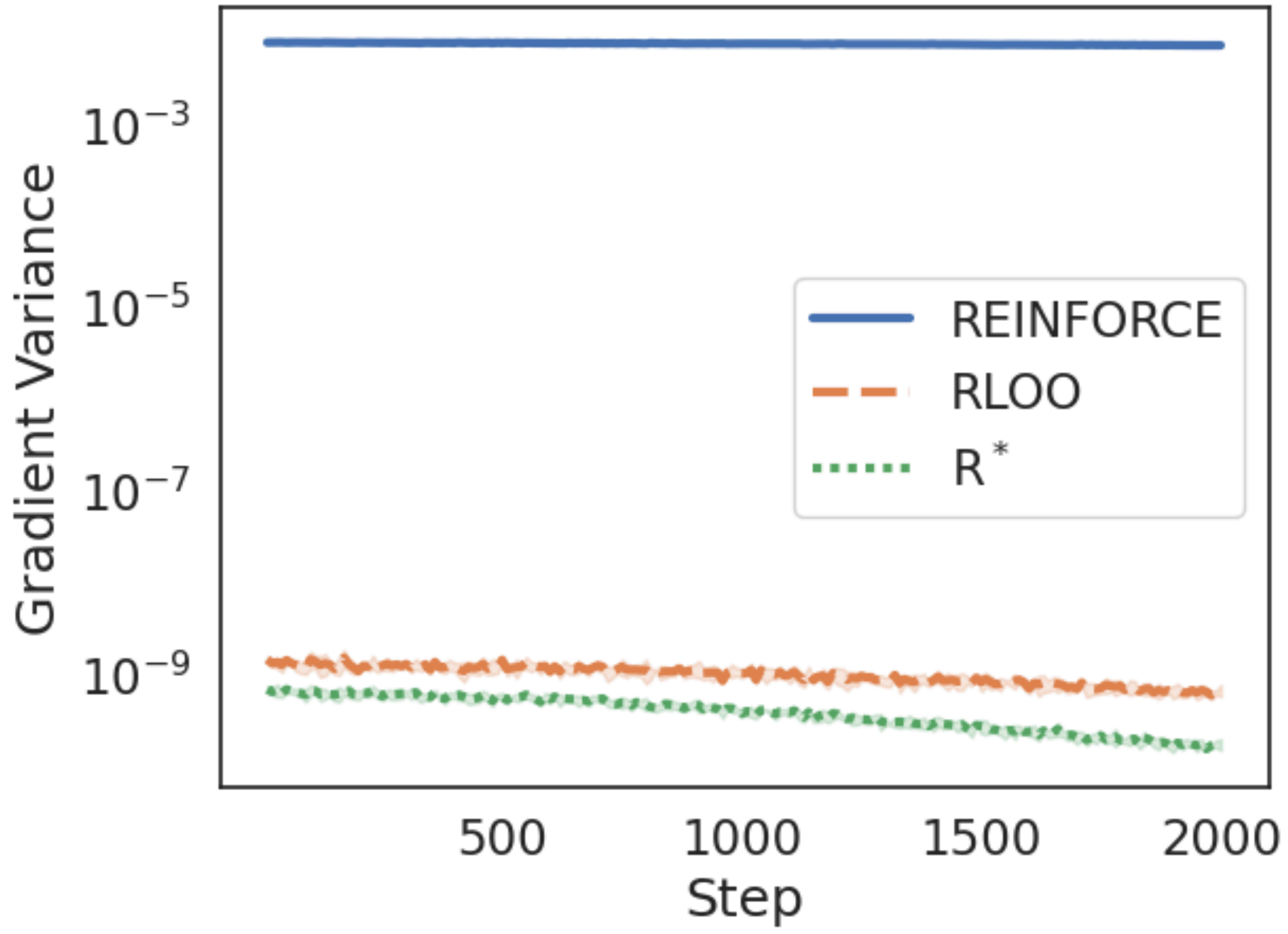
$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

REINFORCE with "baseline"

Control Variates, $\mathbb{E}_{P_\phi}[b\,\nabla_\phi \log p_\phi] = 0$

$$\frac{1}{K}\sum_{k=1}^{K} (f(x_k) - \boxed{b)\,\nabla_\phi \log p_\phi(x_k)}$$





The REINFORCE Estimator

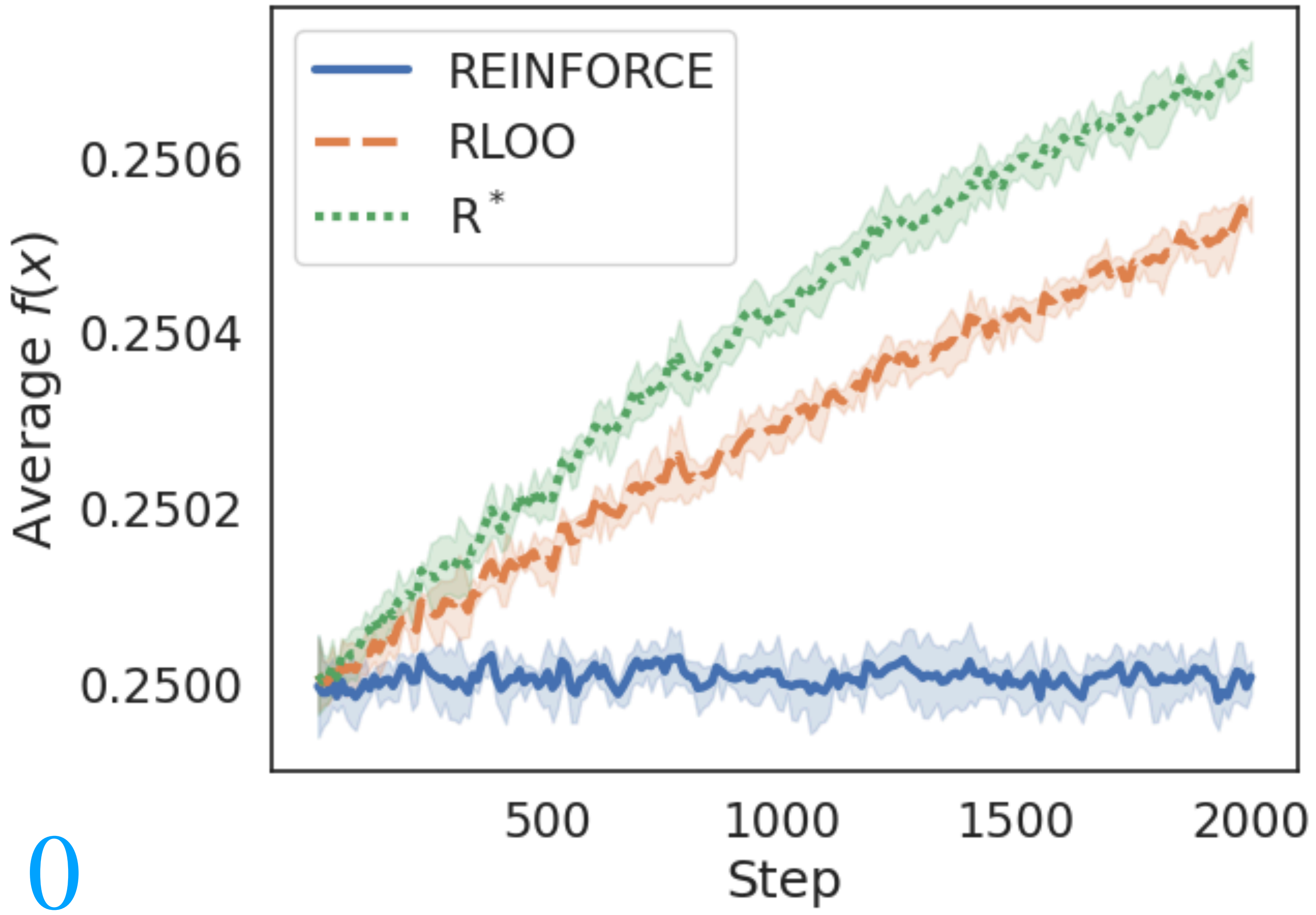
$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

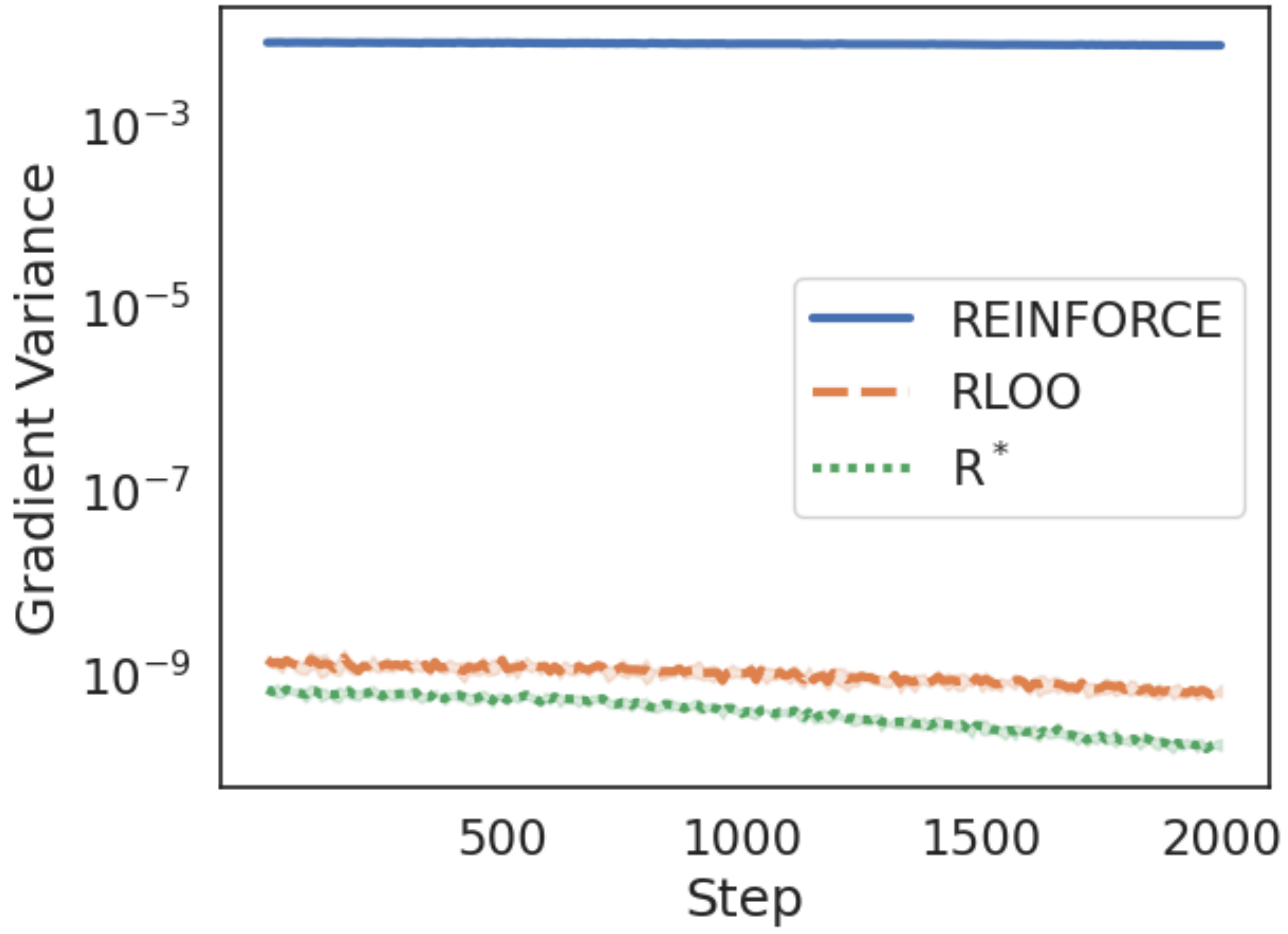
REINFORCE with "baseline"

Control Variates, $\mathbb{E}_{P_\phi}[b\,\nabla_\phi \log p_\phi] = 0$

$$\frac{1}{K}\sum_{k=1}^{K} (f(x_k) - \boxed{b)\,\nabla_\phi \log p_\phi(x_k)}$$

- "baseline" tracks the expected value of $f(x)$





The REINFORCE Estimator

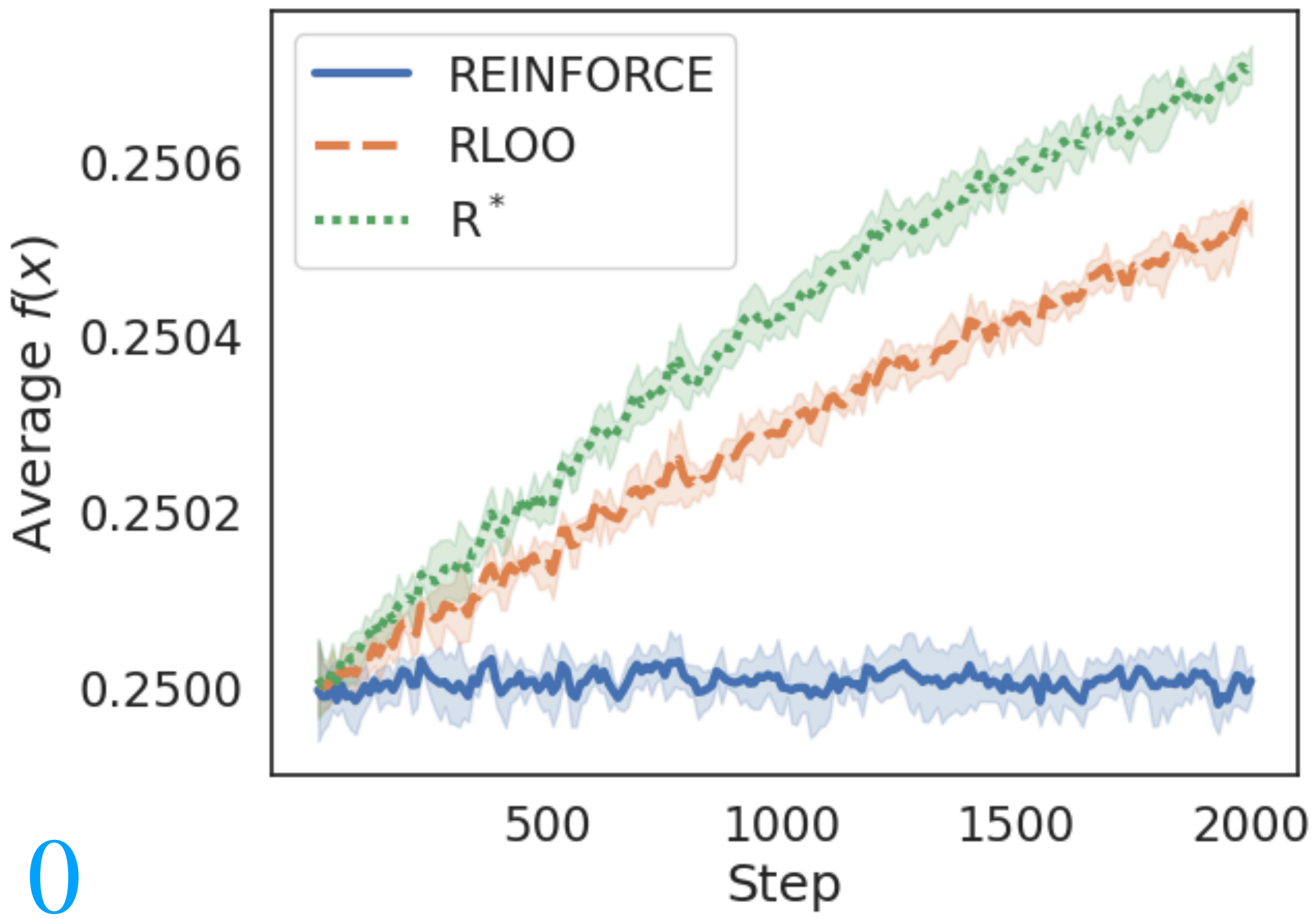
$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

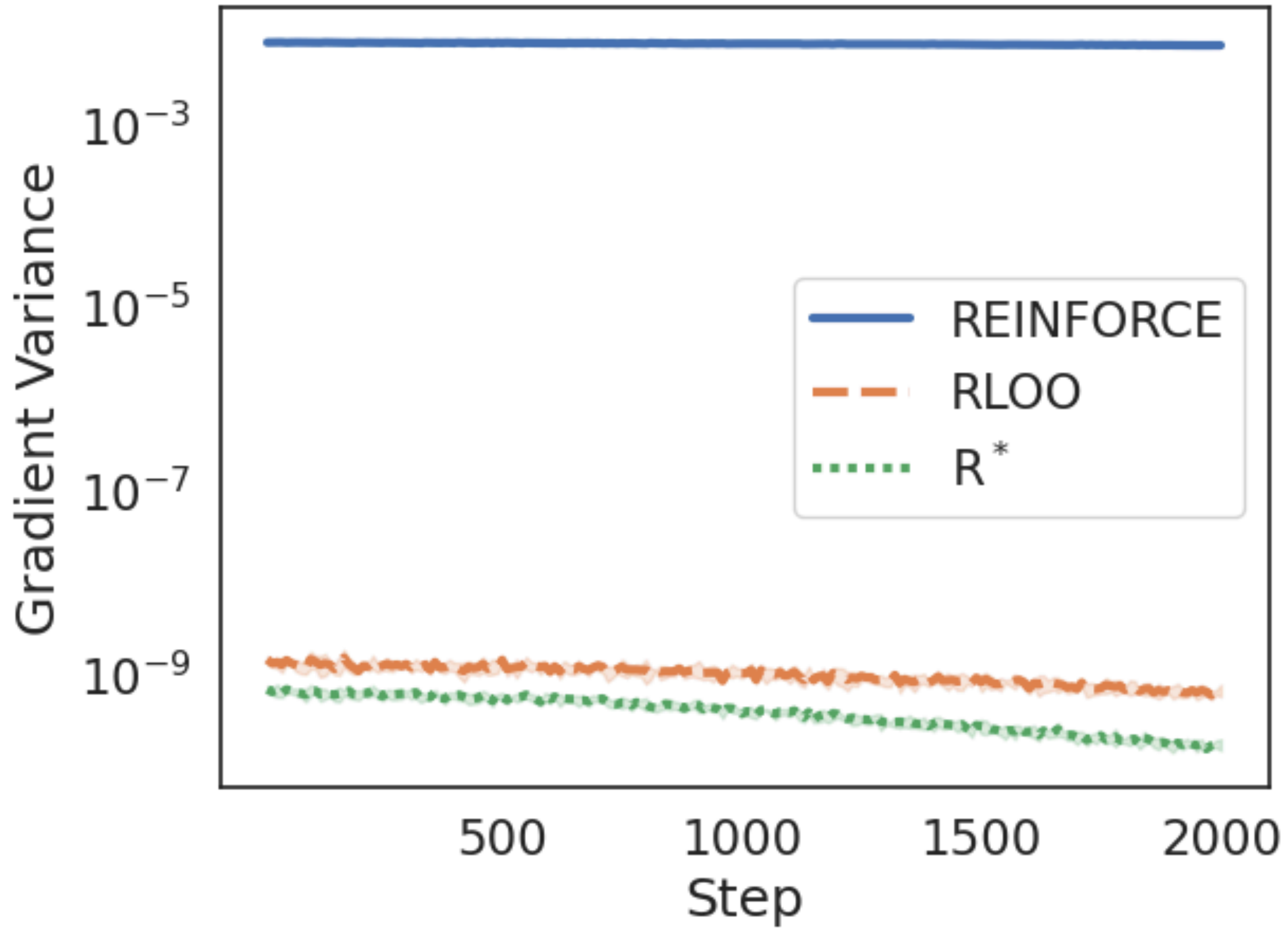
REINFORCE with "baseline"

Control Variates, $\mathbb{E}_{P_\phi}[b\,\nabla_\phi \log p_\phi] = 0$

$$\frac{1}{K}\sum_{k=1}^{K} (f(x_k) - \boxed{b)\,\nabla_\phi \log p_\phi(x_k)}$$

- "baseline" tracks the expected value of $f(x)$
- reduce variance by centering learning signal





The REINFORCE Estimator

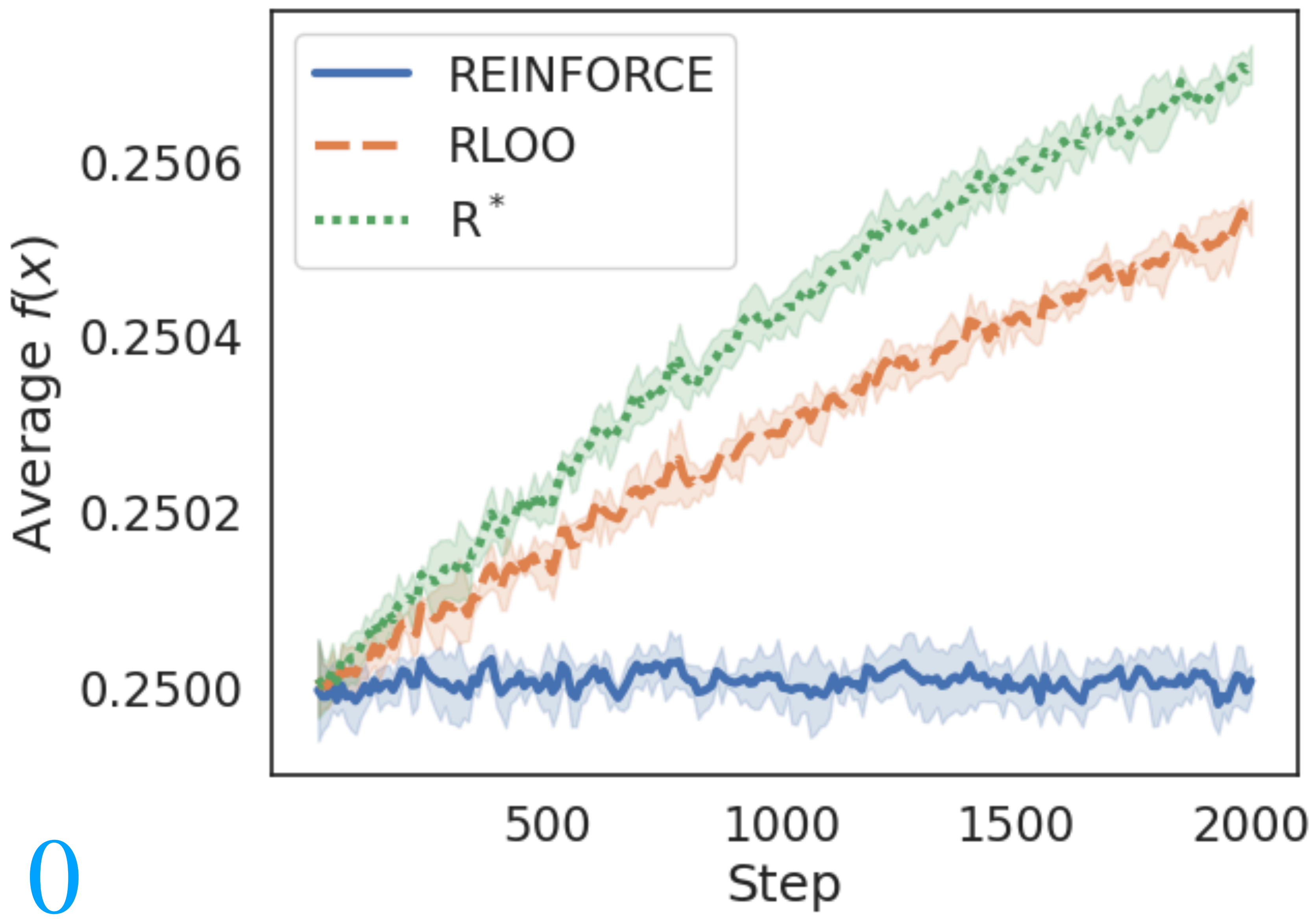
$$\frac{1}{K}\sum_{k=1}^{K} f(x_k)\,\nabla_\phi \log p_\phi(x_k), \quad x_{1:K} \sim P_\phi$$

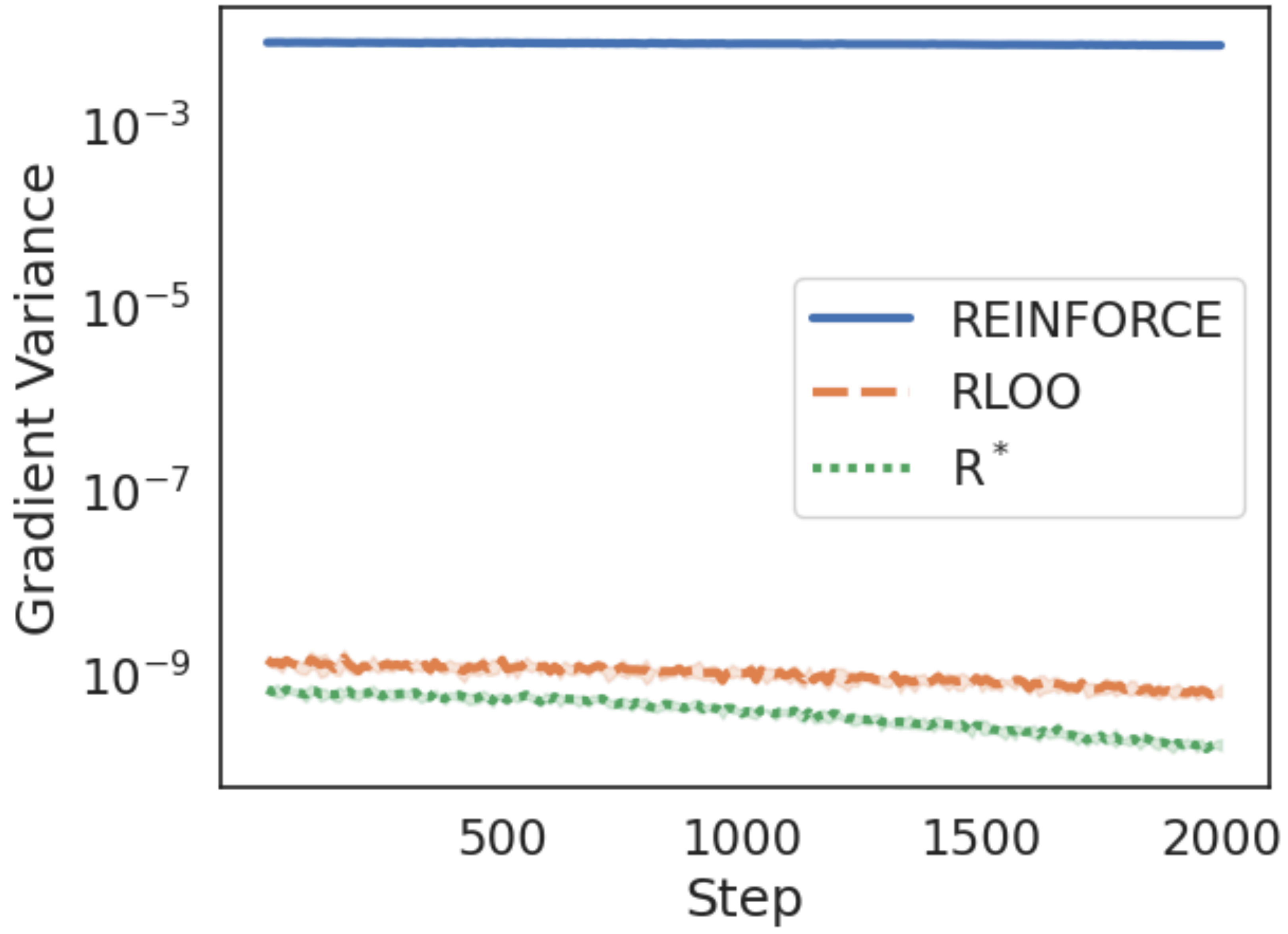
REINFORCE with "baseline"

Control Variates, $\mathbb{E}_{P_\phi}[b\,\nabla_\phi \log p_\phi] = 0$

$$\frac{1}{K}\sum_{k=1}^{K} (f(x_k) - \boxed{b)\,\nabla_\phi \log p_\phi(x_k)}$$

- "baseline" tracks the expected value of $f(x)$
- reduce variance by centering learning signal
- REINFORCE Leave-One-Out: $b = \frac{1}{K-1}\sum_{j \neq k} f(x_j)$





Proposition Define the two gradient estimators:

$$\text{RLOO: } \frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) - \frac{1}{K-1}\sum_{j\neq k} f(x_j)\right)\nabla_\phi \log p_\phi(x_k)$$

$$\text{R*: } \frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) - \mathbb{E}_{X\sim P_\phi}[f(X)]\right)\nabla_\phi \log p_\phi(x_k)$$

Then $\mathrm{Var}(\mathrm{RLOO}) \geq \mathrm{Var}(\mathrm{R}^*)$

Proposition Define the two gradient estimators:

$$\text{RLOO: } \frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) - \frac{1}{K-1}\sum_{j\neq k} f(x_j)\right)\nabla_\phi \log p_\phi(x_k)$$

$$\text{R*: } \frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) - \mathbb{E}_{X\sim P_\phi}[f(X)]\right)\nabla_\phi \log p_\phi(x_k)$$

Then $\mathrm{Var}(\mathrm{RLOO}) \geq \mathrm{Var}(\mathrm{R}^*)$

There is room for improving the state-of-the-art REINFORCE estimators

Proposition Define the two gradient estimators:

$$\text{RLOO: } \frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) - \frac{1}{K-1}\sum_{j\neq k} f(x_j)\right)\nabla_\phi \log p_\phi(x_k)$$

$$\text{R*: } \frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) - \mathbb{E}_{X\sim P_\phi}[f(X)]\right)\nabla_\phi \log p_\phi(x_k)$$

Then $\text{Var(RLOO)} \geq \text{Var(R*)}$

There is room for improving the state-of-the-art REINFORCE estimators

- potential direction: variance reduction for the leave-one-out baseline?

Gradient Estimation for Discrete Expectations

Roadmap

Double Control Variates

A new framework for variance reduction in REINFORCE-type estimators

Discrete Stein Operators

A general recipe for building flexible control variates for discrete distributions

Titsias & **Shi**. Double Control Variates for Gradient Estimation in Discrete Latent-Variable Models. AISTATS 2022
Shi, et al. Gradient Estimation with Discrete Stein Operators. In Submission.

Double Control Variates

Double Control Variates

We start with

Double Control Variates

We start with

$$\frac{1}{K}\sum_{k=1}^{K}\Big(f(x_k) + \alpha b(x_k)\Big)\nabla_\phi \log p_\phi(x_k) - \alpha \mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Double Control Variates

We start with

sample-dependent baseline

$$\frac{1}{K}\sum_{k=1}^{K}\Big(f(x_k) + \alpha b(x_k)\Big)\nabla_\phi \log p_\phi(x_k) - \alpha \mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Double Control Variates

We start with

sample-dependent baseline

correction term

$$\frac{1}{K}\sum_{k=1}^{K}\Big(f(x_k) + \alpha b(x_k)\Big)\nabla_\phi \log p_\phi(x_k) - \alpha \mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Double Control Variates

We start with

sample-dependent baseline

correction term

$$\frac{1}{K}\sum_{k=1}^{K}\left(f(x_k) + \alpha b(x_k)\right)\nabla_\phi \log p_\phi(x_k) - \alpha\mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Idea: Treat $f(x) + \alpha b(x)$ as the effective objective function and apply leave-one-out:

Double Control Variates

We start with

sample-dependent baseline

correction term

$$\frac{1}{K}\sum_{k=1}^{K}\Big(f(x_k) + \alpha b(x_k)\Big)\nabla_\phi \log p_\phi(x_k) - \alpha \mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Idea: Treat $f(x) + \alpha b(x)$ as the effective objective function and apply leave-one-out:

$$\frac{1}{K}\sum_{k=1}^{K}\Big((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\Big)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}$$

Double Control Variates

We start with

sample-dependent baseline

correction term

$$\frac{1}{K}\sum_{k=1}^{K}\Big(f(x_k) + \alpha b(x_k)\Big)\nabla_\phi \log p_\phi(x_k) - \alpha \mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Idea: Treat $f(x) + \alpha b(x)$ as the effective objective function and apply leave-one-out:

$$\frac{1}{K}\sum_{k=1}^{K}\Big((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\Big)\nabla_\phi \log p_\phi(x_k) - \text{corr}$$

"global"

"local"

Double Control Variates

We start with

sample-dependent baseline

correction term

$$\frac{1}{K}\sum_{k=1}^{K}\Big(f(x_k) + \alpha b(x_k)\Big)\nabla_\phi \log p_\phi(x_k) - \alpha \mathbb{E}_{X\sim P_\phi}[b(X)\nabla_\phi \log p_\phi(X)]$$

Idea: Treat $f(x) + \alpha b(x)$ as the effective objective function and apply leave-one-out:

$$\frac{1}{K}\sum_{k=1}^{K}\Big((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\Big)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}$$

"global"

"local"

α is a regression coefficient adapted online by minimizing variance.

Double Control Variates

$$\boxed{\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \text{corr}}$$

Double Control Variates

$$\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \text{corr}$$

Desired properties of the sample-dependent baseline:

Double Control Variates

$$\boxed{\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \text{corr}}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.

Double Control Variates

$$\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b(x_k))-\frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.

$$b(x) = f(\mu) + \nabla f(\mu)^\top (x-\mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

Double Control Variates

$$\boxed{\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.

$$b(x) = \underbrace{f(\mu)}_{\text{canceled}} + \nabla f(\mu)^\top (x-\mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

Double Control Variates

$$\boxed{\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k) + \alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j) + \alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.

$$b(x) = \qquad \nabla f(\mu)^\top (x - \mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

Double Control Variates

$$\boxed{\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.
- requires no extra evaluation of f

$$b(x) = \qquad \nabla f(\mu)^\top (x-\mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

Double Control Variates

$$\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k)+\alpha b_k(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j)+\alpha b_j(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \mathrm{corr}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.
- requires no extra evaluation of f

$$b_k(x) = \left(\frac{1}{K-1}\sum_{j\neq k}\nabla f(x_j)\right)^\top (x-\mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

Double Control Variates

$$\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k) + \alpha b_k(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j) + \alpha b_j(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \text{corr}$$

Desired properties of the sample-dependent baseline:

- The correction term has an analytical form.
- requires no extra evaluation of f

$$b_k(x) = \left(\frac{1}{K-1}\sum_{j\neq k}\nabla f(x_j)\right)^\top (x-\mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

$\{\nabla f(x_k)\}_{k=1}^{K}$ can be obtained **“for free”** from the same backpropagation to compute the θ gradients $\nabla_\theta f_\theta(x)$.

Double Control Variates

$$\frac{1}{K}\sum_{k=1}^{K}\left((f(x_k) + \alpha b_k(x_k)) - \frac{1}{K-1}\sum_{j\neq k}(f(x_j) + \alpha b_j(x_j))\right)\nabla_\phi \log p_\phi(x_k) - \text{corr}$$

Desired properties of the sample-dependent baseline:

	RLOO	Double CV
Time (sec/step)	0.0035	0.0036

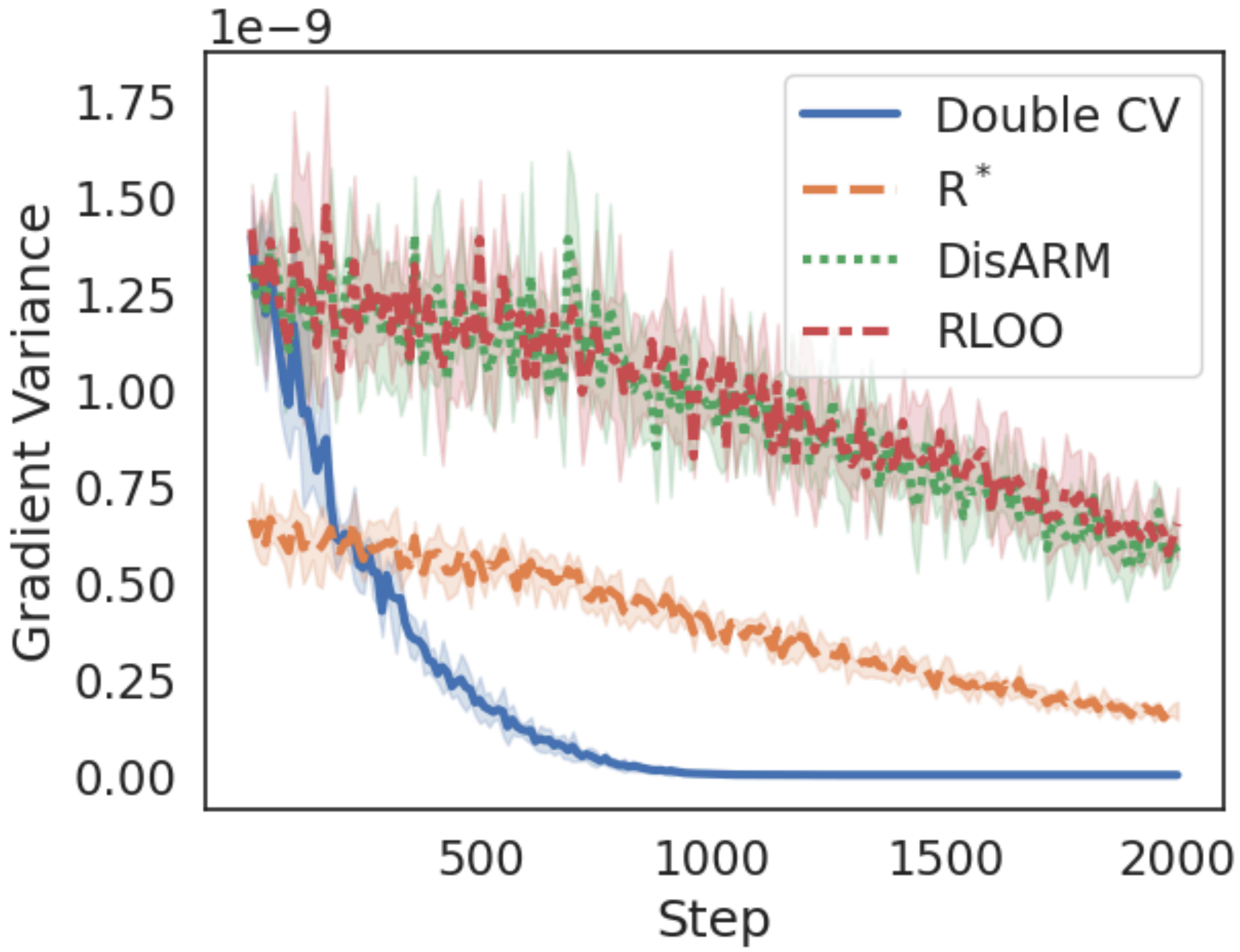
- The correction term has an analytical form.
- requires no extra evaluation of f

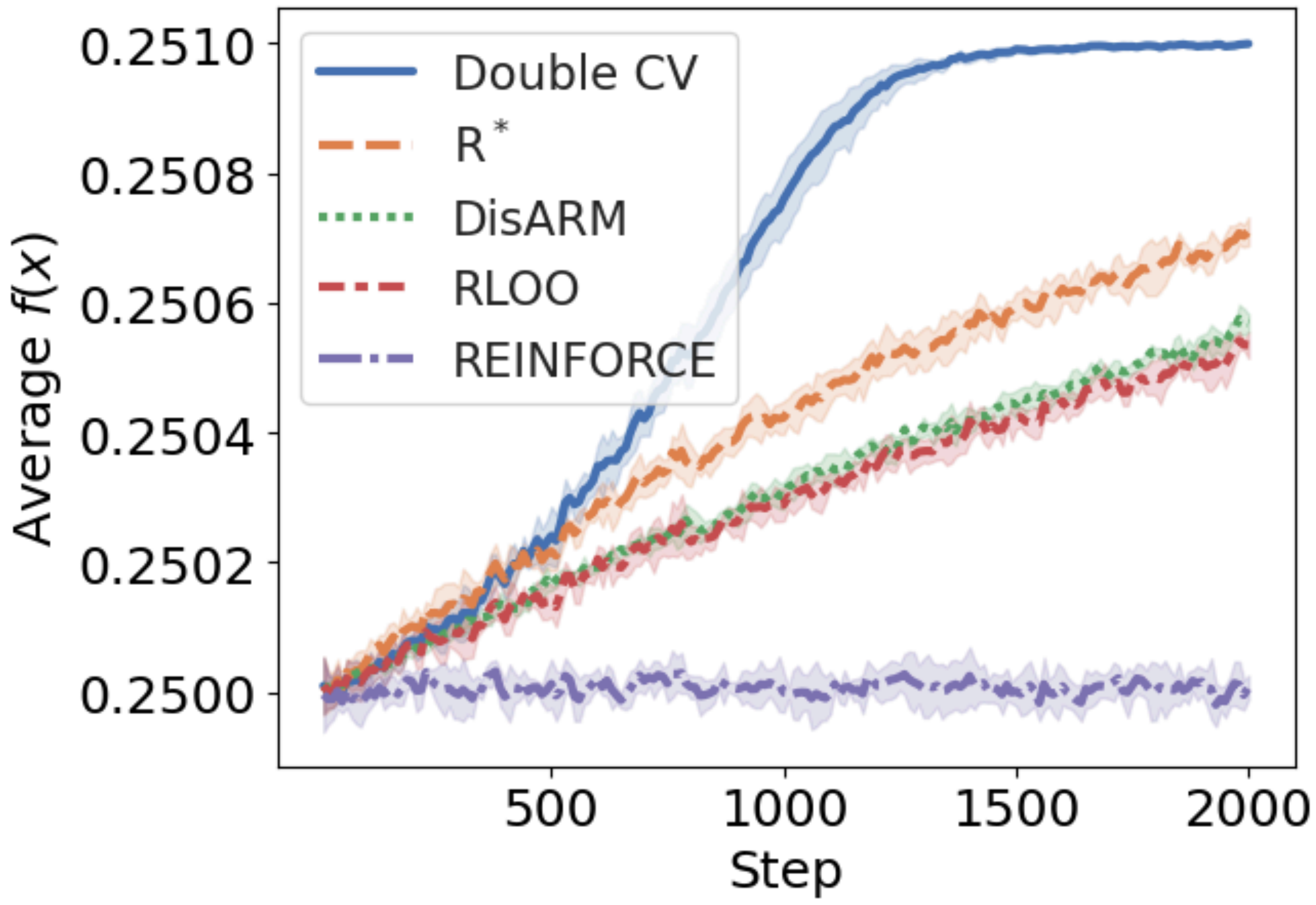
$$b_k(x) = \left(\frac{1}{K-1}\sum_{j\neq k}\nabla f(x_j)\right)^\top (x-\mu), \quad \mu = \mathbb{E}_{P_\phi}[X]$$

$\{\nabla f(x_k)\}_{k=1}^{K}$ can be obtained **"for free"** from the same backpropagation to compute the θ gradients $\nabla_\theta f_\theta(x)$.

Quadratic Loss Example

$$\max_{\eta} \mathbb{E}_{X \sim P_\eta} \left[\frac{1}{d} \sum_{i=1}^{d} (X_i - 0.499)^2 \right], \text{ where } p_\eta(x) = \prod_{i=1}^{d} \sigma(\eta_i)^{x_i} (1 - \sigma(\eta_i))^{1-x_i}$$





The Quest for More Flexible Control Variates

The Quest for More Flexible Control Variates

- Control variates are effective only when they are strongly correlated with the original statistic

The Quest for More Flexible Control Variates

- Control variates are effective only when they are strongly correlated with the original statistic
- Zero variance is achieved with $\alpha b(x) = f(x)$

The Quest for More Flexible Control Variates

- Control variates are effective only when they are strongly correlated with the original statistic

- Zero variance is achieved with $\alpha b(x) = f(x)$

- Ideally, would like a very flexible control variate that can be adapted online to minimize the variance

The Quest for More Flexible Control Variates

- Control variates are effective only when they are strongly correlated with the original statistic
- Zero variance is achieved with $\alpha b(x) = f(x)$
- Ideally, would like a very flexible control variate that can be adapted online to minimize the variance
- Still, they need to have analytic expectations

Stein Operators

Computable functionals that generate zero-mean functions

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

$$\mathbb{E}_{X \sim Q}[(Ah)(X)] = 0$$

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

holds for h in a flexible function class

$$\mathbb{E}_{X \sim Q}[(Ah)(X)] = 0$$

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

holds for h in a flexible function class

$$\mathbb{E}_{X \sim Q}[(Ah)(X)] = 0$$

analytic expectations

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

holds for h in a flexible function class

$$\mathbb{E}_{X \sim Q}[(Ah)(X)] = 0$$ analytic expectations

- introduced by Stein (1972) for characterizing distributional convergence.

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

holds for h in a flexible function class

$$\mathbb{E}_{X \sim Q}[(Ah)(X)] = 0$$

analytic expectations

- introduced by Stein (1972) for characterizing distributional convergence.
- the operator he developed for normal distribution Q:

Stein Operators

Computable functionals that generate zero-mean functions

Definition A *Stein operator* A takes input function h and outputs mean-zero functions under distribution Q:

holds for h in a flexible function class

$$\mathbb{E}_{X \sim Q}[(Ah)(X)] = 0$$

analytic expectations

- introduced by Stein (1972) for characterizing distributional convergence.
- the operator he developed for normal distribution Q:

$$(Ah)(x) = h'(x) - xh(x)$$

Constructing Discrete Stein Operators

A general recipe

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution
- The transition matrix $K_{xy} = P(X_{t+1} = y \,|\, X_t = x)$ satisfies

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution
- The transition matrix $K_{xy} = P(X_{t+1} = y \,|\, X_t = x)$ satisfies

$$\mathbb{E}_Q[(K - I)h] = 0 \quad \text{for any } h.$$

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution
- The transition matrix $K_{xy} = P(X_{t+1} = y \mid X_t = x)$ satisfies

$$\mathbb{E}_Q[\underbrace{(K - I)}_{=A}h] = 0 \quad \text{for any } h.$$

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution
- The transition matrix $K_{xy} = P(X_{t+1} = y \,|\, X_t = x)$ satisfies

$$\mathbb{E}_Q[\underbrace{(K - I)}_{=\mathcal{A}}h] = 0 \quad \text{for any } h.$$

- **Gibbs Stein operator**:

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution
- The transition matrix $K_{xy} = P(X_{t+1} = y \,|\, X_t = x)$ satisfies

$$\mathbb{E}_Q[\underbrace{(K - I)}_{=A}h] = 0 \quad \text{for any } h.$$

- **Gibbs Stein operator**:

$$(Ah)(x) = \frac{1}{d}\sum_{i=1}^{d}\Big(\sum_{\substack{y_i \neq x_i,\\ y_{-i} = x_{-i}}} q(y_i \,|\, x_{-i})h(y) + (q(x_i \,|\, x_{-i}) - 1)h(x)\Big)$$

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution

- The transition matrix $K_{xy} = P(X_{t+1} = y \,|\, X_t = x)$ satisfies

$$\mathbb{E}_Q[\underset{=A}{(K - I)}h] = 0 \quad \text{for any } h.$$

- **Gibbs Stein operator**:

evaluation at neighboring states

$$(Ah)(x) = \frac{1}{d}\sum_{i=1}^{d}\Big(\sum_{\substack{y_i \neq x_i,\\ y_{-i} = x_{-i}}} q(y_i \,|\, x_{-i})h(y) + (q(x_i \,|\, x_{-i}) - 1)h(x)\Big)$$

Constructing Discrete Stein Operators

A general recipe

- Identify a Markov Chain $(X_t)_{t=0}^{\infty}$ with Q the stationary distribution
- The transition matrix $K_{xy} = P(X_{t+1} = y \,|\, X_t = x)$ satisfies

$$\mathbb{E}_Q[\underbrace{(K - I)}_{=A}h] = 0 \quad \text{for any } h.$$

- **Gibbs Stein operator**:

evaluation at neighboring states

$$(Ah)(x) = \frac{1}{d}\sum_{i=1}^{d}\Big(\sum_{\substack{y_i \neq x_i,\\ y_{-i} = x_{-i}}} q(y_i \,|\, x_{-i})h(y) + (q(x_i \,|\, x_{-i}) - 1)h(x)\Big)$$

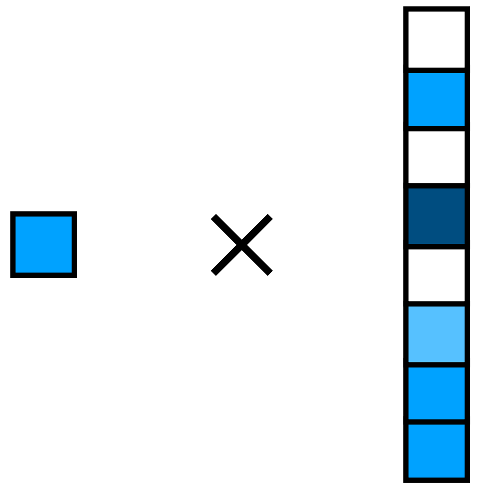
- see paper for generalization to continuous-time chains

Gradient Estimation with Discrete Stein Operators

$$\frac{1}{K}\sum_{k=1}^{K}\left[f(x_k)\,\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)\right]$$

Gradient Estimation with Discrete Stein Operators

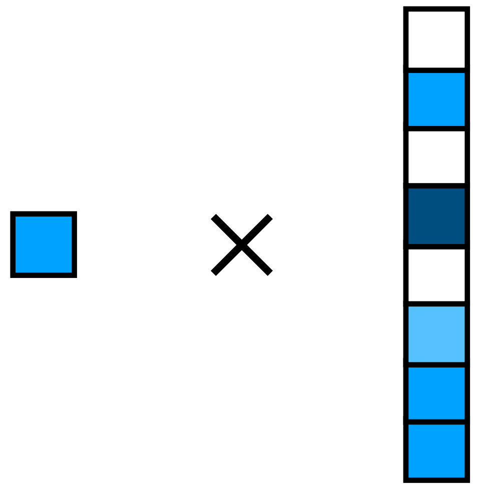
$$\frac{1}{K}\sum_{k=1}^{K}\left[f(x_k)\,\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)\right]$$



Gradient Estimation with Discrete Stein Operators

$\mathcal{X} \to \mathbb{R}^d$

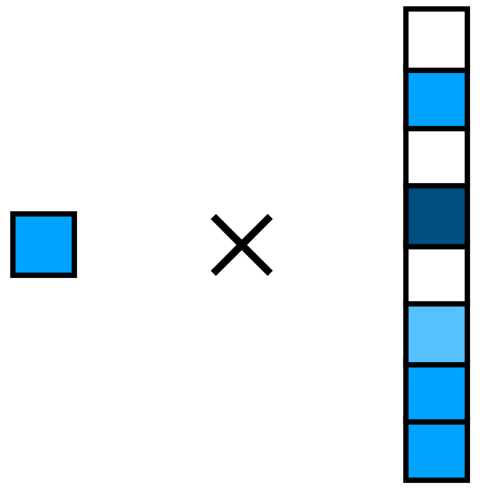
$$\frac{1}{K}\sum_{k=1}^{K}\left[f(x_k)\,\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)\right]$$



Gradient Estimation with Discrete Stein Operators

$$\mathcal{X} \to \mathbb{R}^d$$

$$\frac{1}{K}\sum_{k=1}^{K}\left[f(x_k)\,\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)\right]$$



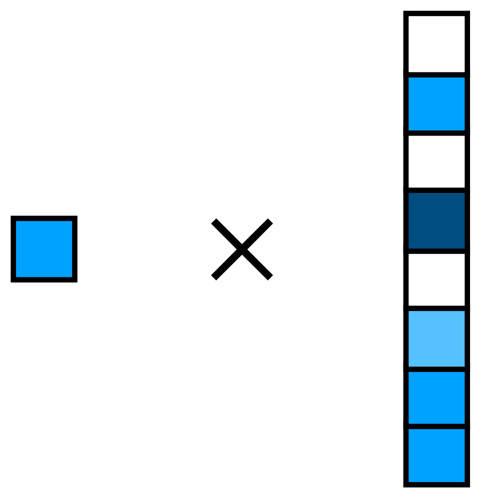
$$A\begin{pmatrix}\tilde{h}_1\\ \tilde{h}_2\\ \vdots\\ \tilde{h}_d\end{pmatrix}$$

Gradient Estimation with Discrete Stein Operators

How to choose $\tilde{h}$:

$$\mathcal{X} \to \mathbb{R}^d$$

$$\frac{1}{K}\sum_{k=1}^{K}\left[f(x_k)\,\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)\right]$$

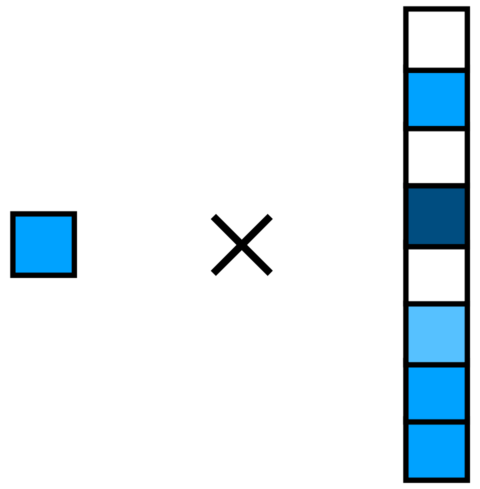


$$A\begin{pmatrix}\tilde{h}_1\\ \tilde{h}_2\\ \vdots\\ \tilde{h}_d\end{pmatrix}$$

Gradient Estimation with Discrete Stein Operators

$\mathcal{X} \to \mathbb{R}^d$

$$\frac{1}{K}\sum_{k=1}^{K} [f(x_k)\,\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)]$$



$$A\begin{pmatrix}\tilde{h}_1\\ \tilde{h}_2\\ \vdots\\ \tilde{h}_d\end{pmatrix}$$

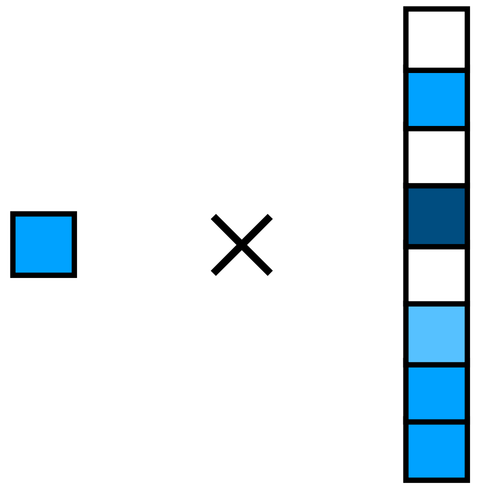
How to choose $\tilde{h}$:

Option 1: Solve d Poisson equations

Gradient Estimation with Discrete Stein Operators

$$\mathcal{X} \to \mathbb{R}^d$$

$$\frac{1}{K}\sum_{k=1}^{K} [f(x_k)\, \nabla_\eta \log q_\eta(x_k) \; + \; (A\tilde{h})(x_k)]$$



$$A \begin{pmatrix} \tilde{h}_1 \\ \tilde{h}_2 \\ \vdots \\ \tilde{h}_d \end{pmatrix}$$

How to choose $\tilde{h}$:

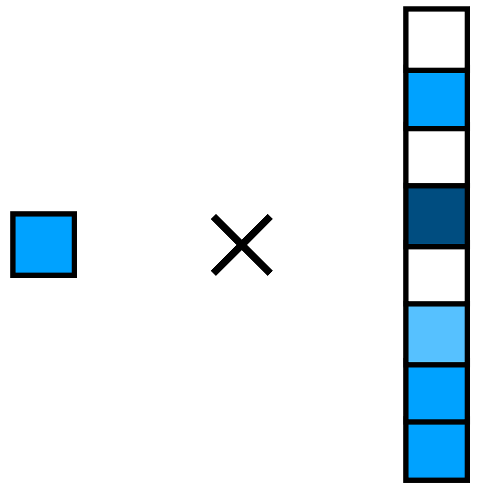
Option 1: Solve d Poisson equations

$$A\tilde{h}_i = \mathbb{E}_Q[f\, \nabla_{\eta_i} \log q_\eta] - f\, \nabla_{\eta_i} \log q_\eta$$

Gradient Estimation with Discrete Stein Operators

$$\mathcal{X} \to \mathbb{R}^d$$

$$\frac{1}{K}\sum_{k=1}^{K}[f(x_k)\nabla_\eta \log q_\eta(x_k) \; + \; (A\tilde{h})(x_k)]$$



$$A\begin{pmatrix}\tilde{h}_1\\ \tilde{h}_2\\ \vdots \\ \tilde{h}_d\end{pmatrix}$$

How to choose $\tilde{h}$:

Option 1: Solve d Poisson equations

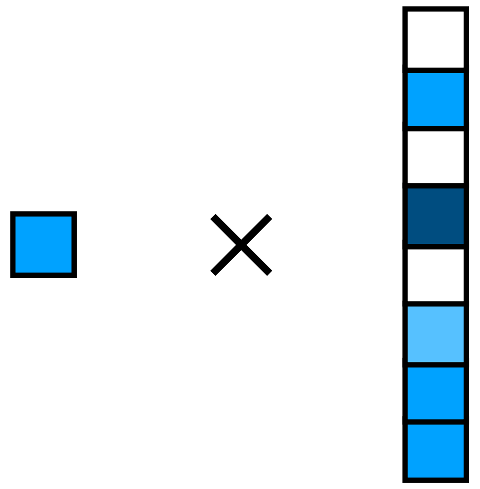
$$A\tilde{h}_i = \mathbb{E}_Q[f\nabla_{\eta_i}\log q_\eta] - f\nabla_{\eta_i}\log q_\eta$$

Option 2: $\tilde{h} := h\nabla_\eta \log q_\eta$

Gradient Estimation with Discrete Stein Operators

$$\mathscr{X} \to \mathbb{R}^d$$

$$\frac{1}{K}\sum_{k=1}^{K}\left[f(x_k)\nabla_\eta \log q_\eta(x_k) \;+\; (A\tilde{h})(x_k)\right]$$



$$A\begin{pmatrix}\tilde{h}_1\\ \tilde{h}_2\\ \vdots\\ \tilde{h}_d\end{pmatrix}$$

How to choose $\tilde{h}$:

Option 1: Solve d Poisson equations

$$A\tilde{h}_i = \mathbb{E}_Q[f\nabla_{\eta_i}\log q_\eta] - f\nabla_{\eta_i}\log q_\eta$$

Option 2: $\tilde{h} := h\nabla_\eta \log q_\eta$

Theorem When $h = f$, estimators with this $\tilde{h}$ reduce to **Rao-Blackwellization** $K(f\nabla_\eta \log q_\eta)$ which guarantees variance reduction

Gradient Estimation with Discrete Stein Operators

Gradient Estimation with Discrete Stein Operators

- By design h is evaluated at all neighbors of x_k

Gradient Estimation with Discrete Stein Operators

- By design h is evaluated at all neighbors of x_k
- make h cheap while informed about f

Gradient Estimation with Discrete Stein Operators

- By design h is evaluated at all neighbors of x_k
- make h cheap while informed about f

$$h_k(y) = \frac{1}{K-1} \sum_{j \neq k} H(f(x_j), \nabla f(x_j)^\top (y - x_j))$$

Gradient Estimation with Discrete Stein Operators

- By design h is evaluated at all neighbors of x_k
- make h cheap while informed about f

$$h_k(y) = \frac{1}{K-1} \sum_{j \neq k} H(f(x_j), \nabla f(x_j)^\top (y - x_j))$$

Important: no extra evaluation of f

Gradient Estimation with Discrete Stein Operators

- By design h is evaluated at all neighbors of x_k
- make h cheap while informed about f

$$h_k(y) = \frac{1}{K-1} \sum_{j \neq k} H(f(x_j), \nabla f(x_j)^\top (y - x_j))$$

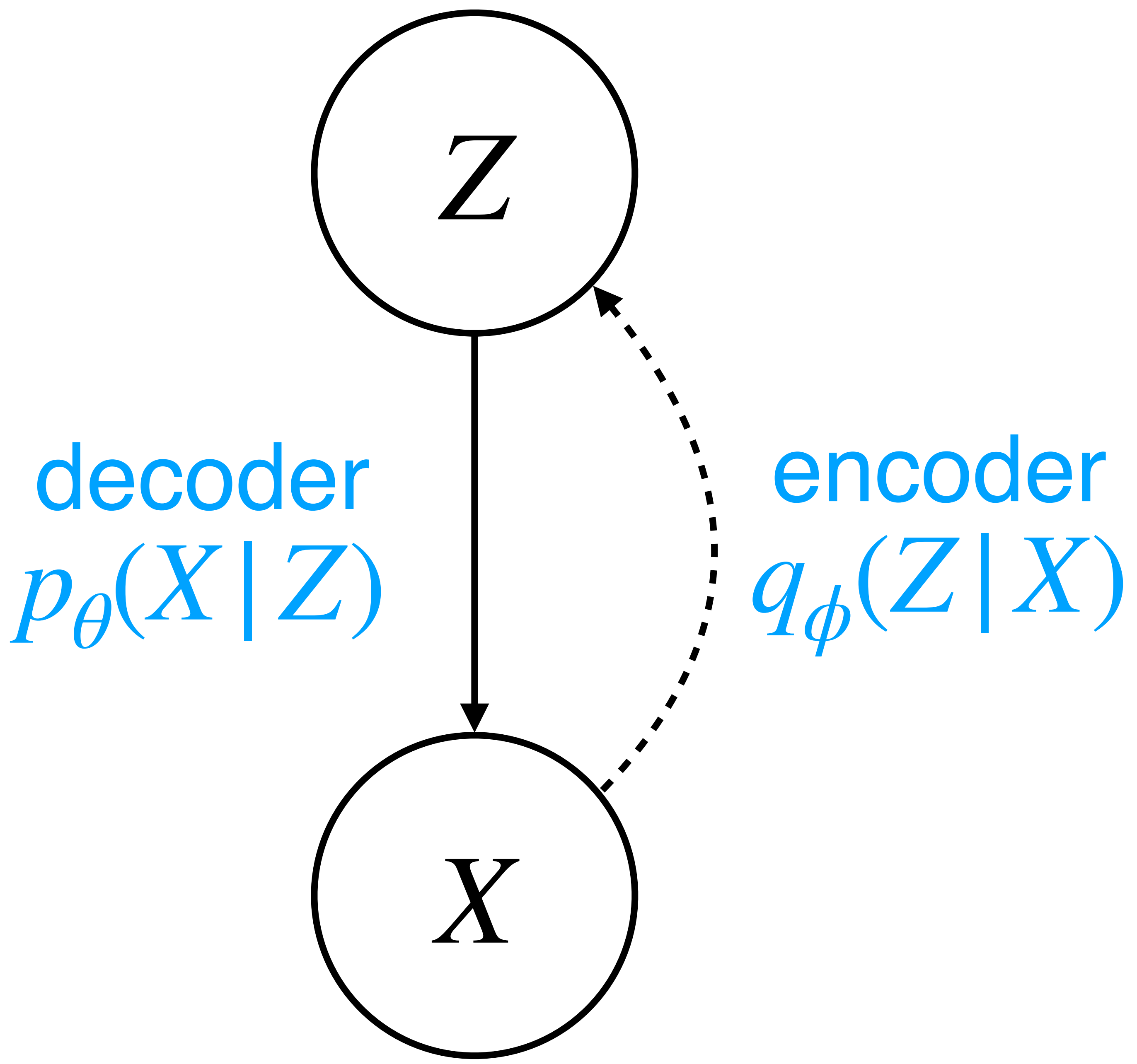
Important: no extra evaluation of f

- Replace both "local" and "global" control variates of double CV using discrete Stein operators.

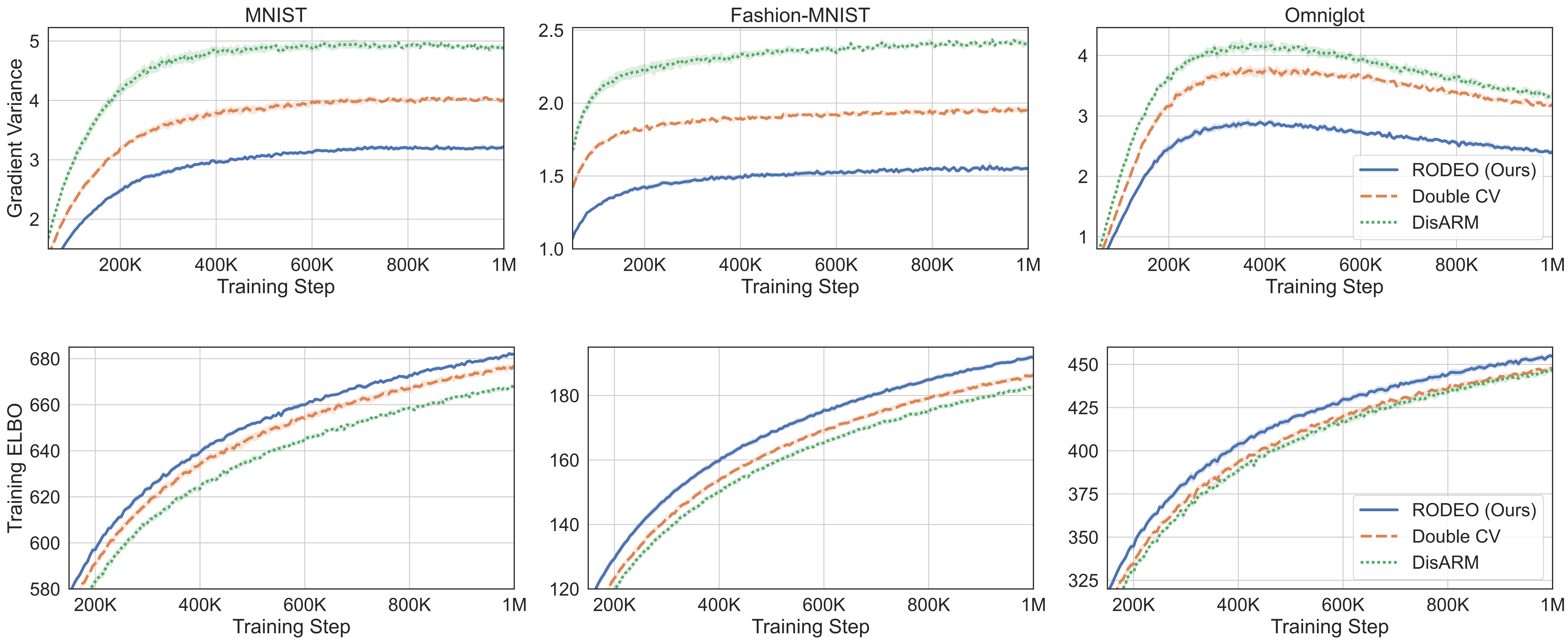
Benchmark: Training Binary Latent VAEs

- Latent-variable model: $p_\theta(X, Z) = p_\theta(X \mid Z)p(Z)$
- Maximizing a lower bound of the log marginal likelihood:

$$\log p_\theta(x) = \log \mathbb{E}_{q_\phi(z|x)}\left[\frac{p_\theta(x, z)}{q_\phi(z \mid x)}\right] \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x \mid z)p(z)}{q_\phi(z \mid x)}\right]$$

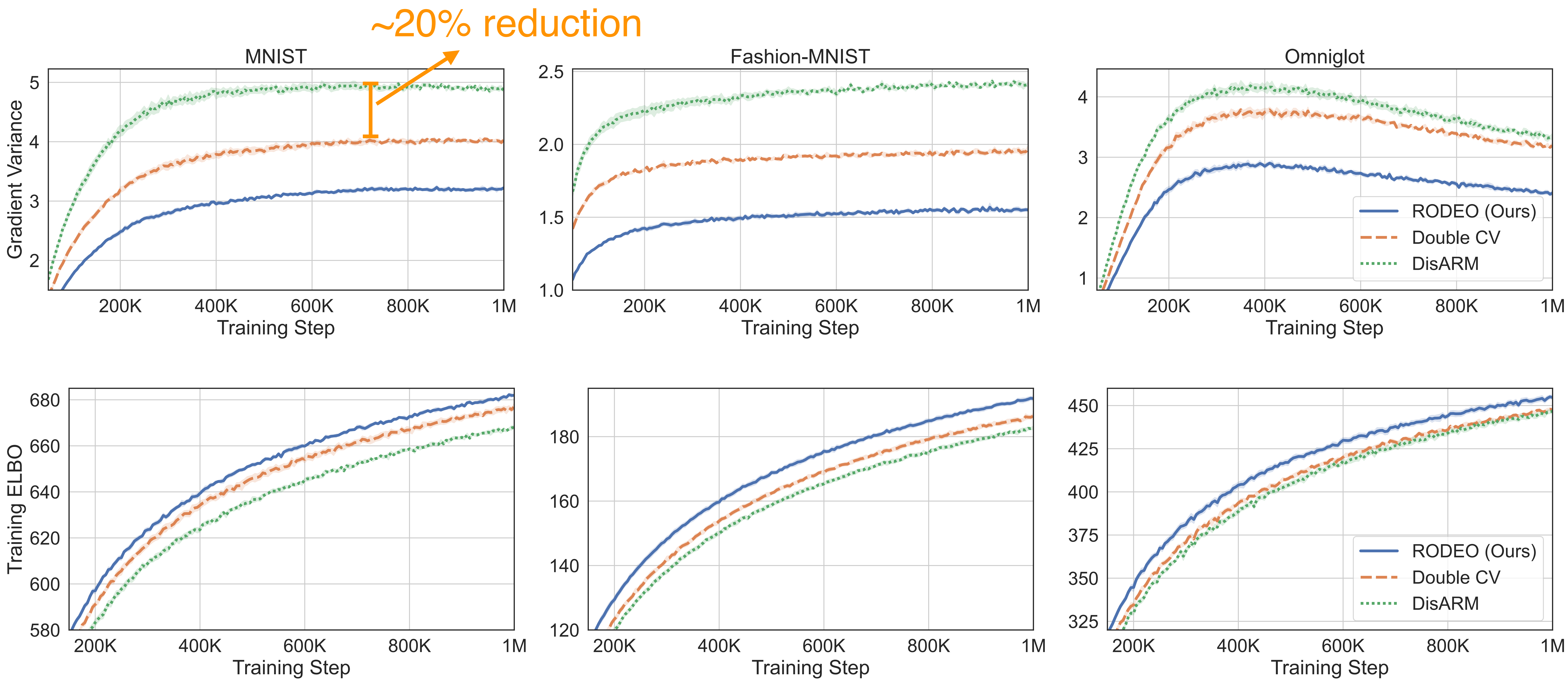


Benchmark: Training Binary Latent VAEs



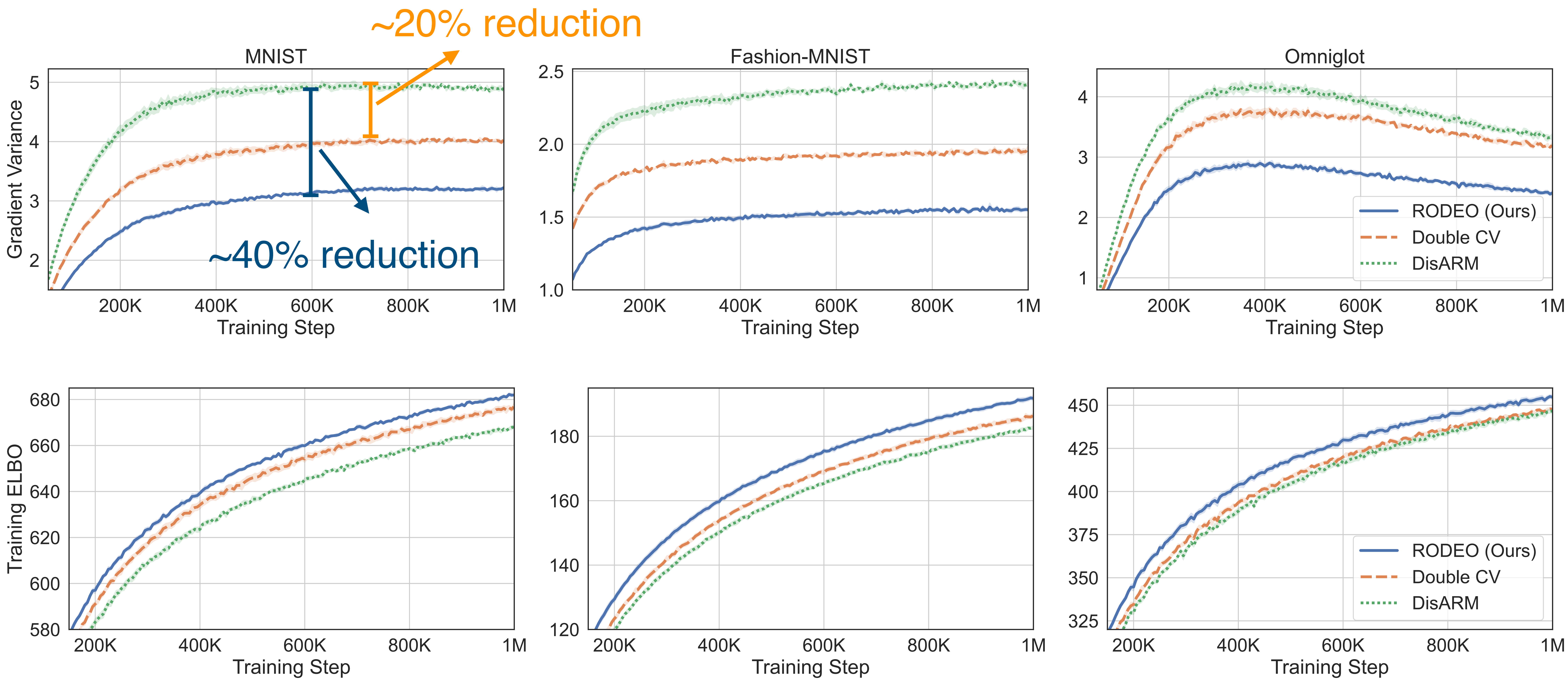
$$K = 2, d = 200$$

Benchmark: Training Binary Latent VAEs



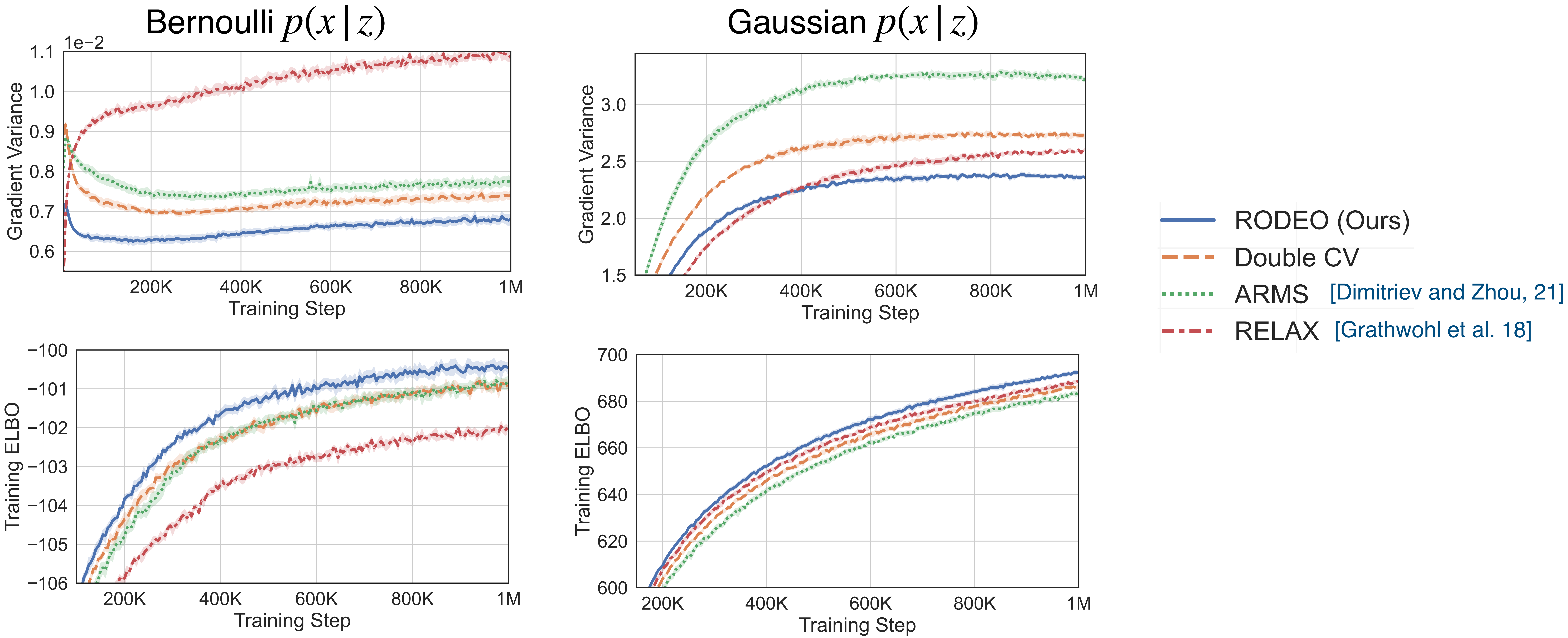
$K = 2, d = 200$

Benchmark: Training Binary Latent VAEs



$$K = 2, d = 200$$

Benchmark: Training Binary Latent VAEs



RELAX needs three evaluations of f, $K = 3$ for other estimators

Today's Talk

Gradient estimation for differentiable programming in probabilistic models

- Gradient Estimation for Discrete Expectations
  - Titsias & **Shi**. (AISTATS'22)
  - **Shi**, Zhou, Hwang, Titsias & Mackey. (In Submission)
- Gradient Estimation for Intractable Densities
  - **Shi**, Sun & Zhu. (ICML'18)
  - Zhou, **Shi** & Zhu. (ICML'20)

Motivation

A difficult example in representation learning

representation — data

$Y \longleftarrow \boxed{\text{Encoder NN}_{\phi}} \longleftarrow X \sim P_X$

[Hjelm et al., 19; Tschannen et al., 19]

Motivation

A difficult example in representation learning

representation ... data

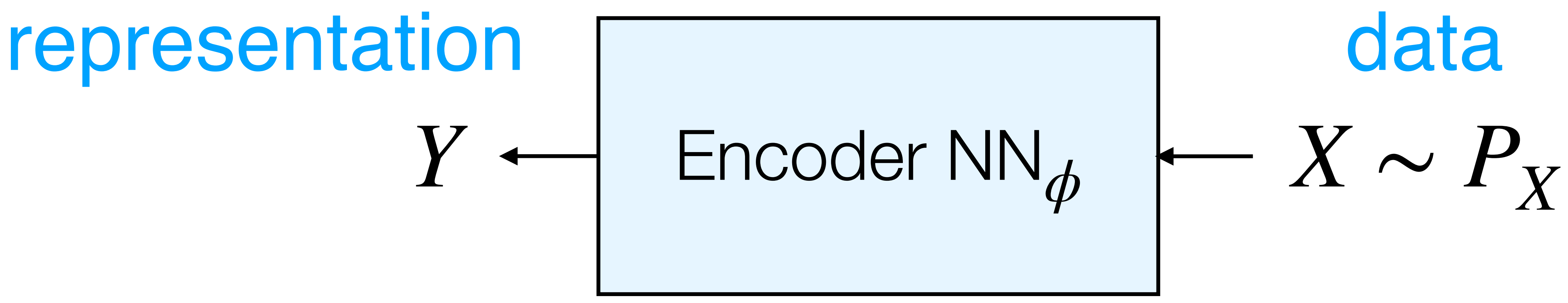
$$Y \longleftarrow \boxed{\text{Encoder NN}_{\phi}} \longleftarrow X \sim P_X$$

[Hjelm et al., 19; Tschannen et al., 19]

- learn by maximizing mutual information:

Motivation

A difficult example in representation learning



[Hjelm et al., 19; Tschannen et al., 19]

- learn by maximizing mutual information:

$$\max_{\phi} I(X, Y) := \mathrm{KL}(P_{X,Y} \| P_X \otimes P_Y)$$

Motivation

A difficult example in representation learning

[Hjelm et al., 19; Tschannen et al., 19]

representation — data

$$Y \longleftarrow \boxed{\text{Encoder NN}_\phi} \longleftarrow X \sim P_X$$

- learn by maximizing mutual information:

$$\max_\phi I(X, Y) := \mathrm{KL}(P_{X,Y} \| P_X \otimes P_Y)$$

$$\mathbb{E}_{P_{X,Y}}\left[\log \frac{p_{X,Y}}{p_X p_Y}\right]$$

Motivation

A difficult example in representation learning

representation		data
$Y \longleftarrow$	Encoder NN$_\phi$	$\longleftarrow X \sim P_X$

[Hjelm et al., 19; Tschannen et al., 19]

- learn by maximizing mutual information:

$$\max_{\phi} I(X, Y) := \mathrm{KL}(P_{X,Y} \| P_X \otimes P_Y)$$

$$\mathbb{E}_{P_{X,Y}}\left[\log \frac{p_{X,Y}}{p_X p_Y}\right]$$

- Often no explicit p_X, and $p_Y, p_{X,Y}$ are intractable

Motivation

A difficult example in representation learning

representation | | data

$$Y \longleftarrow \boxed{\text{Encoder NN}_{\phi}} \longleftarrow X \sim P_X$$

[Hjelm et al., 19; Tschannen et al., 19]

- learn by maximizing mutual information:

$$\mathbb{E}_{P_{X,Y}}\left[\log \frac{p_{X,Y}}{p_X p_Y}\right]$$

$$\max_{\phi} I(X,Y) := \mathrm{KL}(P_{X,Y} \| P_X \otimes P_Y)$$

- Often no explicit p_X, and $p_Y, p_{X,Y}$ are intractable
- Prior estimators assume L is computable in $\nabla_{\phi}\mathbb{E}[L(f(X))]$

Motivation

Gradient estimation for KL-divergence

Motivation

Gradient estimation for KL-divergence

$$\nabla_\phi \mathrm{KL}(q_\phi \| p)$$

Motivation

Gradient estimation for KL-divergence

- $q_\phi(\mathbf{x}), p(\mathbf{x})$ are intractable
- easy access to the sample of q_ϕ through $\epsilon \sim \nu$, $\mathbf{x} = g_\phi(\epsilon)$

$$\nabla_\phi \mathrm{KL}(q_\phi \| p)$$

Motivation

Gradient estimation for KL-divergence

- $q_\phi(\mathbf{x}), p(\mathbf{x})$ are intractable
- easy access to the sample of q_ϕ through $\epsilon \sim \nu$, $\mathbf{x} = g_\phi(\epsilon)$

$$\nabla_\phi \mathrm{KL}(q_\phi \| p) = \mathbb{E}_{\epsilon\sim\nu}[\nabla \log q(\mathbf{x}) \nabla_\phi g_\phi(\epsilon)] - \mathbb{E}_{\epsilon\sim\nu}[\nabla \log p(\mathbf{x}) \nabla_\phi g_\phi(\epsilon)]$$

Motivation

Gradient estimation for KL-divergence

- $q_\phi(\mathbf{x}), p(\mathbf{x})$ are intractable
- easy access to the sample of q_ϕ through $\epsilon \sim \nu$, $\mathbf{x} = g_\phi(\epsilon)$

$$\nabla_\phi \mathrm{KL}(q_\phi \| p) = \mathbb{E}_{\epsilon\sim\nu}[\nabla \log q(\mathbf{x}) \nabla_\phi g_\phi(\epsilon)] - \mathbb{E}_{\epsilon\sim\nu}[\nabla \log p(\mathbf{x}) \nabla_\phi g_\phi(\epsilon)]$$

Score function

Motivation

Gradient estimation for KL-divergence

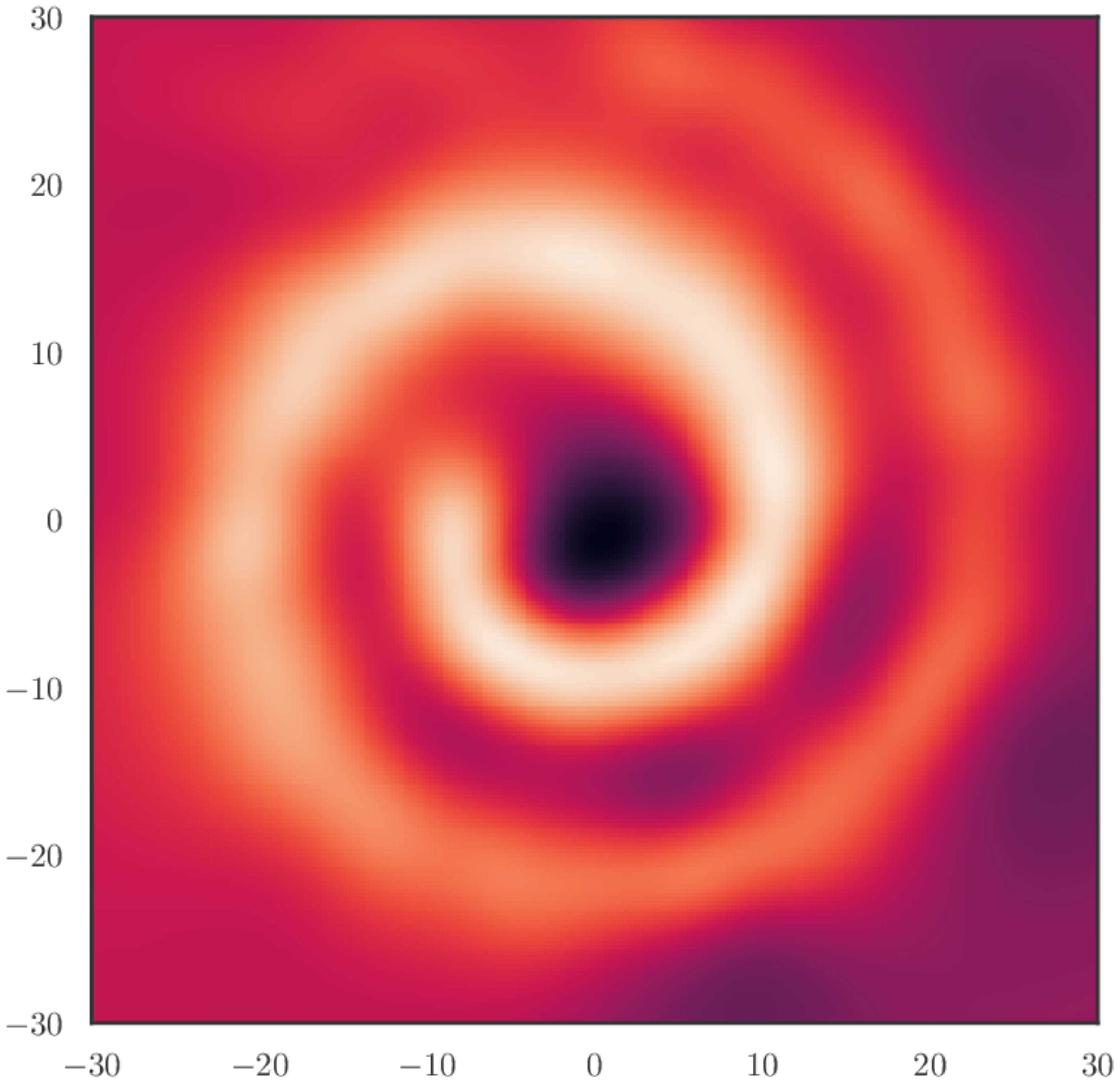
- $q_\phi(\mathbf{x}), p(\mathbf{x})$ are intractable
- easy access to the sample of q_ϕ through $\epsilon \sim \nu$, $\mathbf{x} = g_\phi(\epsilon)$

$$\nabla_\phi \mathrm{KL}(q_\phi \| p) = \mathbb{E}_{\epsilon\sim\nu}[\nabla \log q(\mathbf{x}) \nabla_\phi g_\phi(\epsilon)] - \mathbb{E}_{\epsilon\sim\nu}[\nabla \log p(\mathbf{x}) \nabla_\phi g_\phi(\epsilon)]$$

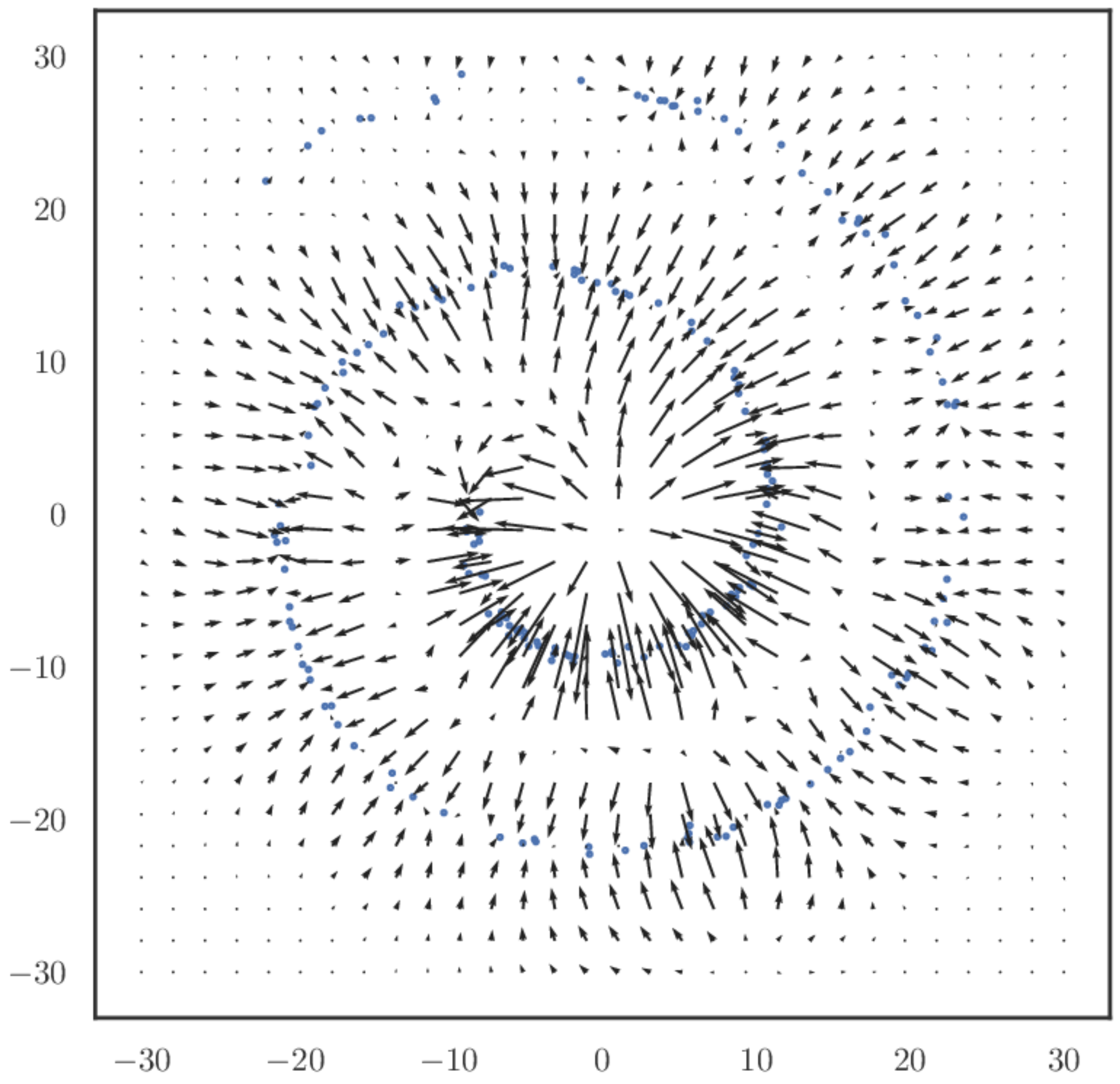
Score function

$$\{\mathbf{x}^j\}_{j=1}^M \overset{\text{i.i.d.}}{\sim} q \longrightarrow \nabla \log q(\mathbf{x})$$

Score Estimation



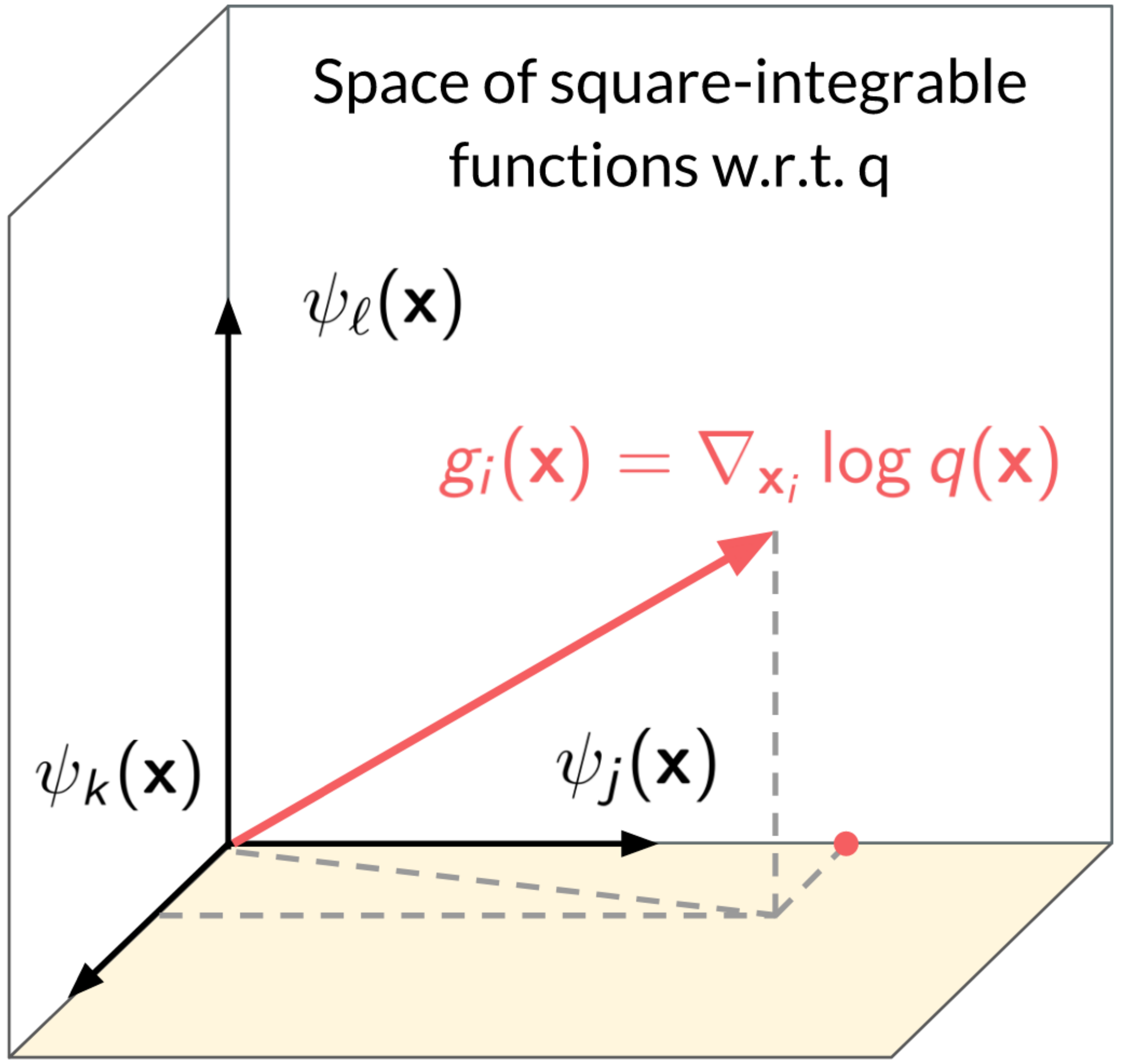
$q(\mathbf{x})$



$\nabla \log q(\mathbf{x})$

A Spectral Estimator

Main result



$$\mathbb{E}_{\mathbf{x}' \sim q}[k(\mathbf{x}, \mathbf{x}')\psi_j(\mathbf{x}')] = \lambda_j \psi_j(\mathbf{x})$$

- Under mild conditions

$$\nabla_{x_i} \log q(\mathbf{x}) = -\sum_{j \geq 1} \mathbb{E}_q \left[\nabla_{x_i} \psi_j(\mathbf{x}) \right] \psi_j(\mathbf{x})$$

- Nyström methods for estimating ψ_j and its derivatives
- Truncating the series at small eigenvalues

Shi et al. A spectral approach to gradient estimation for implicit distributions. ICML 2018

A Spectral Estimator

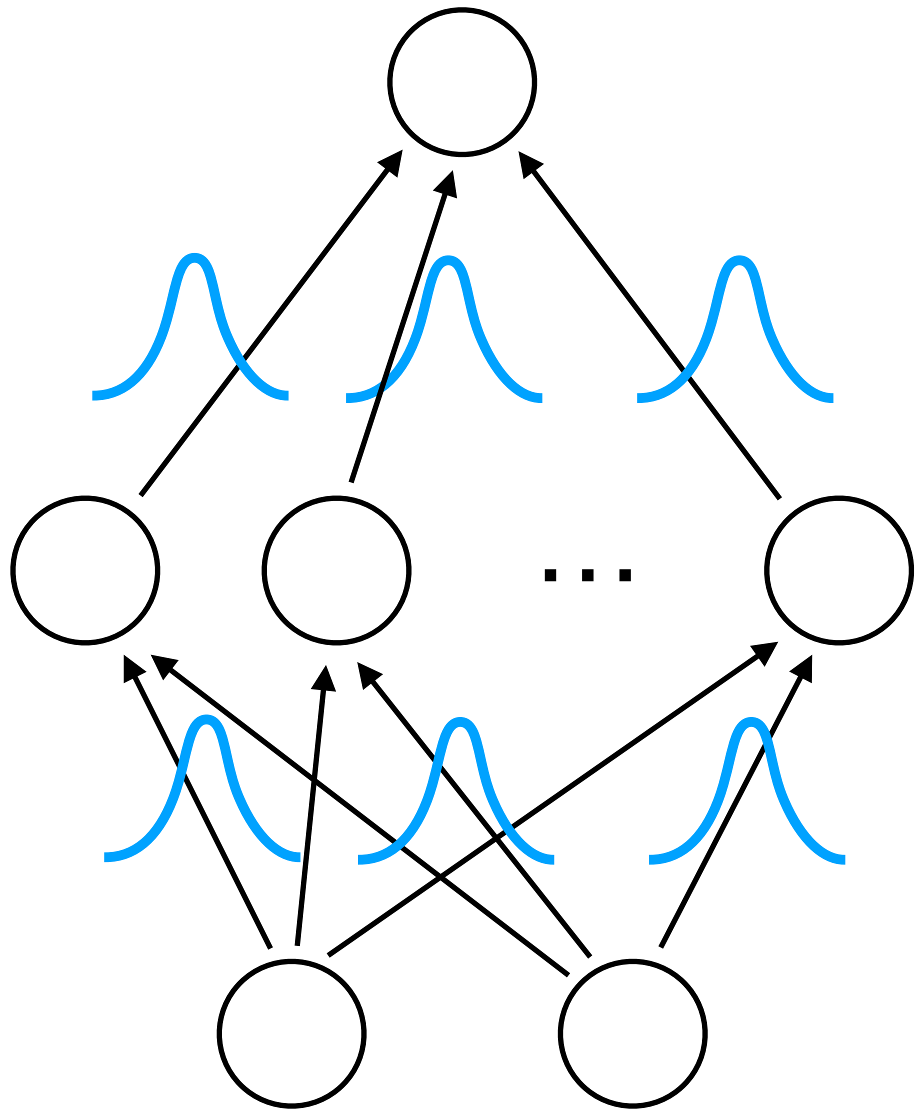
Properties

	Alain & Bengio, 14	Sriperumbudur et al., 13	Li & Turner, 17	This work
Closed-form	**N**	**Y**	**Y**	**Y**
Complexity scales linearly w/ d	**Y**	**N**	**Y**	**Y**
Principled out-of-sample prediction	**Y**	**Y**	**N**	**Y**
Convergence rates	-	[1/4, 1/3]	-	[1/4, 1/2)
	need training	cubic scaling	only in-sample prediction	

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



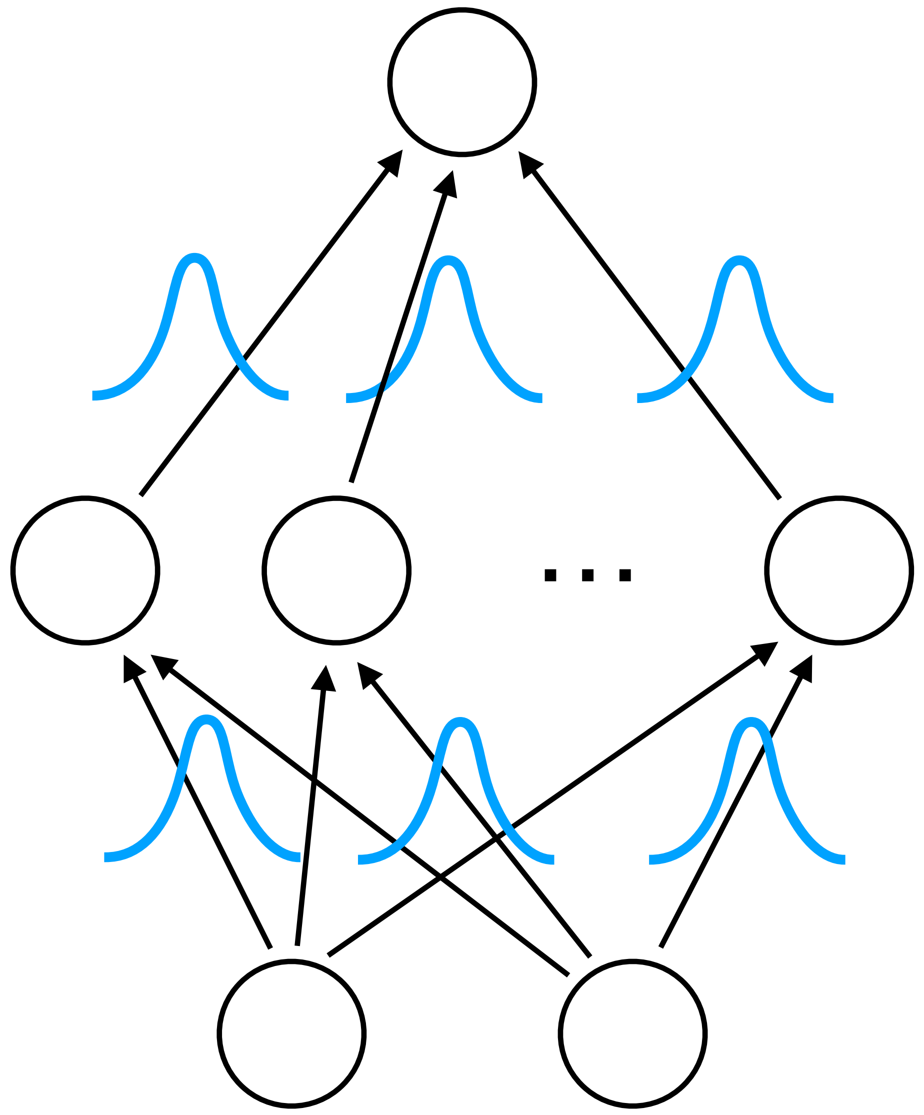
$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

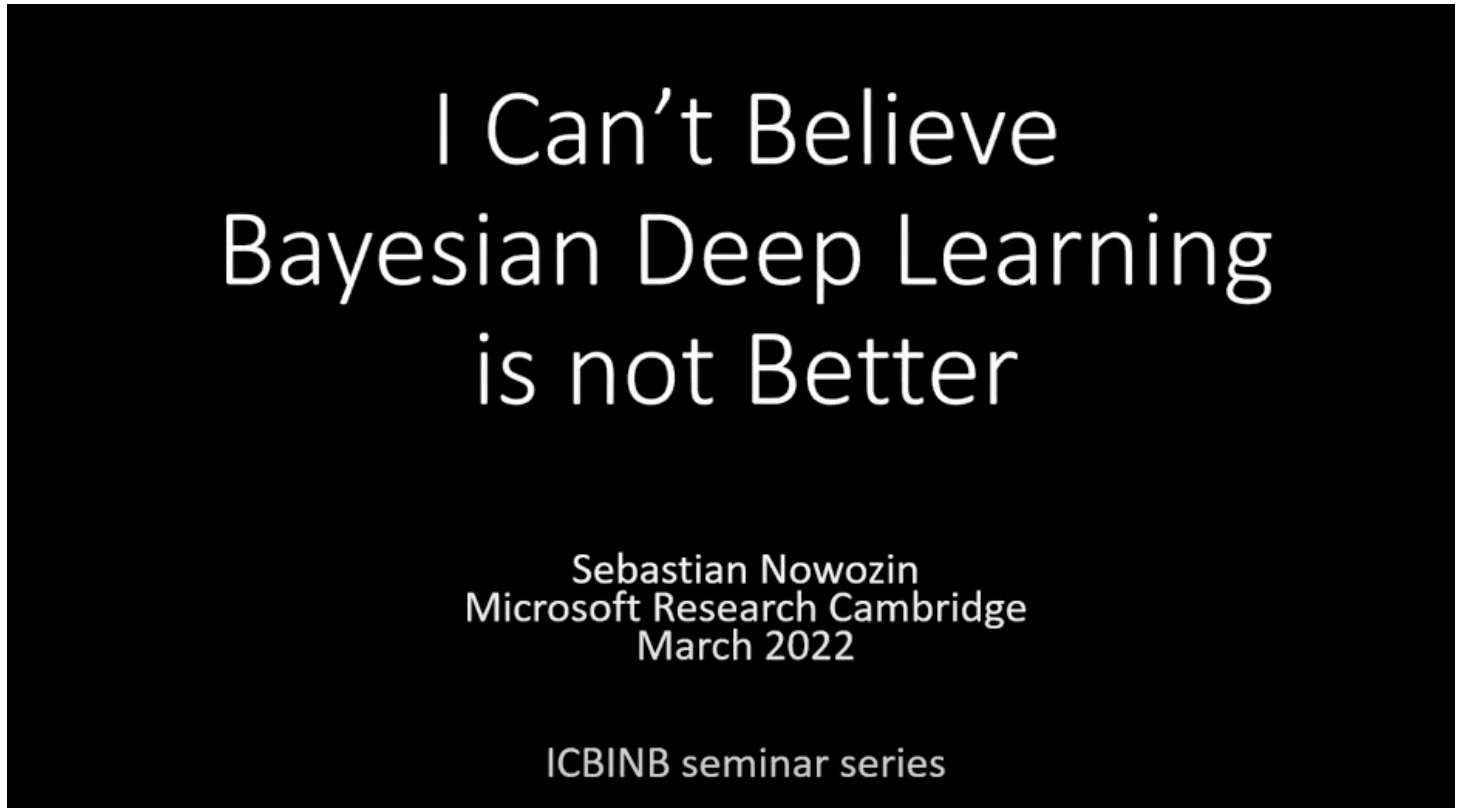
Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w})) p(\mathbf{w})$$

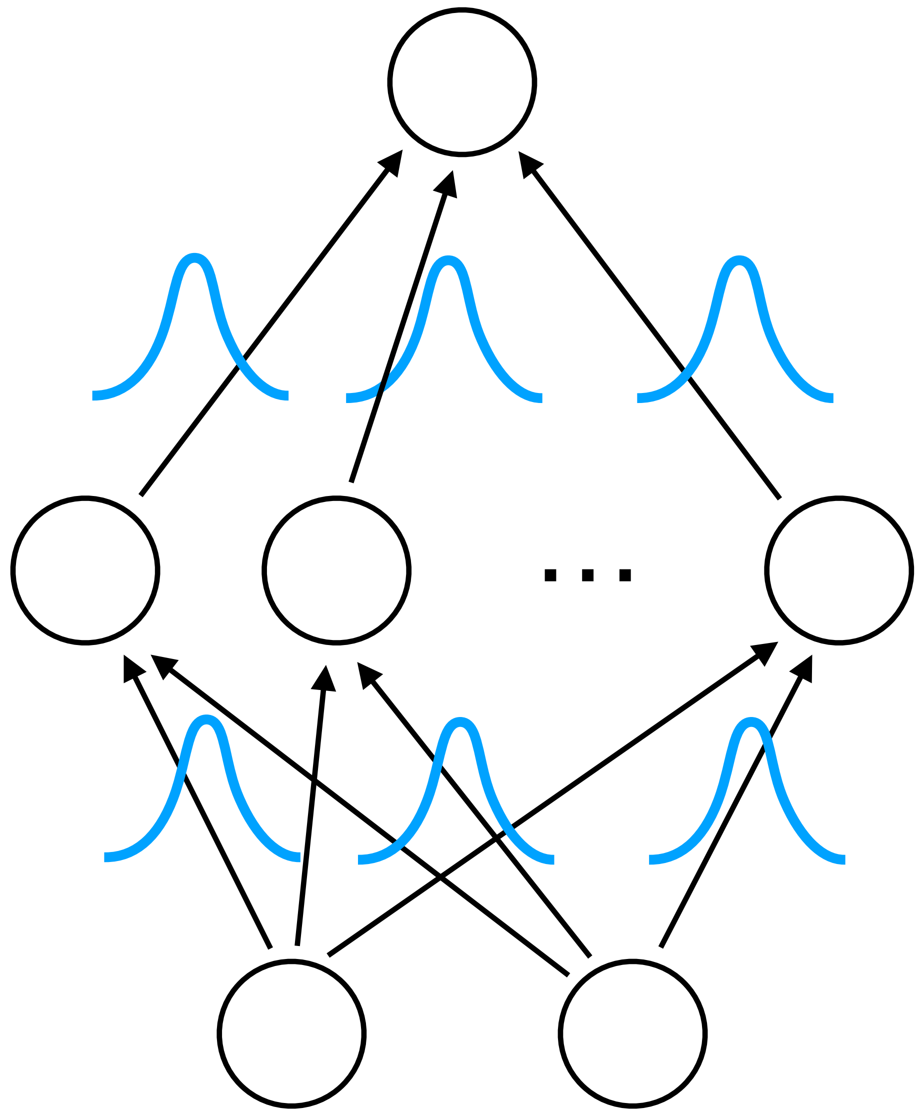


Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

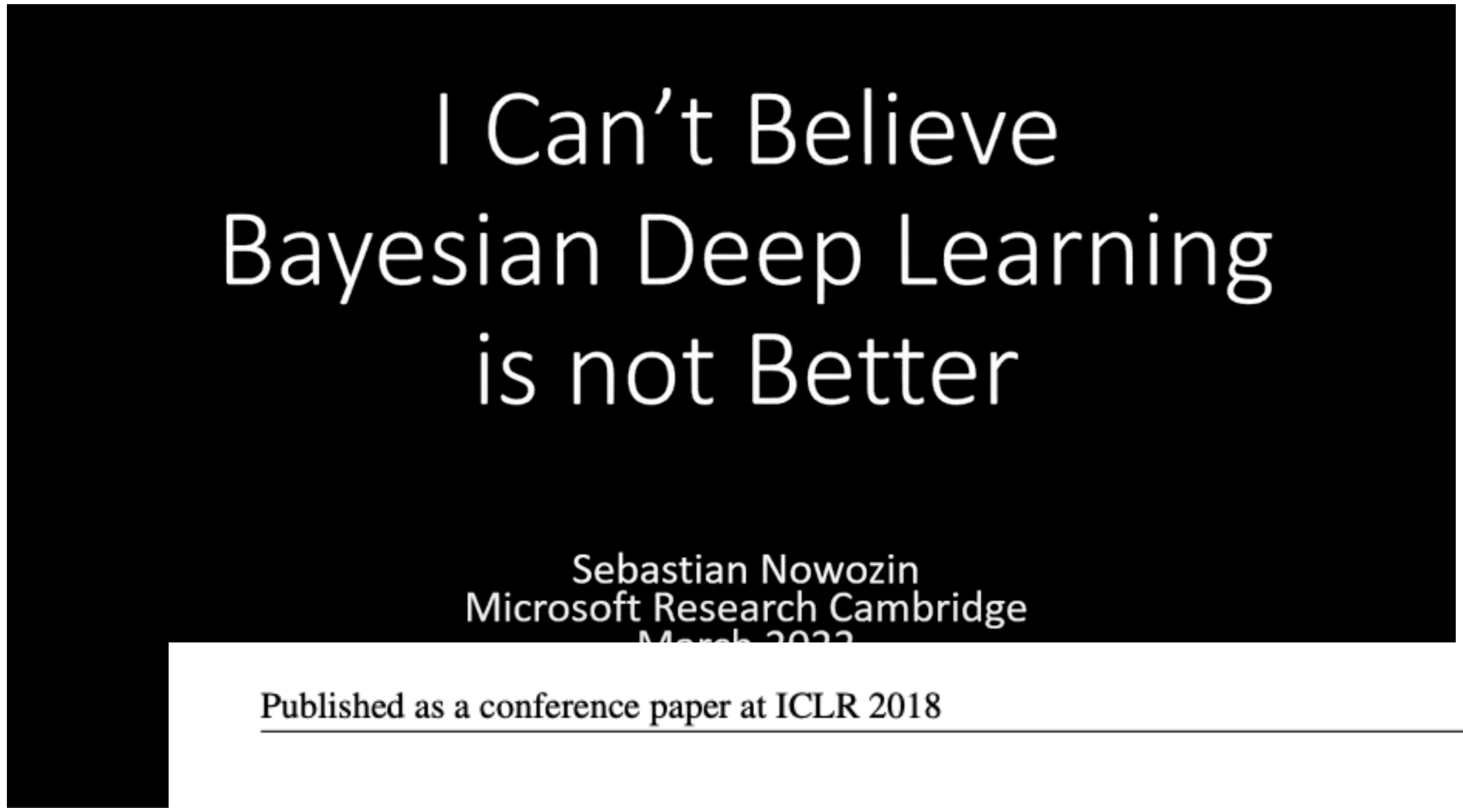
Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks

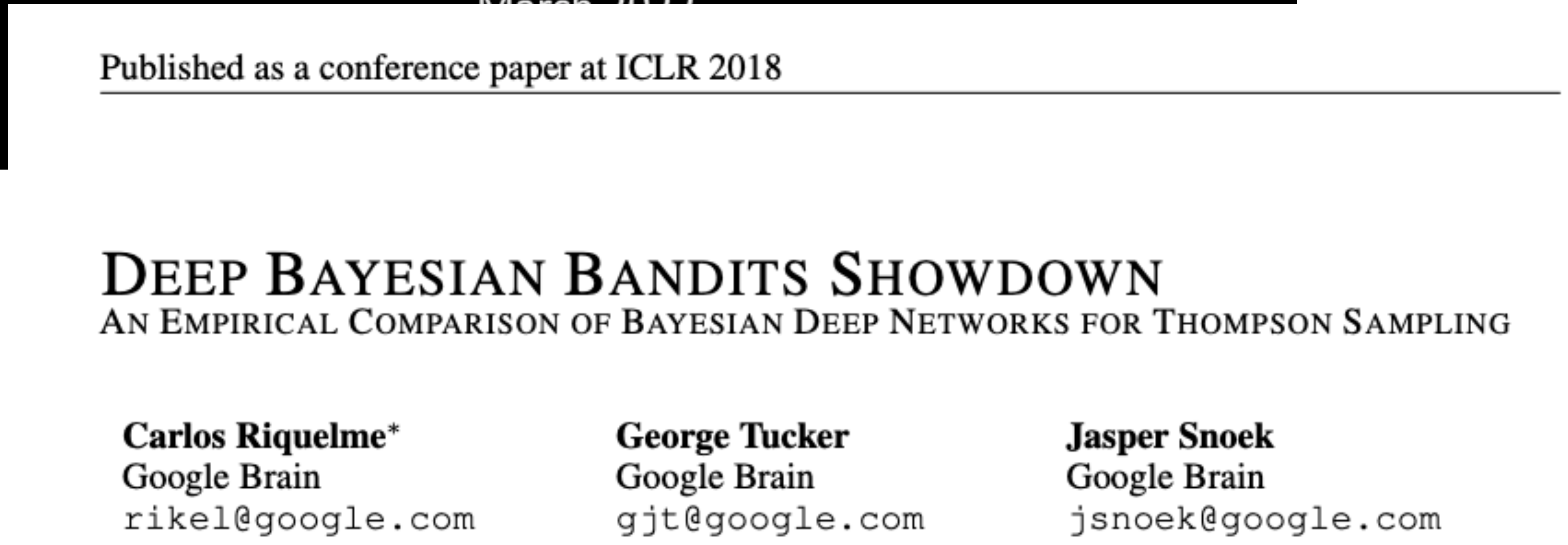


$$p(w \,|\, \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \,|\, f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$



I Can't Believe
Bayesian Deep Learning
is not Better

Sebastian Nowozin
Microsoft Research Cambridge



Published as a conference paper at ICLR 2018

DEEP BAYESIAN BANDITS SHOWDOWN

AN EMPIRICAL COMPARISON OF BAYESIAN DEEP NETWORKS FOR THOMPSON SAMPLING

Carlos Riquelme* Google Brain rikel@google.com

George Tucker Google Brain gjt@google.com

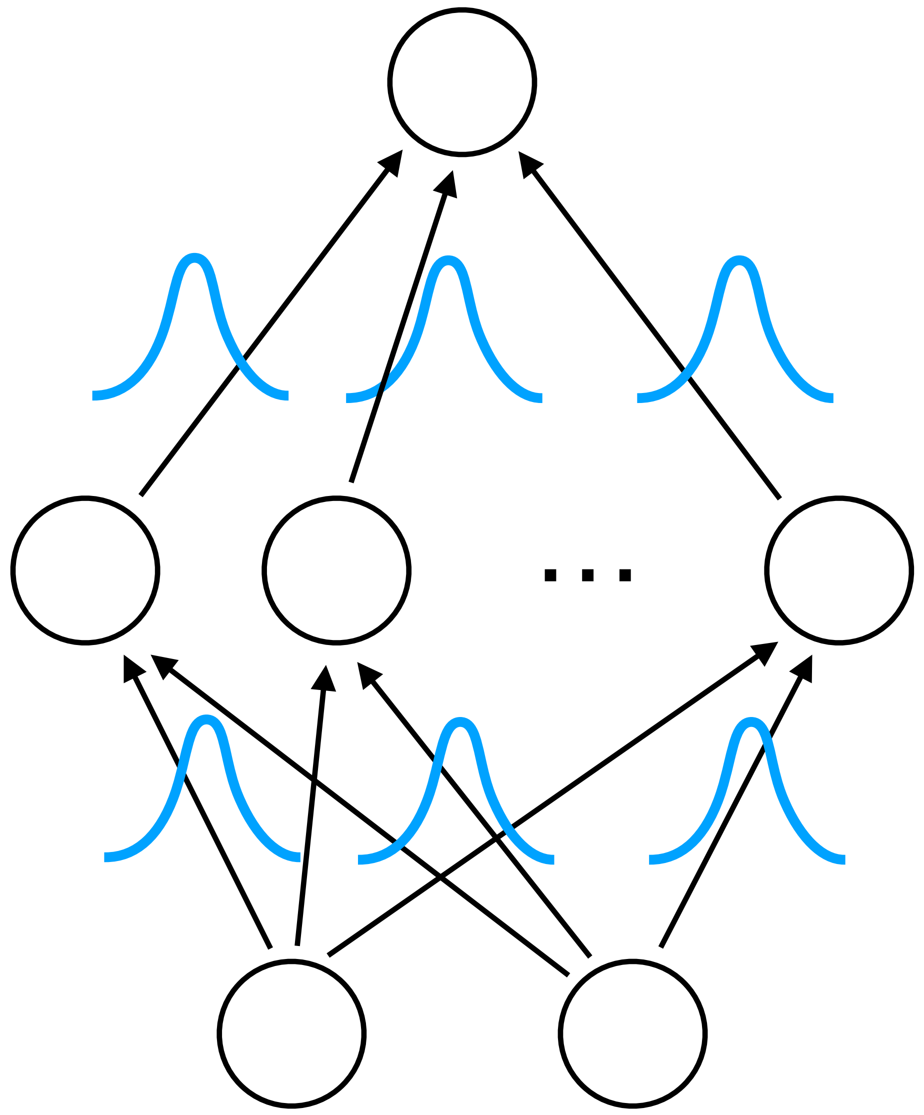
Jasper Snoek Google Brain jsnoek@google.com

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$

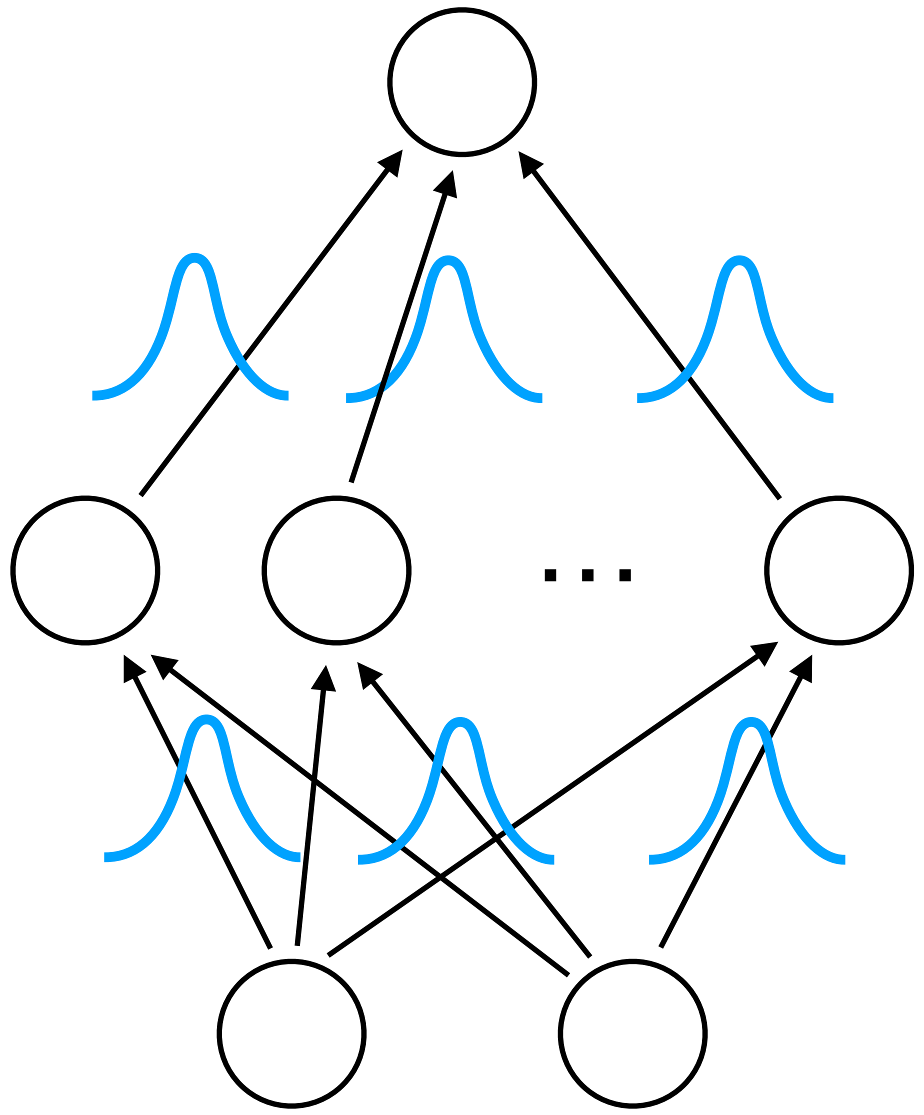
- Problems of weight-space inference:

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$

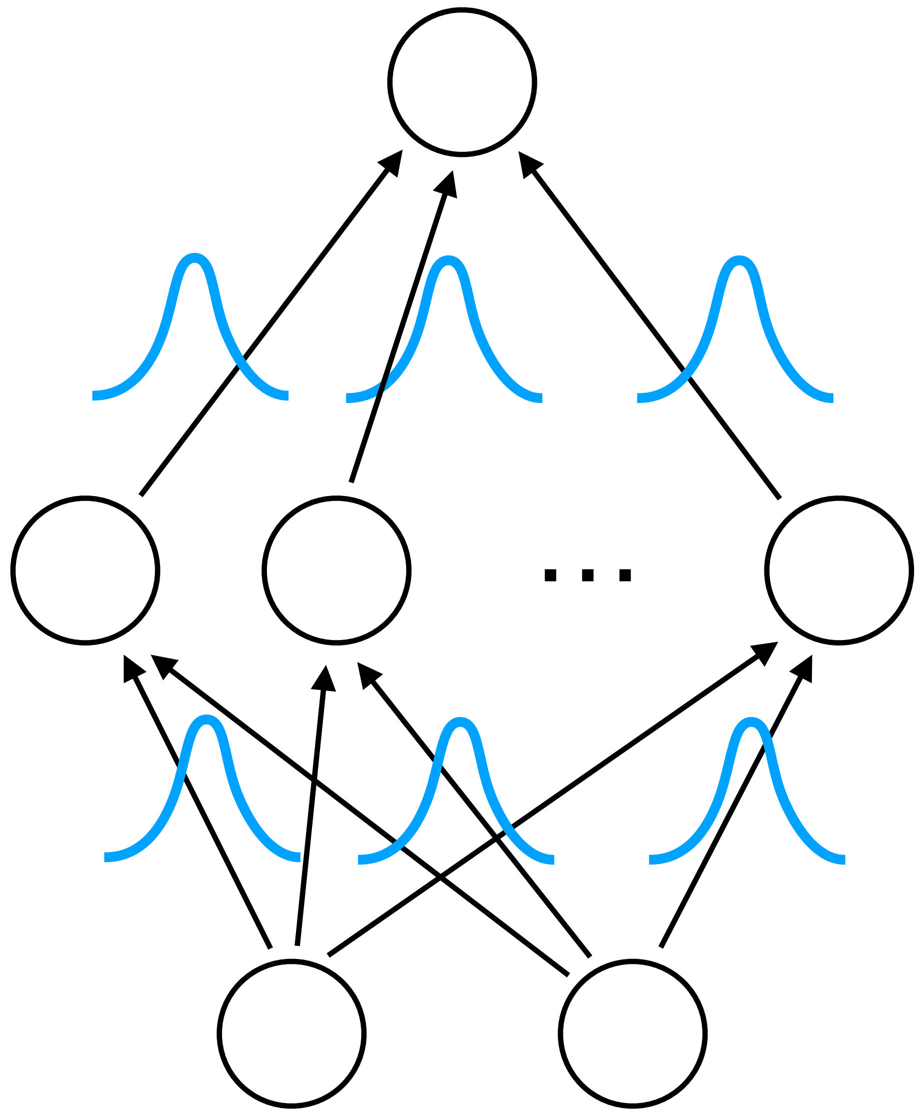
- Problems of weight-space inference:
  - Weights have no meaning, non-identifiable

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w})) p(\mathbf{w})$$

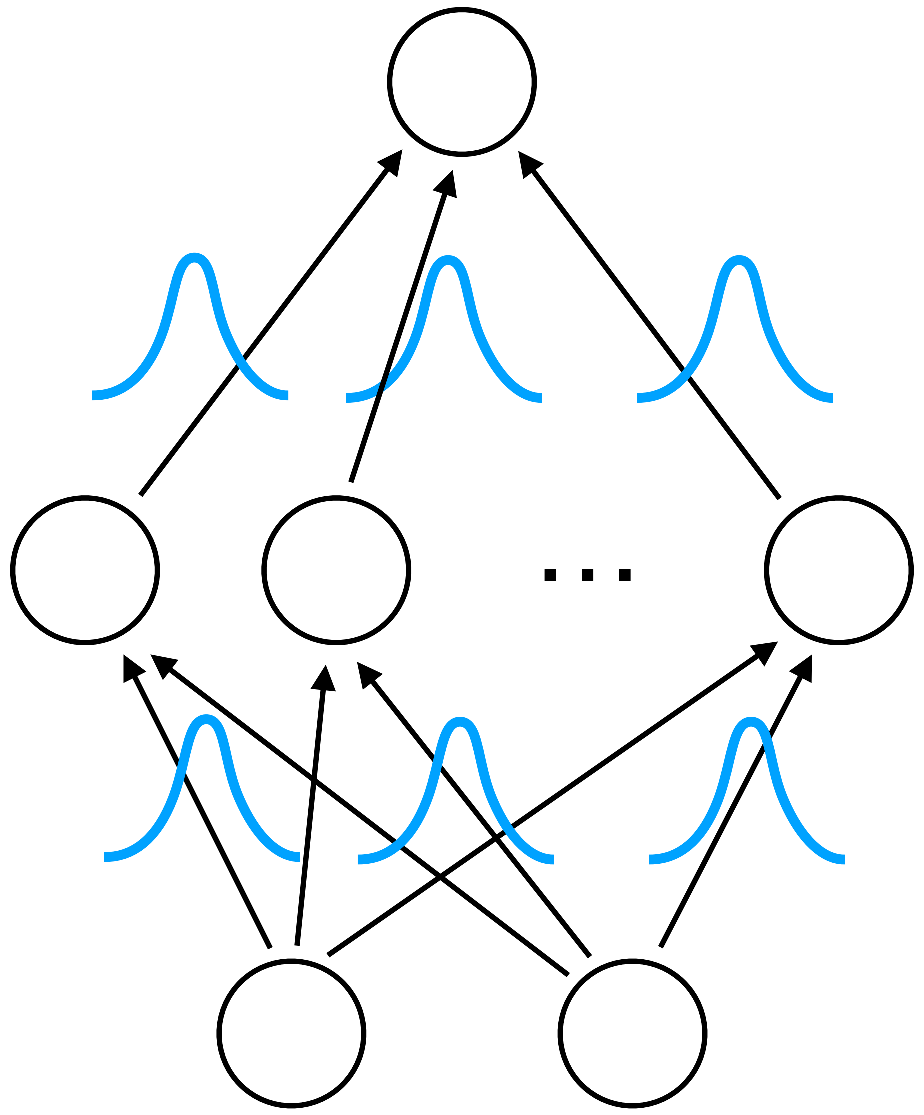
- Problems of weight-space inference:
  - Weights have no meaning, non-identifiable
  - hard to specify meaningful priors

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$

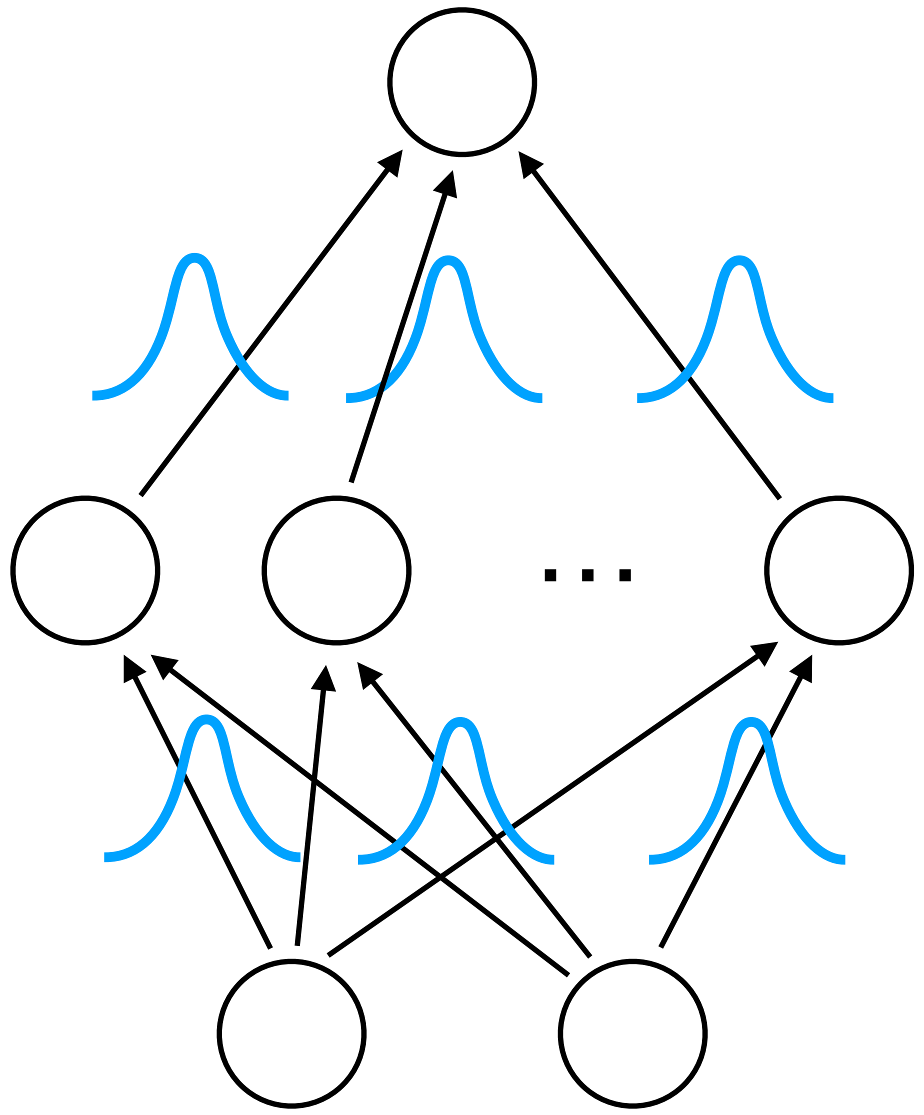
- Problems of weight-space inference:
  - Weights have no meaning, non-identifiable
  - hard to specify meaningful priors
- Function space inference:

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$

- Problems of weight-space inference:
  - Weights have no meaning, non-identifiable
  - hard to specify meaningful priors
- Function space inference:

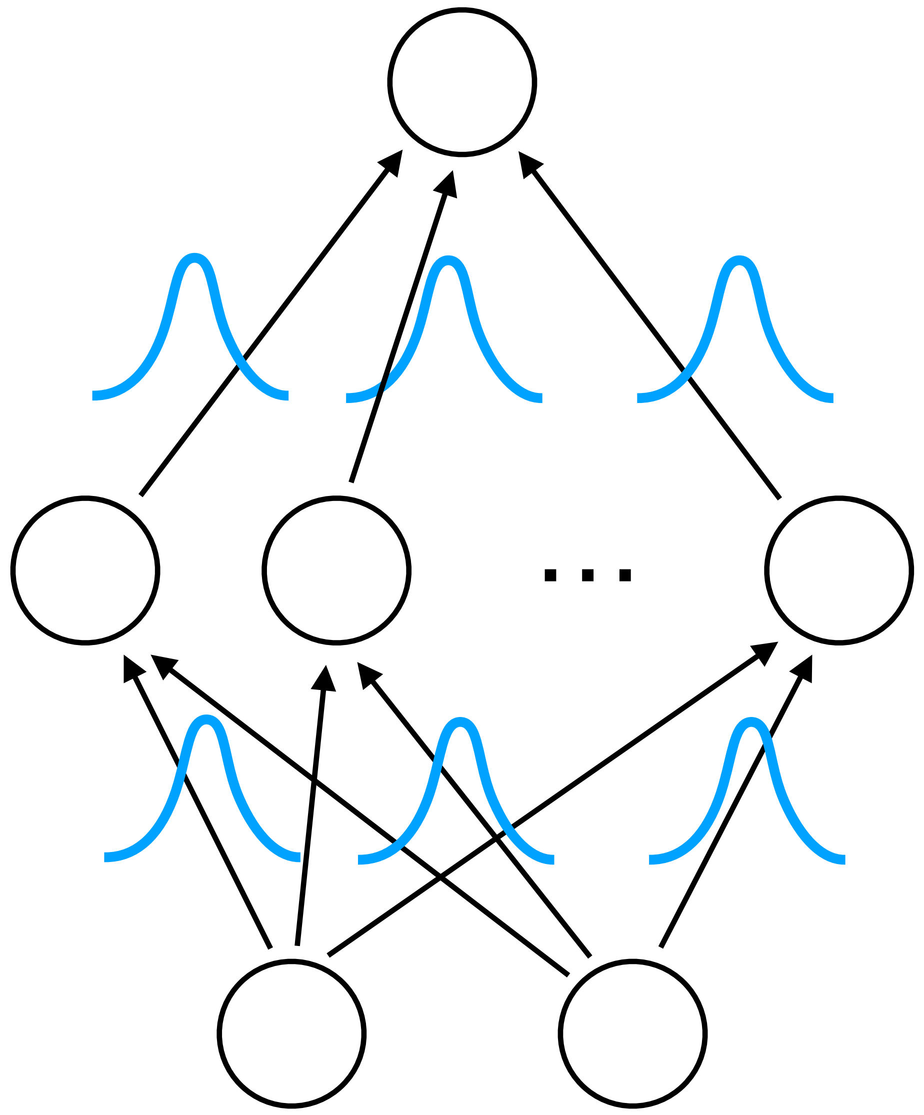
$$\mathbb{E}_{q_\phi(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})] - \mathrm{KL}(q_\phi(\mathbf{f}) \| p(\mathbf{f}))$$

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w})) p(\mathbf{w})$$

- Problems of weight-space inference:
  - Weights have no meaning, non-identifiable
  - hard to specify meaningful priors
- Function space inference:

$$\mathbb{E}_{q_\phi(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})] - \mathrm{KL}(q_\phi(\mathbf{f}) \| p(\mathbf{f}))$$

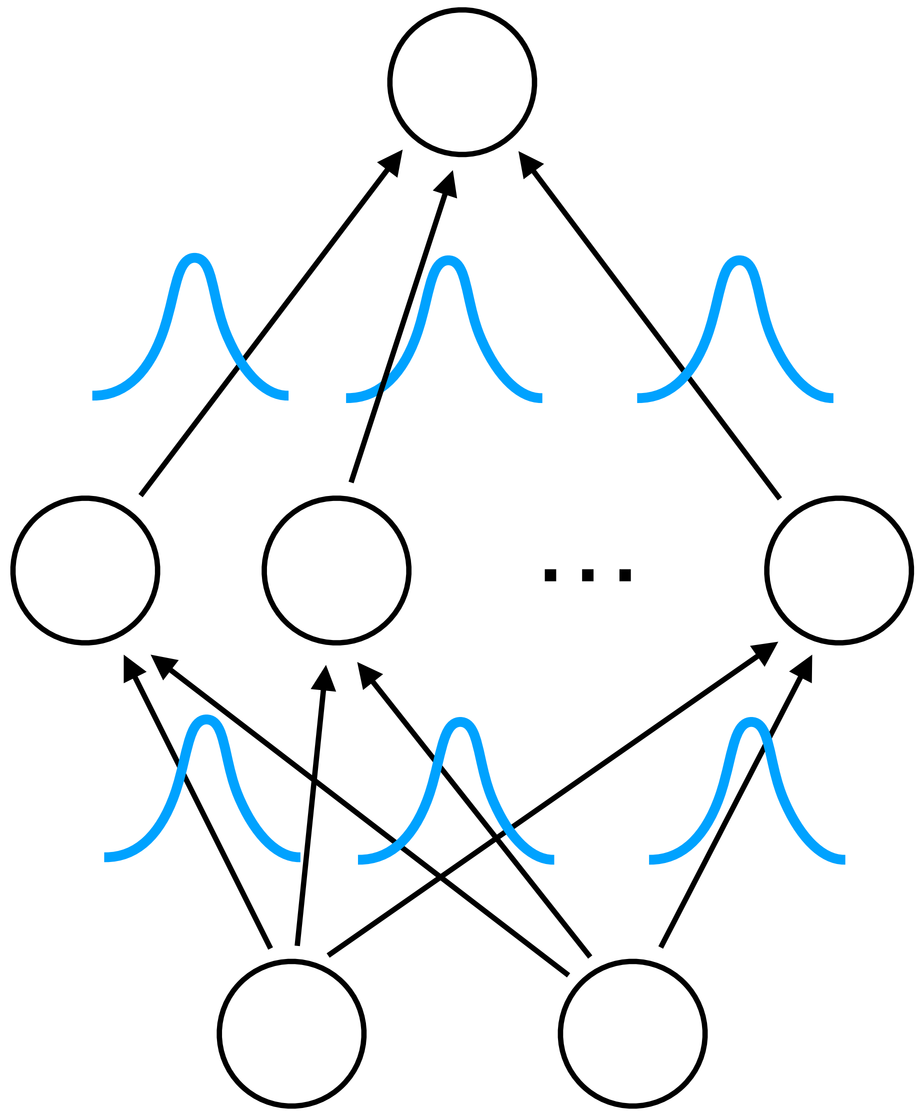
$q_\phi(\mathbf{f})$: induced by $q_\phi(\mathbf{w})$ through $\mathbf{f} = f(\mathbf{X}; \mathbf{w})$

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

Applications

Functional Bayesian Neural Networks

Bayesian Neural Networks



$$p(w \mid \mathbf{X}, \mathbf{y}) \propto \prod_{i=1}^{n} p(y_i \mid f(\mathbf{x}_i; \mathbf{w}))p(\mathbf{w})$$

- Problems of weight-space inference:
  - Weights have no meaning, non-identifiable
  - hard to specify meaningful priors
- Function space inference:

$$\mathbb{E}_{q_\phi(\mathbf{f})}[\log p(\mathbf{y} \mid \mathbf{f})] - \mathrm{KL}(q_\phi(\mathbf{f}) \| p(\mathbf{f}))$$

$q_\phi(\mathbf{f})$: induced by $q_\phi(\mathbf{w})$ through $\mathbf{f} = f(\mathbf{X}; \mathbf{w})$

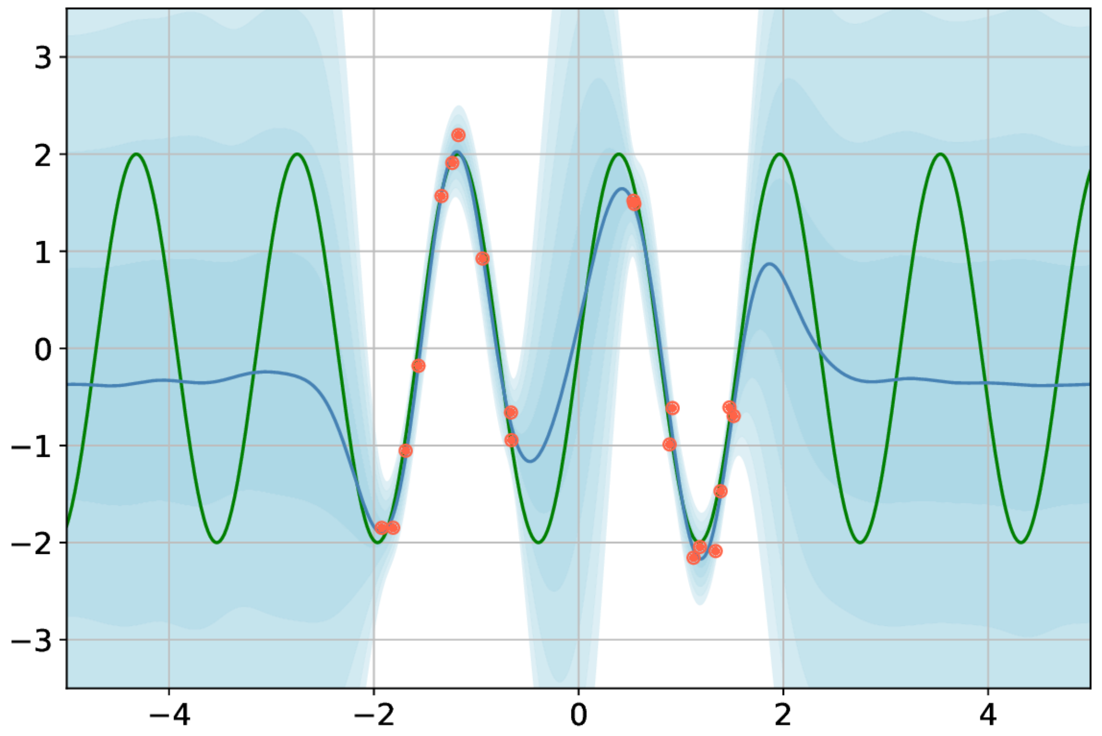
intractable

Sun*, Zhang*, **Shi***, Grosse. Functional variational Bayesian neural networks. ICLR 2019

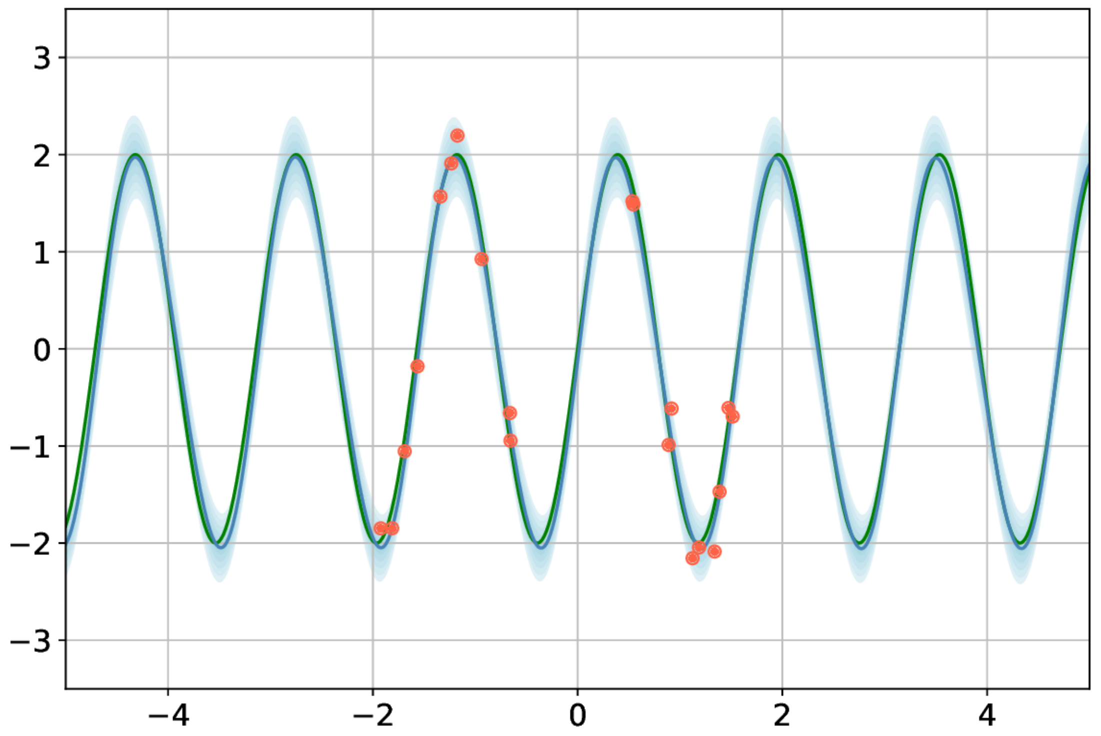
Applications

Functional Bayesian Neural Networks

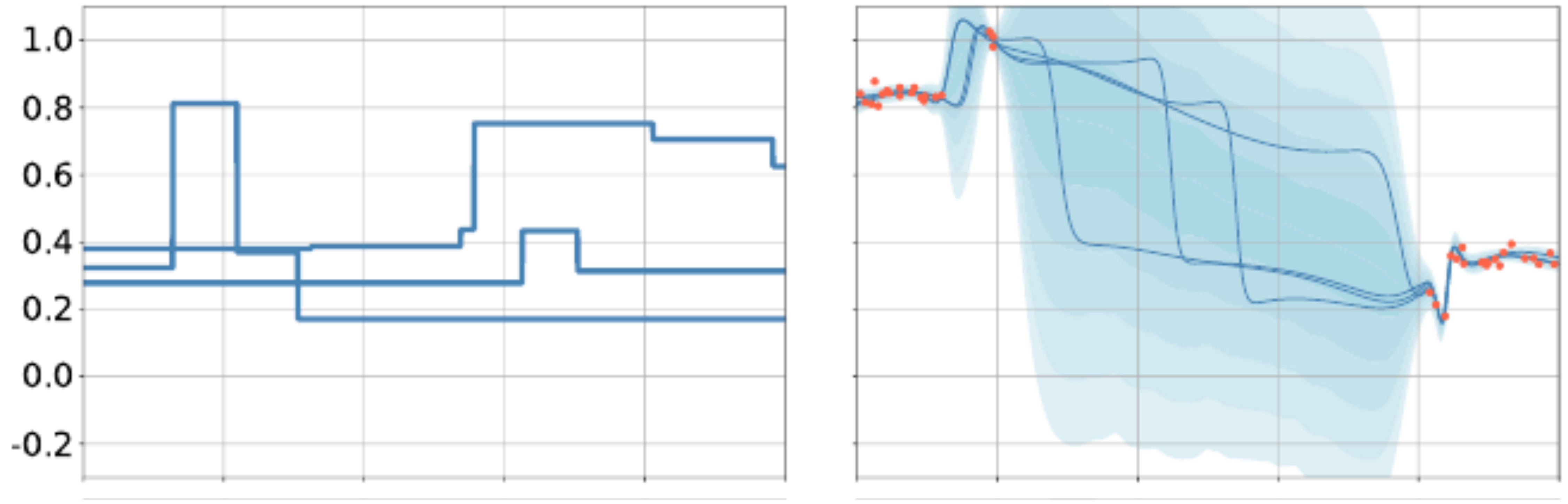
Locally smooth



Periodic



Piecewise constant



	M. Rank	M. Value	MushRoom	Statlog	Covertype	Financial	Jester	Adult
FBNN 1×50	4.7	41.9	21.38 ± 7.00	8.85 ± 4.55	47.16 ± 2.39	9.90 ± 2.40	75.55 ± 5.51	$\mathbf{88.43 \pm 1.95}$
FBNN 2×50	6.5	43.0	24.57 ± 10.81	10.08 ± 5.66	49.04 ± 3.75	11.83 ± 2.95	73.85 ± 6.82	88.81 ± 3.29
FBNN 3×50	7	45.0	34.03 ± 13.95	7.73 ± 4.37	50.14 ± 3.13	14.14 ± 1.99	74.27 ± 6.54	89.68 ± 1.66
FBNN 1×500	**3.8**	41.3	21.90 ± 9.95	6.50 ± 2.97	47.45 ± 1.86	$\mathbf{7.83 \pm 0.77}$	74.81 ± 5.57	89.03 ± 1.78
FBNN 2×500	4.2	41.2	23.93 ± 11.59	7.98 ± 3.08	46.00 ± 2.01	10.67 ± 3.52	$\mathbf{68.88 \pm 7.09}$	89.70 ± 2.01
FBNN 3×500	4.2	**40.9**	19.07 ± 4.97	10.04 ± 5.09	$\mathbf{45.24 \pm 2.11}$	11.48 ± 2.20	69.42 ± 7.56	90.01 ± 1.70
MultitaskGP	4.3	41.7	20.75 ± 2.08	7.25 ± 1.80	48.37 ± 3.50	8.07 ± 1.13	76.99 ± 6.01	88.64 ± 3.20
BBB 1×50	10.8	52.7	24.41 ± 6.70	25.67 ± 3.46	58.25 ± 5.00	37.69 ± 15.34	75.39 ± 6.32	95.07 ± 1.57
BBB 1×500	13.7	66.2	26.41 ± 8.71	51.29 ± 11.27	83.91 ± 4.62	57.20 ± 7.19	78.94 ± 4.98	99.21 ± 0.79
BBAlphaDiv	15	83.8	61.00 ± 6.47	70.91 ± 10.22	97.63 ± 3.21	85.94 ± 4.88	87.80 ± 5.08	99.60 ± 1.06
ParamNoise	10	47.9	20.33 ± 13.12	13.27 ± 2.85	65.07 ± 3.47	17.63 ± 4.27	74.94 ± 7.24	95.90 ± 2.20
NeuralLinear	10.8	48.8	16.56 ± 11.60	13.96 ± 1.51	64.96 ± 2.54	18.57 ± 2.02	82.14 ± 3.64	96.87 ± 0.92
LinFullPost	8.3	46.0	14.71 ± 0.67	19.24 ± 0.77	58.69 ± 1.17	10.69 ± 0.92	77.76 ± 5.67	95.00 ± 1.26
Dropout	5.5	41.7	$\mathbf{12.53 \pm 1.82}$	12.01 ± 6.11	48.95 ± 2.19	14.64 ± 3.95	71.38 ± 7.11	90.62 ± 2.21
RMS	6.5	43.9	15.29 ± 3.06	11.38 ± 5.63	58.96 ± 4.97	10.46 ± 1.61	72.09 ± 6.98	95.29 ± 1.50
BootRMS	4.7	42.6	18.05 ± 11.20	$\mathbf{6.13 \pm 1.03}$	53.63 ± 2.15	8.69 ± 1.30	74.71 ± 6.00	94.18 ± 1.94
Uniform	16	100	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0

Exploration using posterior uncertainty in contextual bandits

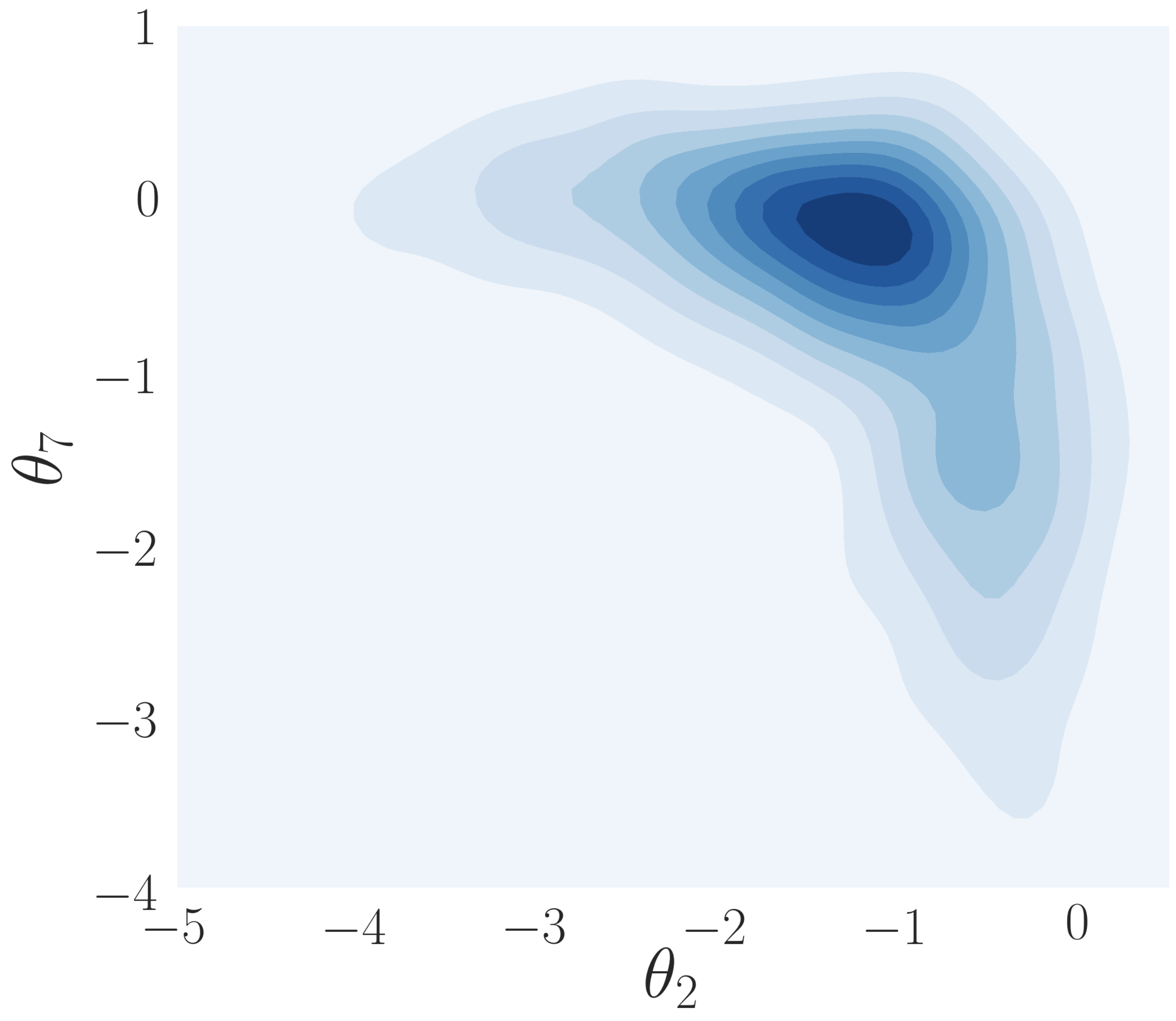
[Sun*, Zhang*, **Shi*** & Grosse, ICLR'19]

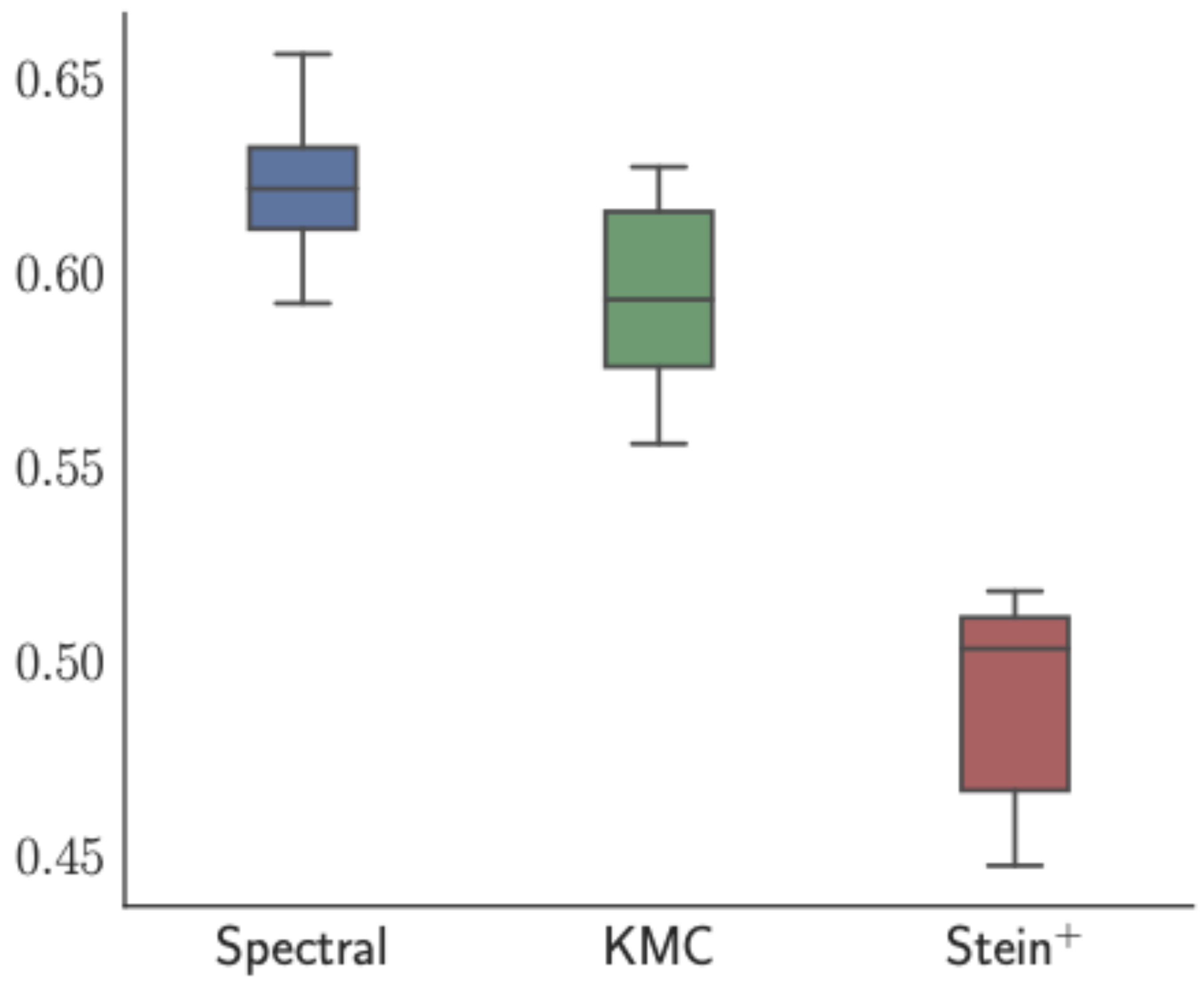
Applications

Learning Wasserstein Autoencoders

[Zhou, **Shi** & Zhu, ICML’20]

Gradient-free Hamiltonian Monte Carlo





[**Shi** et al., ICML’18]

Applications

Mutual Information Gradient Estimation

Performance of Learned Representations

Model	CIFAR-10			CIFAR-100		
	conv	fc(1024)	Y(64)	conv	fc(1024)	Y(64)
DIM (JSD)	55.81%	45.73%	40.67%	28.41%	22.16%	16.50%
DIM (JSD + PM)	52.2%	52.84%	43.17%	24.40%	18.22%	15.22%
DIM (infoNCE)	51.82%	42.81%	37.79%	24.60%	16.54%	12.96%
DIM (infoNCE + PM)	56.77%	49.42%	42.68%	25.51%	20.15%	15.35%
MIGE	**57.95%**	**57.09%**	**53.75%**	**29.86%**	**27.91%**	**25.84%**

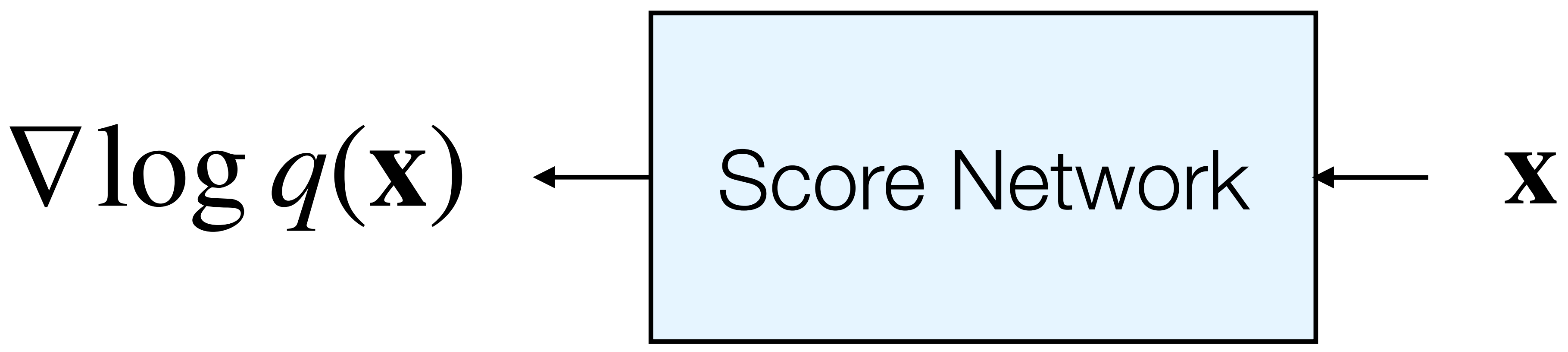
[Wen et al., ICLR’20]

Concluding Remarks

Gradient estimation for differentiable programming in probabilistic models

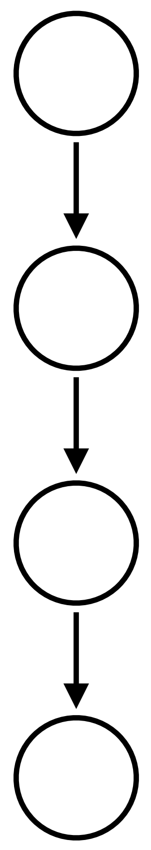
- Gradient Estimation for Discrete Expectations
  - Double control variates—a new framework for variance reduction in REINFORCE-type estimators
  - Discrete Stein operators—a general recipe for constructing flexible control variates for discrete distributions
- Gradient Estimation for Intractable Densities
  - Score estimation—a spectral approach and applications

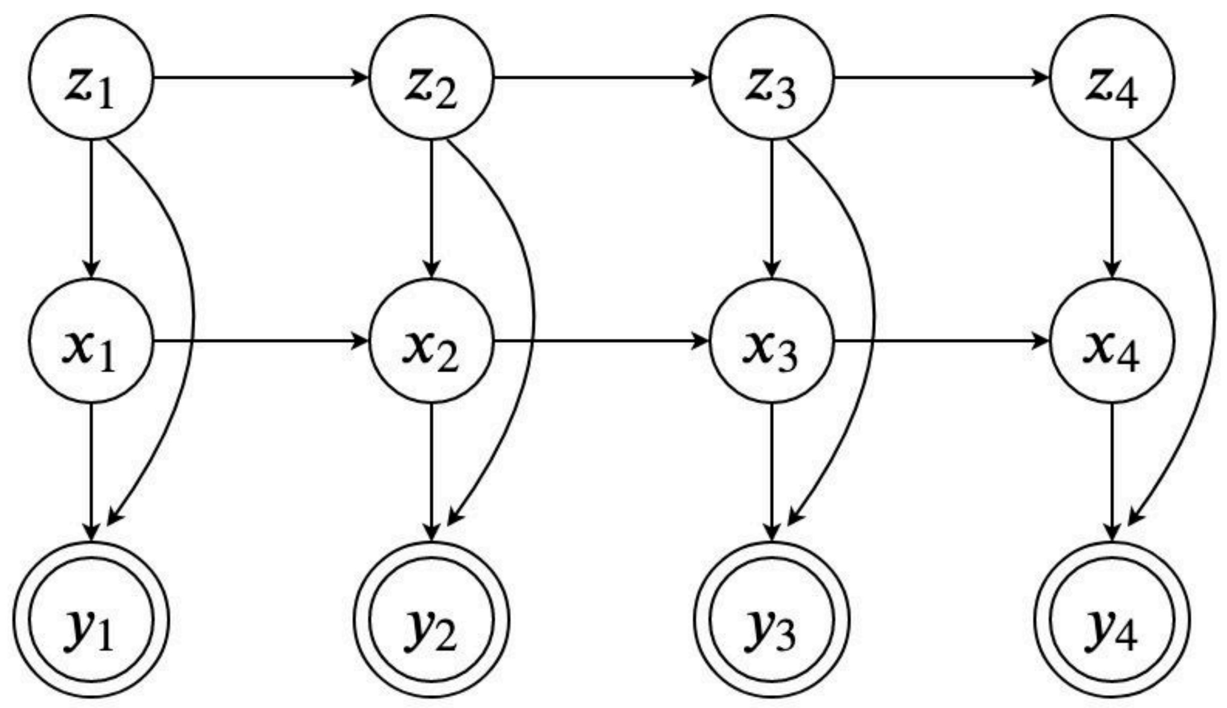
Future Directions



- Score-based probabilistic modeling
  - Parametric score estimators, e.g., sliced score matching [Song, Garg, **Shi**, Ermon, UAI'20]
  - Fit such estimators to data: score-based generative models

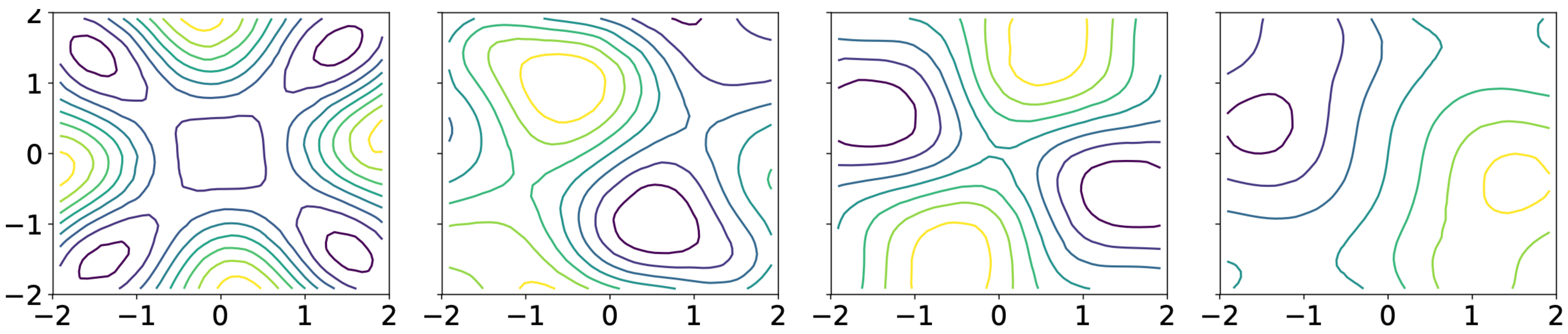
Future Directions





- Gradient estimation for discrete expectations in structured models
    - chains, temporal dependencies, state-space models
    - exploit graphical structure to achieve further variance reduction

Future Directions



[Sun, **Shi,** et al., ICML'21]

- Structured data distribution, symmetry and invariance
  - differentiable programming is good at exploiting invariance/equivariance
  - exploiting such properties in probabilistic inference?

References

- Titsias & Shi. Double Control Variates for Gradient Estimation in Discrete Latent-Variable Models. AISTATS 2022
- Shi, et al. Gradient Estimation with Discrete Stein Operators. In Submission
- Shi et al. A spectral approach to gradient estimation for implicit distributions. ICML 2018
- Sun*, Zhang*, Shi*, Grosse. Functional variational Bayesian neural networks. ICLR 2019
- Zhou, Shi, Zhu. Nonparametric score estimators. ICML 2020

References

- Grathwohl, et al. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. ICLR 2018.
- Dimitriev & Zhou. ARMS: Antithetic-REINFORCE-Multi-Sample gradient for binary variables. ICML 2021.
- Wen, et al. Mutual information gradient estimation for representation learning. ICLR 2020.
- Luo, Tian, Shi, et al. Semi-crowdsourced clustering with deep generative models. NeurIPS 2018
- Zhuo, Liu, Shi, et al. Message passing Stein variational gradient descent. ICML 2018
- Shi et al. Sparse orthogonal variational inference for Gaussian processes. AISTATS 2020