On the Reparameterisation Gradient for Non-Differentiable but Continuous Models

Dominik Wagner Luke Ong

UNIVERSITY OF OXFORD

LAFI 2023

15 January 2023

Variational Inference:

Variational Inference:

frame posterior inference as (deterministic) optimisation problem

Variational Inference:

frame posterior inference as (deterministic) optimisation problem

Posit: *variational family* of "simpler" *guide* distributions

Variational Inference:

frame posterior inference as (deterministic) optimisation problem

Posit: *variational family* of “simpler” *guide* distributions

Aim: find guide is *“closest”* to (true) posterior

Variational Inference:

frame posterior inference as (deterministic) optimisation problem

Posit: *variational family* of "simpler" *guide* distributions

Aim: find guide is *"closest"* to (true) posterior

KL divergence

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} \left[f(\boldsymbol{\theta}, \mathbf{z}) \right]}$$

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} \left[f(\boldsymbol{\theta}, \mathbf{z}) \right]}$$

use ***Stochastic Gradient Descent***

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} \left[f(\boldsymbol{\theta}, \mathbf{z}) \right]}$$

use ***Stochastic Gradient Descent***

Key ingredient: estimation of gradient of expectation

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} \left[f(\boldsymbol{\theta}, \mathbf{z}) \right]}$$

use ***Stochastic Gradient Descent***

Key ingredient: estimation of gradient of expectation

- **Score Estimator:**

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} [f(\boldsymbol{\theta}, \mathbf{z})]}$$

use ***Stochastic Gradient Descent***

Key ingredient: estimation of gradient of expectation

- **Score Estimator:**
  widely applicable but *high variance*

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} \left[f(\boldsymbol{\theta}, \mathbf{z}) \right]}$$

use ***Stochastic Gradient Descent***

Key ingredient: estimation of gradient of expectation

- **Score Estimator:**
  widely applicable but *high variance*
- **Reparametrisation Estimator:**

$$\boxed{\operatorname{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\theta}}} [f(\boldsymbol{\theta}, \mathbf{z})]}$$

expressed in PL with conditionals,
may *not* be *differentiable/continuous*

use ***Stochastic Gradient Descent***

Key ingredient: estimation of gradient of expectation

- **Score Estimator:**
  widely applicable but *high variance*
- **Reparametrisation Estimator:**
  better in practice but *may be biased!* [Lee et al., NeurIPS 2018]

Example: Temperature Regulation

Example: Temperature Regulation

```
let t0 = sample normal(20,1)
```

Example: Temperature Regulation

```
let t0 = sample normal(20,1)
    mu = t0 + if t0 >= 19 then 0
```

Example: Temperature Regulation

```
let t0 = sample normal(20,1)
    mu = t0 + if t0 >= 19 then 0
                          else 2 * (19-t0)
```

Example: Temperature Regulation

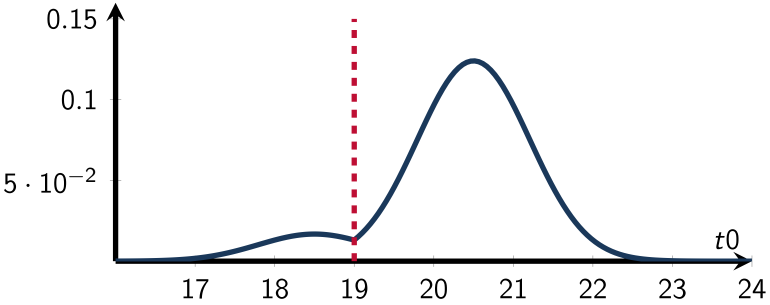
```
let t0 = sample normal(20,1)
    mu = t0 + if t0 >= 19 then 0
                          else 2 * (19-t0)
    observe 21 from normal(mu, 1)
```

Example: Temperature Regulation

```
let t0 = sample normal(20,1)
    mu = t0 + if t0 >= 19 then 0
                           else 2 * (19-t0)
    observe 21 from normal(mu, 1)
in t0
```

Example: Temperature Regulation

```
let t0 = sample normal(20,1)
    mu = t0 + if t0 >= 19 then 0
                          else 2 * (19-t0)
    observe 21 from normal(mu, 1)
in t0
```



Is the Reparametrisation Gradient Estimator biased for continuous but possibly non-differentiable programs?

Is the Reparametrisation Gradient Estimator biased for continuous but possibly non-differentiable programs?

No!

simply typed λ-calculus with $\mathbb{R}$ and primitive operations
$+$ sample $+$ observe

simply typed λ-calculus with $\mathbb{R}$ and primitive operations

$+$ sample $+$ observe

$+$ branching

simply typed λ-calculus with $\mathbb{R}$ and primitive operations

$+$ sample $+$ observe

$+$ branching $+$ *smoothed* branching

Denotational Weight Semantics

- denotational version of weight/density semantics

Denotational Weight Semantics

- denotational version of weight/density semantics
- beyond measuarability: capture piecewise definition and continuity

Denotational Weight Semantics

- denotational version of weight/density semantics
- beyond measuarability: capture piecewise definition and continuity
- complication: smoothed conditionals at higher-order [ESOP23]

Denotational Weight Semantics

- denotational version of weight/density semantics
- beyond measuarability: capture piecewise definition and continuity
- complication: smoothed conditionals at higher-order [ESOP23]

Generalise construction of Frölicher spaces

Denotational Weight Semantics

- denotational version of weight/density semantics
- beyond measuarability: capture piecewise definition and continuity
- complication: smoothed conditionals at higher-order [ESOP23]

Generalise construction of Frölicher spaces

Unbiasedness for terms without conditionals

Denotational Weight Semantics

- denotational version of weight/density semantics
- beyond measuarability: capture piecewise definition and continuity
- complication: smoothed conditionals at higher-order [ESOP23]

Generalise construction of Frölicher spaces

Unbiasedness for terms without conditionals

Example. Rephrase conditional via non-differentiable primitive:

$$c \cdot \left(\underline{\mathrm{ReLU}}\,(19 - t_0)\right)$$

Continuity for Terms with Branching

Continuity for Terms with Branching

naive check intractable!

Continuity for Terms with Branching

naive check intractable!

For analytic primitives f and g,

$$\textbf{if}\ x - y < 0\ \textbf{then}\ \underline{f}\ x\ y\ \textbf{else}\ \underline{g}\ x\ y \qquad \text{is continuous}$$

Continuity for Terms with Branching

naive check intractable!

For analytic primitives f and g,

$$\begin{aligned} & \textbf{if } x - y < 0 \textbf{ then } \underline{f}\, x\, y \textbf{ else } \underline{g}\, x\, y && \text{is continuous} \\ \iff & (f - g)_{|U} = 0 && \text{where } U := \{(x, y) \mid x = y\} \end{aligned}$$

Continuity for Terms with Branching

naive check intractable!

For analytic primitives f and g,

$$\begin{array}{lll} & \textbf{if } x - y < 0 \textbf{ then } \underline{f}\, x\, y \textbf{ else } \underline{g}\, x\, y & \text{is continuous} \\ \iff & (f - g)_{|U} = 0 & \text{where } U := \{(x, y) \mid x = y\} \\ \iff^{*} & f(x, y) = g(x, y) & (x, y) \sim U \end{array}$$

* *with probability 1*

Continuity for Terms with Branching

naive check intractable!

For analytic primitives f and g,

$$\begin{array}{lll} & \textbf{if}\ x - y < 0\ \textbf{then}\ \underline{f}\ x\ y\ \textbf{else}\ \underline{g}\ x\ y & \text{is continuous} \\ \iff & (f - g)_{|U} = 0 & \text{where } U := \{(x,y) \mid x = y\} \\ \iff^{*} & f(x,y) = g(x,y) & (x,y) \sim U \end{array}$$

* *with probability 1*

Restrict guards to affine terms:

Continuity for Terms with Branching

naive check intractable!

For analytic primitives f and g,

$$\begin{array}{lll} & \mathbf{if}\ x - y < 0\ \mathbf{then}\ \underline{f}\ x\ y\ \mathbf{else}\ \underline{g}\ x\ y & \text{is continuous} \\ \iff & (f - g)_{|U} = 0 & \text{where } U := \{(x, y) \mid x = y\} \\ \iff^{*} & f(x, y) = g(x, y) & (x, y) \sim U \end{array}$$

* *with probability 1*

Restrict guards to affine terms:

- efficiently sample from boundary

Continuity for Terms with Branching

naive check intractable!

For analytic primitives f and g,

$$
\begin{array}{lll}
& \textbf{if}\ x - y < 0\ \textbf{then}\ \underline{f}\ x\ y\ \textbf{else}\ \underline{g}\ x\ y & \text{is continuous} \\
\iff & (f - g)_{|U} = 0 & \text{where } U := \{(x, y) \mid x = y\} \\
\iff^{*} & f(x, y) = g(x, y) & (x, y) \sim U
\end{array}
$$

* *with probability 1*

Restrict guards to affine terms:

- efficiently sample from boundary
- efficiently check guard's consistency (linear arithmetic solvers)

Contributions

The Reparametrisation Gradient Estimator is unbiased for *continuous* but possibly *non-differentiable* programs

Contributions

The Reparametrisation Gradient Estimator is unbiased for *continuous* but possibly *non-differentiable* programs

- categorical models
- prove *unbiasedness* in continuous setting
- *establish continuity* in languages with conditionals compositionally
  - for affine guards: efficient *randomised* check employing *linear arithmetic solvers*

Contributions

The Reparametrisation Gradient Estimator is unbiased for *continuous* but possibly *non-differentiable* programs

- categorical models
- prove *unbiasedness* in continuous setting
- *establish continuity* in languages with conditionals compositionally
  - for affine guards: efficient *randomised* check employing *linear arithmetic solvers*

Foundation for *fast yet correct* (variational) inference for probabilistic programming