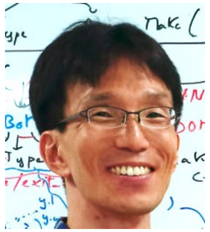
Smoothness Analysis for Probabilistic Programs with Application to Optimised Variational Inference



Wonyeol Lee[1] Xavier Rival[2] Hongseok Yang[3]

[1]Stanford, USA [2]INRIA/ENS/CNRS, France [3]KAIST, South Korea

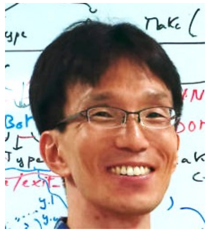
POPL 2023

Part 1

Smoothness Analysis for Probabilistic Programs with Application to Optimised Variational Inference

Part 2





Wonyeol Lee[1] Xavier Rival[2] Hongseok Yang[3]

[1]Stanford, USA [2]INRIA/ENS/CNRS, France [3]KAIST, South Korea

POPL 2023

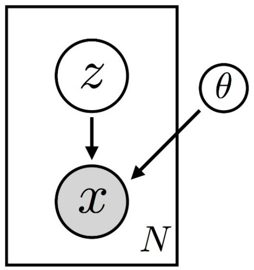
Why Care About Smoothness?

"Smoothness" = {differentiability, Lipschitz continuity, continuity, …}.

~~infinitely differentiable (in mathematics)~~

Why Care About Smoothness?

"Smoothness" = {differentiability, Lipschitz continuity, continuity, …}.

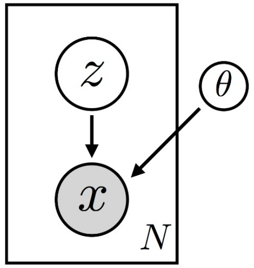


probabilistic model

Can apply many inference algorithms.

Why Care About Smoothness?

"Smoothness" = {differentiability, Lipschitz continuity, continuity, …}.



differentiable

probabilistic model

Can apply more efficient inference algorithms.

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})}\left[f(\mathbf{x})\right] = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\epsilon};\boldsymbol{\theta}))\right]$$

pathwise gradient estimator

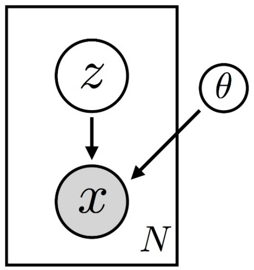
$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}$$
$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

Hamiltonian Monte Carlo

…

Why Care About Smoothness?

“Smoothness” = {differentiability, Lipschitz continuity, continuity, …}.



differentiable

probabilistic model

Can apply more efficient inference algorithms.

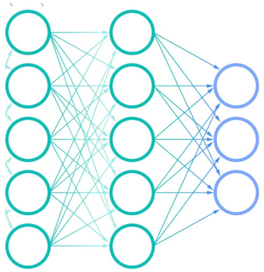
$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})}\left[f(\mathbf{x})\right]$$
$$= \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\epsilon};\boldsymbol{\theta}))\right]$$

pathwise gradient estimator

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}$$
$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

Hamiltonian Monte Carlo

…



Lipschitz continuous

neural network

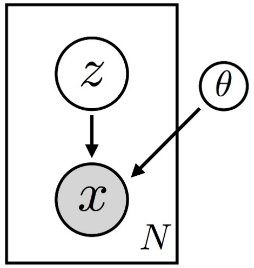
Can provide provable robustness.

Can give guaranteed generalization bounds.

…

Why Care About Smoothness?

“Smoothness” = {differentiability, Lipschitz continuity, continuity, …}.



differentiable

probabilistic model

Can apply more efficient inference algorithms.

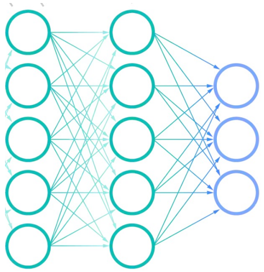
$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\mathbf{x};\boldsymbol{\theta})}\left[f(\mathbf{x})\right]$$
$$= \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\epsilon};\boldsymbol{\theta}))\right]$$

pathwise gradient estimator

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}$$
$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

Hamiltonian Monte Carlo

…



Lipschitz continuous

neural network

Can provide provable robustness.

Can give guaranteed generalization bounds.

…

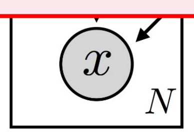
… many more examples!

…

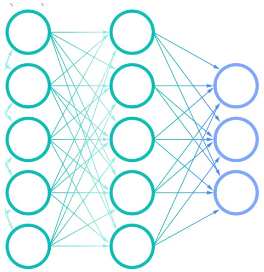
Why Care About Smoothness?

“Smoothness” = {differentiability, Lipschitz continuity, continuity, …}.

Goal: Find out **smoothness** properties, **automatically** and **soundly**.



probabilistic model



neural network

… many more examples!

$$= \mathbb{E}_{p(\epsilon)}\left[\nabla_{\theta} f(g(\epsilon;\theta))\right]$$

pathwise gradient estimator

$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

Hamiltonian Monte Carlo

…

Can provide provable robustness.

Can give guaranteed generalization bounds.

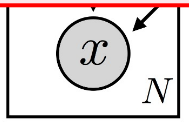
…

Why Care About Smoothness?

"Smoothness" = {differentiability, Lipschitz continuity, continuity, …}.

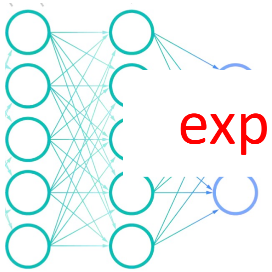
of **programs**

Goal: Find out **smoothness** properties, **automatically** and **soundly**.



probabilistic model

expressed by programs



neural network

… many more examples!

$$= \mathbb{E}_{p(\epsilon)}\left[\nabla_{\boldsymbol{\theta}} f(g(\epsilon; \boldsymbol{\theta}))\right]$$

pathwise gradient estimator

$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}$$

Hamiltonian Monte Carlo

…

Can provide provable robustness.

Can give guaranteed generalization bounds.

…

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

Smoothness Properties of Programs

Smoothness = differentiability.   Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

real-valued

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

- $[\![\mathrm{P}]\!] \; : \; \mathbb{R}^3 \; \to \; \mathbb{R}^3$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ \exp(x) \\ \operatorname{sgn}(x) \end{pmatrix}$$

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

- $[\![\mathrm{P}]\!] \quad : \quad \mathbb{R}^3 \quad \to \quad \mathbb{R}^3$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ \exp(x) \\ \mathrm{sgn}(x) \end{pmatrix}$$

- P $\vdash$ y is differentiable in x

$\iff$ …

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

- $[\![\mathrm{P}]\!] \;:\; \mathbb{R}^3 \;\to\; \mathbb{R}^3$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ \exp(x) \\ \operatorname{sgn}(x) \end{pmatrix}$$

- P $\vdash$ y is differentiable $\boxed{\text{in } x}$

$\iff$

$$\begin{pmatrix} x \\ y_0 \\ z_0 \end{pmatrix} \xmapsto{[\![\mathrm{P}]\!]} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$$

for all $y_0, z_0 \in \mathbb{R}$.

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

- $[\![\mathrm{P}]\!] \; : \; \mathbb{R}^3 \to \mathbb{R}^3$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ \exp(x) \\ \mathrm{sgn}(x) \end{pmatrix}$$

- P $\vdash$ $\boxed{y \text{ is}}$ differentiable $\boxed{\text{in } x}$

$\iff$ $f : \mathbb{R} \to \mathbb{R}$ for all $y_0, z_0 \in \mathbb{R}$.

$$\begin{pmatrix} \boxed{x} \\ y_0 \\ z_0 \end{pmatrix} \overset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x' \\ \boxed{y'} \\ z' \end{pmatrix}$$

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

- $[\![\mathrm{P}]\!] \;:\; \mathbb{R}^3 \;\to\; \mathbb{R}^3$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ \exp(x) \\ \mathrm{sgn}(x) \end{pmatrix}$$

- P $\vdash$ y is differentiable in x

$\Leftrightarrow$ $f : \mathbb{R} \to \mathbb{R}$ is differentiable for all $y_0, z_0 \in \mathbb{R}$.

$$\begin{pmatrix} \boxed{x} \\ y_0 \\ z_0 \end{pmatrix} \overset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x' \\ \boxed{y'} \\ z' \end{pmatrix}$$

Smoothness Properties of Programs

Smoothness = differentiability. Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

- $[\![\mathrm{P}]\!] \quad : \quad \mathbb{R}^3 \quad \to \quad \mathbb{R}^3$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ \exp(x) \\ \mathrm{sgn}(x) \end{pmatrix}$$

- P $\vdash$ y is differentiable in x.

  P $\nvdash$ z is differentiable in x.

  …

Smoothness Properties of Programs

Smoothness = differentiability.   Programs = deterministic, imperative programs.

```
P ≜ ( y:=exp(x) ; if (x>0) {z:=1} else {z:=-1} )
```

It is surprisingly subtle to find out such smoothness properties

(1) automatically,

(2) soundly, and

(3) precisely enough.

Differentiability Analysis (Composition Case)

Want to check differentiability of a program in a compositional way.

Challenging case: Given P, P' and $U, V \subseteq \mathrm{Var}$, want to check

$$\forall u \in U, \forall v \in V. \quad \text{P ; P'} \ \vdash \ u \text{ is differentiable in } v \,.$$

Differentiability Analysis (Composition Case)

Based on the chain rule:

$$\exists T \subseteq \mathrm{Var}. \qquad \frac{\begin{array}{lll} \forall u \in U, \forall t \in T. & \text{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in T, \forall v \in V. & \text{P} & \vdash t \text{ is differentiable in } v \end{array}}{\forall u \in U, \forall v \in V. \quad \text{P;P'} \vdash u \text{ is differentiable in } v}$$

Differentiability Analysis (Composition Case)

- Looks sound by chain rule.
- Previously considered:
  e.g., [CACM'12] for continuity.

Based on the chain rule:

$$\exists T \subseteq \mathrm{Var}. \quad \frac{\forall u \in U, \forall t \in T. \quad \mathrm{P'} \vdash u \text{ is differentiable in } t \qquad \forall t \in T, \forall v \in V. \quad \mathrm{P} \vdash t \text{ is differentiable in } v}{\forall u \in U, \forall v \in V. \quad \mathrm{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq}$$

Differentiability Analysis (Composition Case)

- Looks sound by chain rule.
- Previously considered: e.g., [CACM'12] for continuity.

Based on the chain rule:

$$\exists T \subseteq \mathrm{Var}. \quad \frac{\begin{array}{lll} \forall u \in U, \forall t \in T. & \text{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in T, \forall v \in V. & \text{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \text{P} ; \text{P'} & \vdash u \text{ is differentiable in } v \end{array}} \ \text{Seq}$$

Is this rule indeed sound?

Issue 1: Soundness

`P;P' ≜ (y:=sgn(x) ; z:=x+y).`

$$\dfrac{\begin{array}{lcl} \forall u \in U, \forall\, t \in T. & \mathrm{P'} & \vdash\ u \text{ is differentiable in } t \\ \forall\, t \in T, \forall v \in V. & \mathrm{P} & \vdash\ t \text{ is differentiable in } v \end{array}}{\begin{array}{lcl} \forall u \in U, \forall v \in V. & \mathrm{P;P'} & \vdash\ u \text{ is differentiable in } v \end{array}}\ \text{Seq}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \underset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ z \end{pmatrix}, \quad \begin{pmatrix} x \\ y \\ z \end{pmatrix} \underset{[\![\mathrm{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ y \\ x+y \end{pmatrix}$$

Issue 1: Soundness

`P;P’ ≜ (y:=sgn(x) ; z:=x+y).` $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}.$

$$\frac{\begin{array}{lll} \forall u \in U, \forall t \in T. & \mathtt{P’} & \vdash u \text{ is differentiable in } t \\ \forall t \in T, \forall v \in V. & \mathtt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \mathtt{P;P’} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \underset{[\![\mathtt{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ z \end{pmatrix}, \quad \begin{pmatrix} x \\ y \\ z \end{pmatrix} \underset{[\![\mathtt{P’}]\!]}{\longmapsto} \begin{pmatrix} x \\ y \\ x+y \end{pmatrix}$$

Issue 1: Soundness

`P;P’` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}.$

$$\dfrac{\begin{array}{llll} \checkmark & \forall u \in U, \forall t \in T. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \checkmark & \forall t \in T, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{llll} \times & \forall u \in U, \forall v \in V. & \texttt{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \underset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ z \end{pmatrix} \underset{[\![\mathrm{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ \boxed{x + \mathrm{sgn}(x)} \end{pmatrix}$$

Issue 1: Soundness

`P;P’` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}.$

$$\dfrac{\begin{array}{lll} ✓\ \forall u \in U, \forall t \in T. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ ✓\ \forall t \in T, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{✗\ \forall u \in U, \forall v \in V. \quad \texttt{P;P'} \vdash u \text{ is differentiable in } v} \ \text{Seq}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \underset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ z \end{pmatrix} \underset{[\![\mathrm{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ \boxed{x + \mathrm{sgn}(x)} \end{pmatrix}$$

This rule is unsound! But why?

Issue 1: Soundness

`P;P’ ≜ (y:=sgn(x) ; z:=x+y).` $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}.$

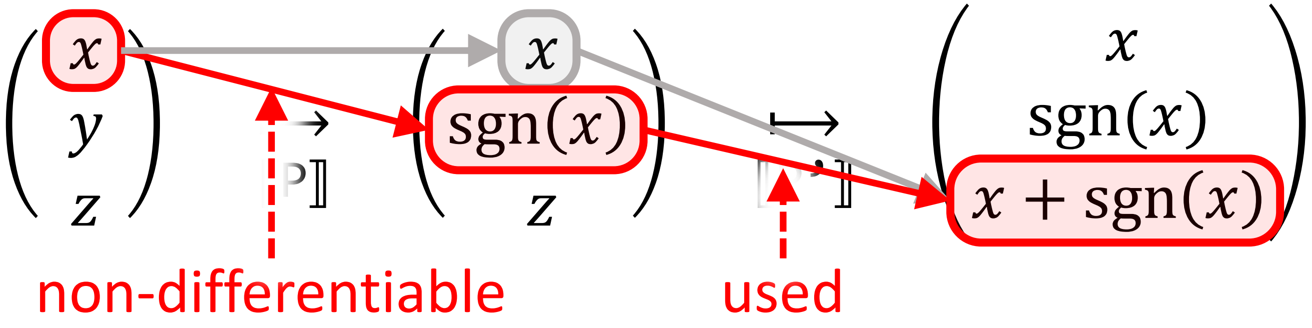
$$\dfrac{\begin{array}{lll} ✓\ \forall u \in U, \forall t \in T. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ ✓\ \forall t \in T, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{✗\ \forall u \in U, \forall v \in V. \quad \texttt{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \xrightarrow[\llbracket \texttt{P} \rrbracket]{} \begin{pmatrix} \boxed{x} \\ \operatorname{sgn}(x) \\ z \end{pmatrix} \xrightarrow[\llbracket \texttt{P'} \rrbracket]{} \begin{pmatrix} x \\ \operatorname{sgn}(x) \\ \boxed{x + \operatorname{sgn}(x)} \end{pmatrix}$$

Issue 1: Soundness

`P;P'` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}.$

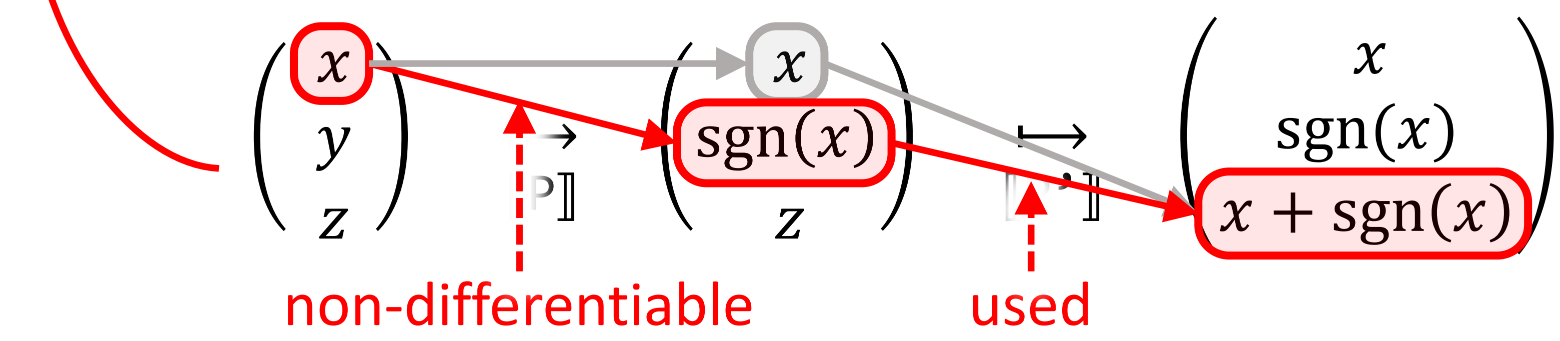
$$\dfrac{\begin{array}{lll} \checkmark\ \forall u \in U, \forall t \in T. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \checkmark\ \forall t \in T, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\times\ \forall u \in U, \forall v \in V. \quad \texttt{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq}$$



Issue 1: Soundness

P;P' $\triangleq$ (y:=sgn(x) ; z:=x+y). $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}$.

$$\dfrac{\checkmark\ \forall u \in U, \forall t \in T.\quad \text{P'} \vdash u \text{ is differentiable in } t \qquad \checkmark\ \forall t \in T, \forall v \in V.\quad \text{P} \vdash t \text{ is differentiable in } v}{\times\ \forall u \in U, \forall v \in V.\quad \text{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq}$$



Issue 1: Soundness

P;P' ≜ (y:=sgn(x) ; z:=x+y). $U \triangleq \{z\}, T \triangleq \{x\}, V \triangleq \{x\}$.

$$\dfrac{\begin{array}{lll} \forall u \in U, \forall t \in T. & \text{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in T, \forall v \in V. & \text{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \text{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq}$$

Lesson: Need to consider dependency between variables.

non-differentiable

used

Issue 1: Possible Fix

`P;P’` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $\quad U \triangleq \{z\}, \cancel{T \triangleq \{x\},}\ V \triangleq \{x\}.$

Var

$$\dfrac{\begin{array}{lll} \forall u \in U, \forall t \in \cancel{T}. & \texttt{P’} & \vdash u \text{ is differentiable in } t \\ \forall t \in \cancel{T}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\forall u \in U, \forall v \in V. \quad \texttt{P;P’} \vdash u \text{ is differentiable in } v}\ \text{Seq’}$$

✗

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \xmapsto[\llbracket \texttt{P} \rrbracket]{} \begin{pmatrix} x \\ \operatorname{sgn}(x) \\ z \end{pmatrix} \xmapsto[\llbracket \texttt{P’} \rrbracket]{} \begin{pmatrix} x \\ \operatorname{sgn}(x) \\ x + \operatorname{sgn}(x) \end{pmatrix}$$

Issue 1: Possible Fix

`P;P'` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $U \triangleq \{z\}, \cancel{T \triangleq \{x\}}, V \triangleq \{x\}.$

Var

$$\dfrac{\begin{array}{lll} \forall u \in U, \forall t \in \cancel{T}. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in \cancel{T}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \texttt{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq'}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \xmapsto{[\![\texttt{P}]\!]} \begin{pmatrix} x \\ \boxed{\operatorname{sgn}(x)} \\ z \end{pmatrix} \xmapsto{[\![\texttt{P'}]\!]} \begin{pmatrix} x \\ \operatorname{sgn}(x) \\ x + \operatorname{sgn}(x) \end{pmatrix}$$

non-differentiable

Issue 1: Possible Fix

$\texttt{P;P'} \triangleq (\texttt{y:=sgn(x) ; z:=x+y}).\quad U \triangleq \{z\}, \cancel{T \triangleq \{x\}}, V \triangleq \{x\}.$

Var

$$\dfrac{\begin{array}{llll} & \forall u \in U, \forall t \in \cancel{T}. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \times & \forall t \in \cancel{T}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{llll} \times & \forall u \in U, \forall v \in V. & \texttt{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq'}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \overset{[\![\texttt{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \boxed{\mathrm{sgn}(x)} \\ z \end{pmatrix} \overset{[\![\texttt{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ x + \mathrm{sgn}(x) \end{pmatrix}$$

This rule now looks sound. Are we done?

Issue 2: Precision

`P;P'` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $\quad U \triangleq \{x\}$, $\cancel{T \triangleq \{x\}}$, $V \triangleq \{x\}$.

Var

$$\dfrac{\begin{array}{lll} \forall u \in U, \forall t \in \cancel{T}. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in \cancel{T}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \texttt{P;P'} & \vdash u \text{ is differentiable in } v \end{array}} \;\text{Seq'}$$

✓

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \underset{[\![\texttt{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \operatorname{sgn}(x) \\ z \end{pmatrix} \underset{[\![\texttt{P'}]\!]}{\longmapsto} \begin{pmatrix} \boxed{x} \\ \operatorname{sgn}(x) \\ x + \operatorname{sgn}(x) \end{pmatrix}$$

Issue 2: Precision

`P;P’` $\triangleq$ `(y:=sgn(x) ; z:=x+y).` $\quad U \triangleq \{x\}, \cancel{T \triangleq \{x\}}, V \triangleq \{x\}.$

Var

$$\dfrac{\begin{array}{lll} \forall u \in U, \forall t \in \cancel{T}. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in \cancel{T}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \texttt{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq'}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \xmapsto[\llbracket \texttt{P} \rrbracket]{} \begin{pmatrix} x \\ \boxed{\operatorname{sgn}(x)} \\ z \end{pmatrix} \xmapsto[\llbracket \texttt{P'} \rrbracket]{} \begin{pmatrix} x \\ \operatorname{sgn}(x) \\ x + \operatorname{sgn}(x) \end{pmatrix}$$

non-differentiable

Issue 2: Precision

$\texttt{P;P'} \triangleq \texttt{(y:=sgn(x) ; z:=x+y)}.\quad U \triangleq \{x\}, \cancel{\mathcal{T} \triangleq \{x\}}, V \triangleq \{x\}.$

Var

$$\dfrac{\begin{array}{lll} \forall u \in U, \forall t \in \cancel{\mathcal{T}}. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \times\quad \forall t \in \cancel{\mathcal{T}}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\checkmark\quad \forall u \in U, \forall v \in V. \quad \texttt{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq'}$$

cannot deduce →

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \overset{[\![\texttt{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ \boxed{\mathrm{sgn}(x)} \\ z \end{pmatrix} \overset{[\![\texttt{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ \mathrm{sgn}(x) \\ x + \mathrm{sgn}(x) \end{pmatrix}$$

non-differentiable

This rule is too imprecise!

Issue 2: Precision

$$\frac{\begin{array}{lll} \forall u \in U,\ \forall t \in \mathrm{Var}. & \mathrm{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in \mathrm{Var}, \forall v \in V. & \mathrm{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \mathrm{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq'}$$

Admit the imprecision for now.

Is this rule indeed sound?

Issue 3: Soundness (Again)

`P;P’ ≜ (y:=x ; z:=f(x,y))` for $f(x,y) \triangleq \begin{cases} xy/(x^2+y^2) & \text{if } (x,y) \neq (0,0) \\ 0 & \text{if } (x,y) = (0,0). \end{cases}$

$$\frac{\begin{array}{llll} \forall u \in U, \quad \forall t \in \mathrm{Var}. & \texttt{P'} & \vdash & u \text{ is differentiable in } t \\ \forall t \in \mathrm{Var}, \forall v \in V. & \texttt{P} & \vdash & t \text{ is differentiable in } v \end{array}}{\forall u \in U, \forall v \in V. \quad \texttt{P;P'} \vdash u \text{ is differentiable in } v} \text{Seq'}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \xmapsto[⟦\texttt{P}⟧]{} \begin{pmatrix} x \\ x \\ z \end{pmatrix} \qquad \begin{pmatrix} x \\ y \\ z \end{pmatrix} \xmapsto[⟦\texttt{P'}⟧]{} \begin{pmatrix} x \\ y \\ f(x,y) \end{pmatrix}$$

Issue 3: Soundness (Again)

$$\texttt{P;P'} \triangleq \texttt{(y:=x ; z:=f(x,y))} \quad \text{for} \quad f(x,y) \triangleq \begin{cases} xy/(x^2+y^2) & \text{if } (x,y) \neq (0,0) \\ 0 & \text{if } (x,y) = (0,0). \end{cases}$$

$U \triangleq \{z\}, V \triangleq \{x\}.$

$$\dfrac{\begin{array}{lll} \forall u \in U, \quad \forall t \in \mathrm{Var}. & \texttt{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in \mathrm{Var}, \forall v \in V. & \texttt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\forall u \in U, \forall v \in V. \quad \texttt{P;P'} \vdash u \text{ is differentiable in } v} \ \text{Seq'}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \underset{[\![\texttt{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ x \\ z \end{pmatrix} \qquad \begin{pmatrix} x \\ y \\ z \end{pmatrix} \underset{[\![\texttt{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ y \\ \boxed{f(x,y)} \end{pmatrix}$$

Issue 3: Soundness (Again)

P;P' $\triangleq$ (y:=x ; z:=f(x,y)) for $f(x,y) \triangleq \begin{cases} xy/(x^2+y^2) & \text{if } (x,y) \neq (0,0) \\ 0 & \text{if } (x,y) = (0,0). \end{cases}$

$U \triangleq \{z\}, V \triangleq \{x\}$.

$$\dfrac{\begin{array}{lll} ✓\ \forall u \in U,\ \forall t \in \mathrm{Var}. & \text{P'} & \vdash u \text{ is differentiable in } t \\ ✓\ \forall t \in \mathrm{Var}, \forall v \in V. & \text{P} & \vdash t \text{ is differentiable in } v \end{array}}{✗\ \forall u \in U, \forall v \in V. \quad \text{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq'}$$

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \xmapsto[\llbracket \text{P} \rrbracket]{} \begin{pmatrix} x \\ x \\ z \end{pmatrix} \xmapsto[\llbracket \text{P'} \rrbracket]{} \begin{pmatrix} x \\ x \\ \boxed{1[x \neq 0] \cdot \frac{1}{2}} \end{pmatrix}$$

Issue 3: Soundness (Again)

`P;P’` $\triangleq$ `(y:=x ; z:=f(x,y))` for $f(x,y) \triangleq \begin{cases} xy/(x^2+y^2) & \text{if } (x,y) \neq (0,0) \\ 0 & \text{if } (x,y) = (0,0). \end{cases}$

$U \triangleq \{z\}, V \triangleq \{x\}.$

✓ $\forall u \in U,\ \forall t \in \mathrm{Var}.$ `P’` $\vdash u$ is differentiable in t

✓ $\forall t \in \mathrm{Var}, \forall v \in V.$ `P` $\vdash t$ is differentiable in v

$$\frac{\forall u \in U,\ \forall t \in \mathrm{Var}.\ \texttt{P'} \vdash u \text{ is differentiable in } t \qquad \forall t \in \mathrm{Var}, \forall v \in V.\ \texttt{P} \vdash t \text{ is differentiable in } v}{\forall u \in U, \forall v \in V.\ \texttt{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq'}$$

✗ $\forall u \in U, \forall v \in V.$ `P;P’` $\vdash u$ is differentiable in v

$$\begin{pmatrix} \boxed{x} \\ y \\ z \end{pmatrix} \underset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x \\ x \\ z \end{pmatrix} \underset{[\![\mathrm{P'}]\!]}{\longmapsto} \begin{pmatrix} x \\ x \\ \boxed{1[x \neq 0] \cdot \frac{1}{2}} \end{pmatrix}$$

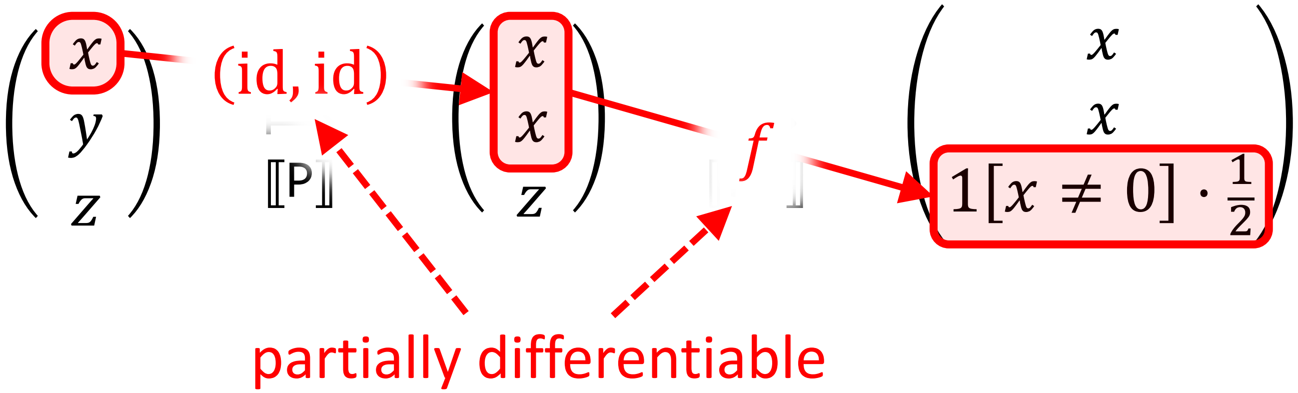
This rule is still unsound! But why?

Issue 3: Soundness (Again)

P;P' $\triangleq$ `(y:=x ; z:=f(x,y))` for $f(x,y) \triangleq \begin{cases} xy/(x^2+y^2) & \text{if } (x,y) \neq (0,0) \\ 0 & \text{if } (x,y) = (0,0). \end{cases}$

$U \triangleq \{z\}, V \triangleq \{x\}.$

$$\dfrac{\begin{array}{ll} \checkmark & \forall u \in U,\ \forall t \in \mathrm{Var}.\quad \text{P'} \vdash u \text{ is differentiable in } t \\ \checkmark & \forall t \in \mathrm{Var}, \forall v \in V.\quad \text{P} \vdash t \text{ is differentiable in } v \end{array}}{\times \quad \forall u \in U, \forall v \in V.\quad \text{P;P'} \vdash u \text{ is differentiable in } v}\ \text{Seq'}$$



Issue 3: Soundness (Again)

P;P' $\equiv$ … $\quad$ … if $(x,y) \neq (0,0)$

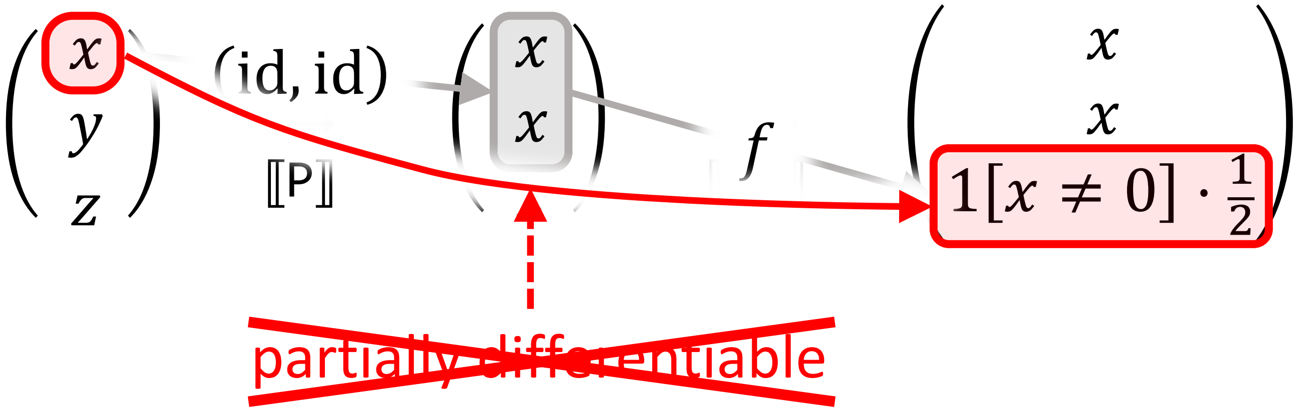
$U \triangleq \{z$ … $\quad$ … $= (0,0)$.

Fact: g, h are partially differentiable $\not\Rightarrow$ $g \circ h$ does so.

✓ $\forall u \in U, \quad \forall t \in \mathrm{Var}. \quad$ P' $\vdash$ u is differentiable in t

✓ $\forall t \in \mathrm{Var}, \forall v \in V. \quad$ P $\vdash$ t is differentiable in v

✗ $\forall u \in U, \forall v \in V. \quad$ P;P' $\vdash$ u is differentiable in v $\quad$ Seq'



Issue 3: Soundness (Again)

`P;P'` $\triangleq$ `(y:` ... if $(x,y) \neq (0,0)$

... $= (0,0)$.

$U \triangleq \{z\}, V \triangleq$ ...

g, h are partially differentiable $\Longrightarrow$ $g \circ h$ does so.

assumed implicitly, but invalid.

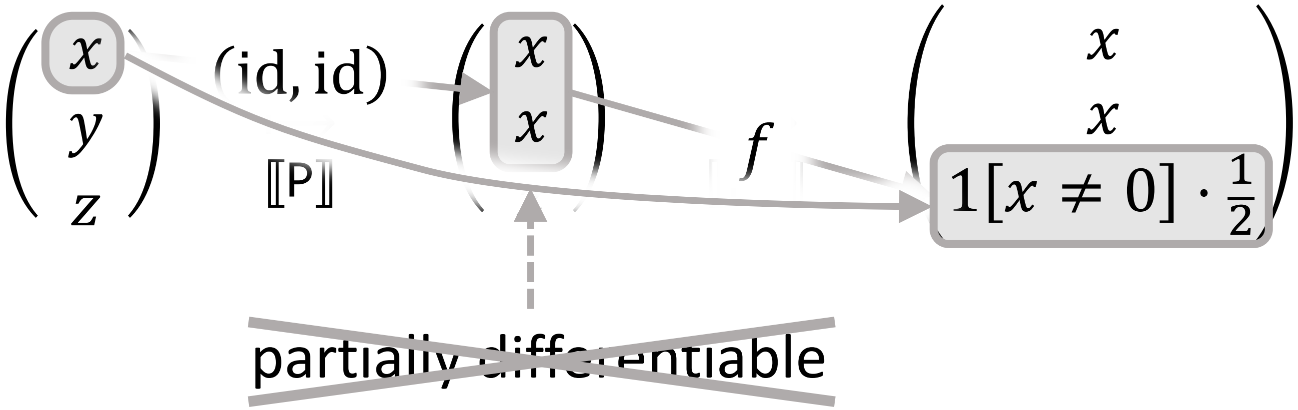
✓ $\forall u \in U,\ \forall t \in \mathrm{Var}.$ P' $\vdash$ u is differentiable in t

✓ $\forall t \in \mathrm{Var}, \forall v \in V.$ P $\vdash$ t is differentiable in v

---

Seq'

✗ $\forall u \in U, \forall v \in V.$ P;P' $\vdash$ u is differentiable in v



Issue 3: Soundness (Again)

g, h are partially differentiable $\implies g \circ h$ does so.

assumed implicitly, but invalid.

$$\dfrac{\begin{array}{lll} \forall u \in U,\ \forall t \in \mathrm{Var}. & \mathtt{P'} & \vdash u \text{ is differentiable in } t \\ \forall t \in \mathrm{Var}, \forall v \in V. & \mathtt{P} & \vdash t \text{ is differentiable in } v \end{array}}{\begin{array}{lll} \forall u \in U, \forall v \in V. & \mathtt{P;P'} & \vdash u \text{ is differentiable in } v \end{array}}\ \text{Seq'}$$

Lesson: Need to identify & check assumptions on target smoothness.

~~partially differentiable~~

It is subtle to do smoothness analysis, soundly and precisely.

→ Our approach for smoothness analysis

Our Approach: Smoothness Property

$\mathrm{P} \ \vdash \ U$ is ϕ-smooth in $V \qquad (U, V \subseteq \mathrm{Var})$

Our Approach: Smoothness Property

$$\text{P} \ \vdash \ U \text{ is } \phi\text{-smooth in } V \qquad (U, V \subseteq \mathrm{Var})$$

- $\phi \subseteq \{f: \mathbb{R}^n \to \mathbb{R}^m\}$: any set of "smooth" functions.
  - E.g., $\{f$: partially differentiable$\}$.

Our Approach: Smoothness Property

$$\text{P} \;\vdash\; U \text{ is } \phi\text{-smooth in } V \qquad (U, V \subseteq \mathrm{Var})$$

- $\phi \subseteq \{f : \mathbb{R}^n \to \mathbb{R}^m\}$: any set of "smooth" functions.
  - E.g., $\{f$: partially differentiable$\}$, $\{f$: jointly differentiable$\}$, $\{f$: continuous$\}$, $\cdots$.

Our Approach: Smoothness Property

$$\mathrm{P} \ \vdash \ U \text{ is } \phi\text{-smooth in } V \qquad (U, V \subseteq \mathrm{Var})$$

- $\phi \subseteq \{f : \mathbb{R}^n \to \mathbb{R}^m\}$: any set of "smooth" functions.
  - E.g., $\{f$: partially differentiable$\}$, $\{f$: jointly differentiable$\}$, $\{f$: continuous$\}$, $\cdots$.

- P $\vdash$ $\{x, y, z\}$ is ϕ-smooth in $\{y, z\}$

  $\iff$ $\cdots$

Our Approach: Smoothness Property

$$\mathrm{P} \;\vdash\; U \text{ is } \phi\text{-smooth in } V \qquad (U, V \subseteq \mathrm{Var})$$

- $\phi \subseteq \{f : \mathbb{R}^n \to \mathbb{R}^m\}$: any set of "smooth" functions.
  - E.g., $\{f$: partially differentiable$\}$, $\{f$: jointly differentiable$\}$, $\{f$: continuous$\}$, $\cdots$.

- $\mathrm{P} \;\vdash\; \{x, y, z\}$ is ϕ-smooth $\boxed{\text{in } \{y, z\}}$

$\iff$ for any $x_0 \in \mathbb{R}$.

$$\begin{pmatrix} x_0 \\ y \\ z \end{pmatrix} \overset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$$

Our Approach: Smoothness Property

$$\mathrm{P} \;\vdash\; U \text{ is } \phi\text{-smooth in } V \qquad (U, V \subseteq \mathrm{Var})$$

- $\phi \subseteq \{f : \mathbb{R}^n \to \mathbb{R}^m\}$: any set of "smooth" functions.
  - E.g., $\{f$: partially differentiable$\}$, $\{f$: jointly differentiable$\}$, $\{f$: continuous$\}$, $\cdots$.

- P $\vdash$ $\boxed{\{x, y, z\} \text{ is}}$ ϕ-smooth $\boxed{\text{in } \{y, z\}}$

$\iff$ $f : \mathbb{R}^2 \to \mathbb{R}^3$ for any $x_0 \in \mathbb{R}$.

$$\begin{pmatrix} x_0 \\ \boxed{\begin{matrix} y \\ z \end{matrix}} \end{pmatrix} \xmapsto{[\![\mathrm{P}]\!]} \begin{pmatrix} \boxed{\begin{matrix} x' \\ y' \\ z' \end{matrix}} \end{pmatrix}$$

Our Approach: Smoothness Property

$$\mathrm{P} \;\vdash\; U \text{ is } \phi\text{-smooth in } V \qquad (U, V \subseteq \mathrm{Var})$$

- $\phi \subseteq \{f : \mathbb{R}^n \to \mathbb{R}^m\}$: any set of "smooth" functions.
  - E.g., $\{f$: partially differentiable$\}$, $\{f$: jointly differentiable$\}$, $\{f$: continuous$\}$, $\cdots$.

- P $\vdash$ $\boxed{\{x, y, z\} \text{ is}}$ ϕ-smooth $\boxed{\text{in } \{y, z\}}$

$$\Longleftrightarrow \quad f : \mathbb{R}^2 \to \mathbb{R}^3 \in \phi \quad \text{for any } x_0 \in \mathbb{R}.$$

$$\begin{pmatrix} x_0 \\ \boxed{\begin{matrix} y \\ z \end{matrix}} \end{pmatrix} \overset{[\![\mathrm{P}]\!]}{\longmapsto} \begin{pmatrix} \boxed{\begin{matrix} x' \\ y' \\ z' \end{matrix}} \end{pmatrix}$$

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \rightarrow 2^{\mathrm{Var}}$$

dependency
info of P

smoothness
info of P

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

smoothness info of P

- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.

  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

smoothness info of P

- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.

  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

- Rules: $d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)$.

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

smoothness info of P

- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.

  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

- Rules: $d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)$.

  $s_{\mathrm{P;P'}}(V) \triangleq s_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big) \cap d_{\mathrm{P}}(s_{\mathrm{P'}}(V)^c)^c$.

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

smoothness info of P

- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.

  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

- Rules: $d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)$.

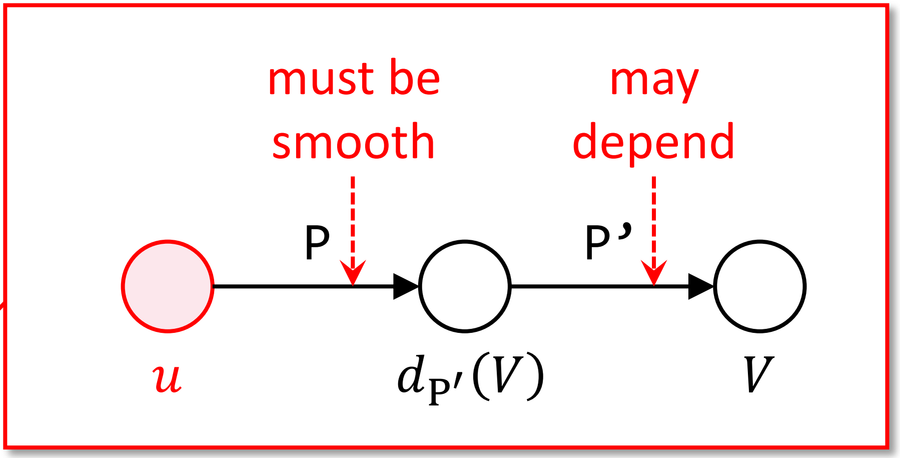
  $\underbrace{s_{\mathrm{P;P'}}(V)}_{u\,\in} \triangleq \underbrace{s_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)}_{u\,\in} \underset{\wedge}{\cap} \underbrace{d_{\mathrm{P}}(s_{\mathrm{P'}}(V)^c)^c}_{u\,\in}$.

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

smoothness info of P



- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.

  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

- Rules: $d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)$.

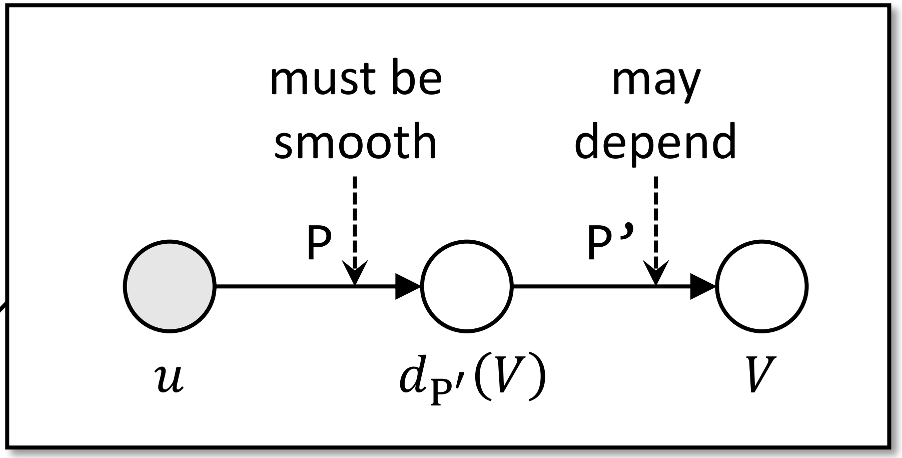
$$\underbrace{s_{\mathrm{P;P'}}(V)}_{u\,\in} \triangleq \underbrace{s_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)}_{u\,\in} \underset{\wedge}{\cap} \underbrace{d_{\mathrm{P}}(s_{\mathrm{P'}}(V)^c)^c}_{u\,\in}.$$

Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

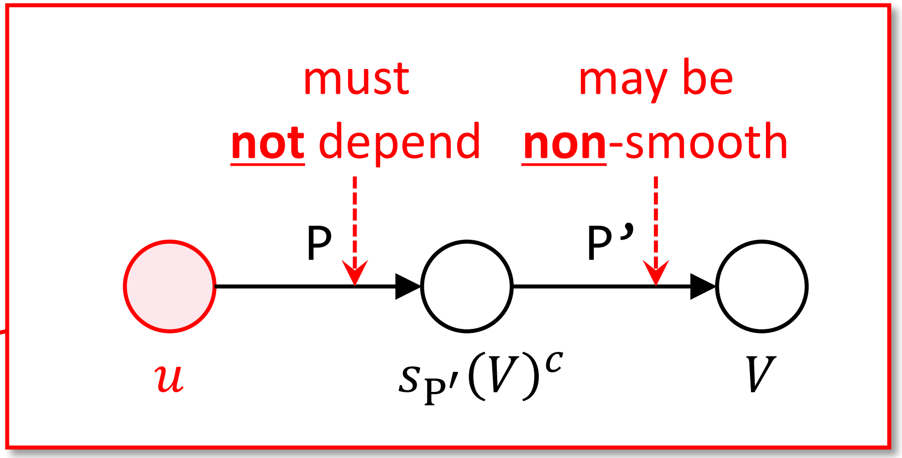
smoothness info of P



- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.
  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

- Rules: $d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)$.

$$\underbrace{s_{\mathrm{P;P'}}(V)}_{u\,\in} \triangleq \underbrace{s_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)}_{u\,\in} \overset{}{\underset{\land}{\cap}} \underbrace{d_{\mathrm{P}}(s_{\mathrm{P'}}(V)^c)^c}_{u\,\in}.$$



Our Approach: Smoothness Analysis

$$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$$

dependency info of P

smoothness info of P

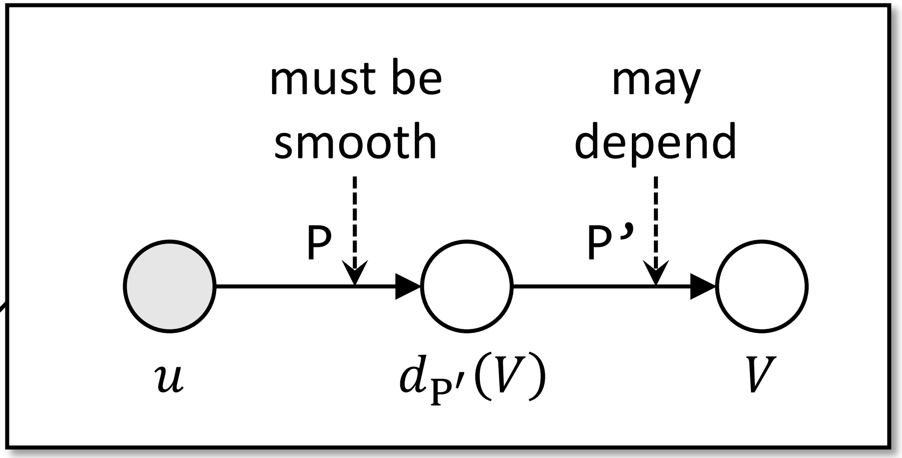
- Invariants: P $\vdash$ V is dependent at most on $d_{\mathrm{P}}(V)$.

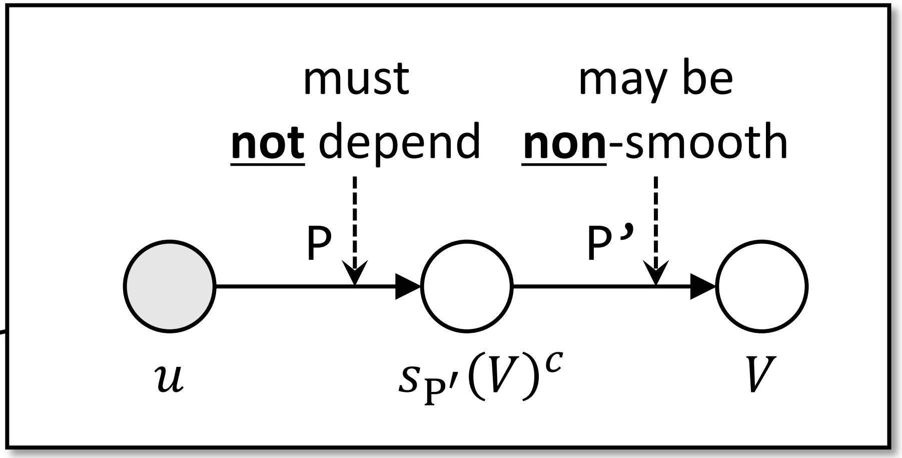
  P $\vdash$ V is ϕ-smooth in $s_{\mathrm{P}}(V)$.

- Rules: $d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)$.

  $\underbrace{s_{\mathrm{P;P'}}(V)}_{u\,\in} \triangleq \underbrace{s_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big)}_{u\,\in} \cap \underbrace{d_{\mathrm{P}}\big(s_{\mathrm{P'}}(V)^c\big)^c}_{u\,\in}$. $\quad \wedge$

$\Longrightarrow$ can apply the chain rule





Our Approach: Smoothness Analysis

$\mathrm{Var} \to 2^{\mathrm{Var}}$

$d_{\mathrm{P}},\ s_{\mathrm{P}}\ :\ 2^{\mathrm{Var}} \to 2^{\mathrm{Var}}$

dependency info of P

smoothness info of P

- Invariants:

P : first-order, imperative language
(+ probabilistic prog. constructs)

- Rules:

$d_{\mathrm{P;P'}}(V) \triangleq d_{\mathrm{P}}\big(d_{\mathrm{P'}}(V)\big).$

$s_{\mathrm{P;P'}}$

$u \in$

More details are in the paper.

Our Approach: Soundness

<u>Theorem</u> Our ϕ-smoothness analysis is sound if ϕ satisfies five assumptions:

- …
- …
- …
- …
- …

Our Approach: Soundness

Theorem Our ϕ-smoothness analysis is sound if ϕ satisfies five assumptions:

- Composition: $\forall f\colon \mathbb{R}^n \to \mathbb{R}^m, g\colon \mathbb{R}^m \to \mathbb{R}^l.\quad f, g \in \phi \implies (g \circ f) \in \phi.$
- Pairing: $\forall f\colon \mathbb{R}^n \to \mathbb{R}^m, g\colon \mathbb{R}^n \to \mathbb{R}^l.\quad f, g \in \phi \implies (f, g) \in \phi.$
- …
- …
- …

Our Approach: Soundness

<u>Theorem</u> Our ϕ-smoothness analysis is sound if ϕ satisfies five assumptions:

- Composition: $\forall f: \mathbb{R}^n \to \mathbb{R}^m, g: \mathbb{R}^m \to \mathbb{R}^l. \quad f, g \in \phi \implies (g \circ f) \in \phi.$
- Pairing: $\forall f: \mathbb{R}^n \to \mathbb{R}^m, g: \mathbb{R}^n \to \mathbb{R}^l. \quad f, g \in \phi \implies (f, g) \in \phi.$
- …
- …
- …

Target smoothness property	A3 (proj.)	A4 (pair.)	A5 (rest.)	A6 (comp.)	A7 (stri.)
cont. (C^0)	○	○	○	○	○
locally Lipschitz (= $\phi^{(l)}$)	○	○	○	○	○
uniformly cont.	○	○	○	○	○
Lipschitz cont.	○	○	○	○	○
jointly diff.	○	○	○	○	○
continuously diff. (C^1)	○	○	○	○	○
smooth (C^∞)	○	○	○	○	○
real analytic (C^ω)	○	○	○	○	○
partially cont. (= $\phi^{(pc)}$)	○	○	○	×	○
partially diff. (= $\phi^{(pd)}$)	○	○	○	×	○
almost-everywhere cont.	○	○	×	×	○
almost-everywhere diff.	○	○	×	×	○
coordinatewise non-decreasing	○	○	○	○	○
locally bounded	○	○	○	○	○
bounded	×	○	○	○	○
Borel measurable	○	○	○	○	○

Our Approach: Soundness

<u>Theorem</u> Our ϕ-smoothness analysis is sound if ϕ satisfies five assumptions:

- Composition: $\forall f: \mathbb{R}^n \to \mathbb{R}^m, g: \mathbb{R}^m \to \mathbb{R}^l. \quad f, g \in \phi \implies (g \circ f) \in \phi.$
- Pairing: $\forall f: \mathbb{R}^n \to \mathbb{R}^m, g: \mathbb{R}^n \to \mathbb{R}^l. \quad f, g \in \phi \implies (f, g) \in \phi.$
- …
- …
- …

Target smoothness property	A3 (proj.)	A4 (pair.)	A5 (rest.)	A6 (comp.)	A7 (stri.)
cont. (C^0)	○	○	○	○	○
locally Lipschitz ($= \phi^{(l)}$)	○	○	○	○	○
uniformly cont.	○	○	○	○	○
Lipschitz cont.	○	○	○	○	○
jointly diff.	○	○	○	○	○
continuously diff. (C^1)	○	○	○	○	○
smooth (C^∞)	○	○	○	○	○
real analytic (C^ω)	○	○	○	○	○
partially cont. ($= \phi^{(pc)}$)	○	○	○	×	○
partially diff. ($= \phi^{(pd)}$)	○	○	○	×	○
almost-everywhere cont.	○	○	×	×	○
almost-everywhere diff.	○	○	×	×	○
coordinatewise non-decreasing	○	○	○	○	○
locally bounded	○	○	○	○	○
bounded	×	○	○	○	○
Borel measurable	○	○	○	○	○

Our Approach: Soundness

Theorem Our ϕ-smoothness analysis is sound if ϕ satisfies five assumptions:

- Composition: $\forall f: \mathbb{R}^n \to \mathbb{R}^m, g: \mathbb{R}^m \to \mathbb{R}^l.\quad f, g \in \phi \implies (g \circ f) \in \phi.$
- Pairing: $\forall f: \mathbb{R}^n \to \mathbb{R}^m, g: \mathbb{R}^n \to \mathbb{R}^l.\quad f, g \in \phi \implies (f, g) \in \phi.$
- …
- …
- …

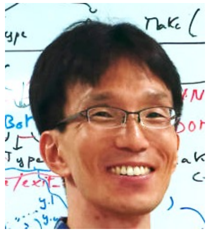
Target smoothness property	A3 (proj.)	A4 (pair.)	A5 (rest.)	A6 (comp.)	A7 (stri.)
cont. (C^0)	○	○	○	○	○
locally Lipschitz ($= \phi^{(l)}$)	○	○	○	○	○
uniformly cont.	○	○	○	○	○
Lipschitz cont.	○	○	○	○	○
jointly diff.	○	○	○	○	○
continuously diff. (C^1)	○	○	○	○	○
smooth (C^∞)	○	○	○	○	○
real analytic (C^ω)	○	○	○	○	○
partially cont. ($= \phi^{(pc)}$)	○	○	○	×	○
partially diff. ($= \phi^{(pd)}$)	○	○	○	×	○
almost-everywhere cont.	○	○	×	×	○
almost-everywhere diff.	○	○	×	×	○
coordinatewise non-decreasing	○	○	○	○	○
locally bounded	○	○	○	○	○
bounded	×	○	○	○	○
Borel measurable	○	○	○	○	○

Part 1

Smoothness Analysis for Probabilistic Programs with Application to Optimised Variational Inference

Part 2





Wonyeol Lee[1] Xavier Rival[2] Hongseok Yang[3]

[1]Stanford, USA [2]INRIA/ENS/CNRS, France [3]KAIST, South Korea

Variational Inference

- Problem: Given probability density functions $p(z, x)$ and $q_\theta(z)$,

$$\text{minimize} \quad \mathcal{L}(\theta) \triangleq \mathbb{E}_{q_\theta(z)}\left[\log \frac{q_\theta(z)}{p(z,x)}\right] \quad \text{over } \theta \in \mathbb{R}^n.$$

- Typical approach: Apply a gradient descent algorithm.

$$\theta_{t+1} \coloneqq \theta_t - \eta \cdot \nabla_\theta \mathcal{L}(\theta_t).$$

Variational Inference

- Problem: Given probability density functions $p(z, x)$ and $q_\theta(z)$,

$$\text{minimize} \quad \mathcal{L}(\theta) \triangleq \mathbb{E}_{q_\theta(z)}\left[\log \frac{q_\theta(z)}{p(z,x)}\right] \quad \text{over } \theta \in \mathbb{R}^n.$$

- Typical approach: Apply a gradient descent algorithm.

$$\theta_{t+1} \coloneqq \theta_t - \eta \cdot \nabla_\theta \mathcal{L}(\theta_t).$$

difficult to compute, so only estimate.

Gradient Estimators

- Basic estimator (called score estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \cdots \qquad \text{for } z \sim q_\theta(-).$$

- "Optimized" estimator (called pathwise gradient estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \cdots \qquad \text{for } z \sim r(-).$$

Gradient Estimators

- Basic estimator (called score estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \log \frac{q_\theta(z)}{p(z,x)} \times \nabla_\theta (\log q_\theta(z)) \qquad \text{for } z \sim q_\theta(-).$$

- "Optimized" estimator (called pathwise gradient estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \nabla_\theta \left(\log \frac{q_\theta\big(t_\theta(z)\big)}{p(t_\theta(z),x)} \right) \qquad \text{for } z \sim r(-).$$

Gradient Estimators

- Basic estimator (called score estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \log \frac{q_\theta(z)}{p(z,x)} \times \nabla_\theta (\log q_\theta(z)) \qquad \text{for } z \sim q_\theta(-).$$

Requirements: $q_\theta(z)$ should be differentiable in $\theta, \cdots$.

- "Optimized" estimator (called pathwise gradient estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \nabla_\theta \left(\log \frac{q_\theta(t_\theta(z))}{p(t_\theta(z),x)} \right) \qquad \text{for } z \sim r(-).$$

Requirements: $q_\theta(z)$ and $p(z,x)$ should be differentiable in θ and $z, \cdots$.

Gradient Estimators

- Basic estimator (called score estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \log \frac{q_\theta(z)}{p(z,x)} \times \nabla_\theta(\log q_\theta(z)) \qquad \text{for } z \sim q_\theta(-).$$

Requirements: $q_\theta(z)$ should be differentiable in $\theta, \cdots$.

- "Optimized" estimator (called pathwise gradient estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \nabla_\theta \left(\log \frac{q_\theta(t_\theta(z))}{p(t_\theta(z),x)}\right) \qquad \text{for } z \sim r(-).$$

Requirements: $q_\theta(z)$ and $p(z,x)$ should be differentiable in θ and $z, \cdots$.

In practice, we apply optimized estimator selectively to some of $z_1, \cdots, z_m$.
To do so soundly, we need to know differentiable parts of $q_\theta(z)$ and $p(z,x)$.

Gradient Estimators

- Basic estimator (called score estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \log \frac{q_\theta(z)}{p(z,x)} \times \nabla_\theta(\log q_\theta(z)) \qquad \text{for } z \sim q_\theta(-).$$

Requirements: $q_\theta(z)$ should be differentiable in θ, $\cdots$.

- "Optimized" estimator (called pathwise gradient estimator):

$$\nabla_\theta \mathcal{L}(\theta) \approx \nabla_\theta \left(\log \frac{q_\theta(t_\theta(z))}{p(t_\theta(z),x)}\right) \qquad \text{for } z \sim r(-).$$

Requirements: $q_\theta(z)$ and $p(z,x)$ should be dif

expressed by programs

Part 1 is used here.

In practice, we apply optimized estimator selectively to some of $z_1, \cdots, z_m$.
To do so soundly, we need to know differentiable parts of $q_\theta(z)$ and $p(z,x)$.

Evaluation Results

PL for variational inference
(with neural nets, ⋯)

of "optimized" random var's

Example in Pyro	LoC	Our optimizer	Pyro's default optimizer	
		# rv (Sound)	# rv (Sound)	# rv (Unsound)
Splitting normal	16			
⋯ (7 examples omitted)	⋯			
Deep exponential family	105			
Deep Markov model	112			
Hidden Markov model	137			
Single-cell annotation	147			
Attend-infer-repeat	174			
Conditional VAE	205			

Evaluation Results

PL for variational inference
(with neural nets, ⋯)

of "optimized" random var's

Example in Pyro	LoC	Our optimizer	Pyro's default optimizer	
		# rv (Sound)	# rv (Sound)	# rv (Unsound)
Splitting normal	16			
⋯ (7 examples omitted)	⋯			
Deep exponential family	105			
Deep Markov model	112			
Hidden Markov model	137			
Single-cell annotation	147			
Attend-infer-repeat	174			
Conditional VAE	205			

- Ours checks req's using smoothness analysis.
- Pyro does not check req's, so it can be unsound.

Evaluation Results

PL for variational inference (with neural nets, ⋯)

\# of "optimized" random var's

Example in Pyro	LoC	Our optimizer	Pyro's default optimizer	
		# rv (Sound)	# rv (Sound)	# rv (Unsound)
Splitting normal	16	1	1	1
⋯ (7 examples omitted)	⋯	⋯	⋯	⋯
Deep exponential family	105	6	6	0
Deep Markov model	112	1	1	0
Hidden Markov model	137	2	2	0
Single-cell annotation	147	3	3	0
Attend-infer-repeat	174	1	1	1
Conditional VAE	205	1	1	0

Evaluation Results

PL for variational inference (with neural nets, ⋯)

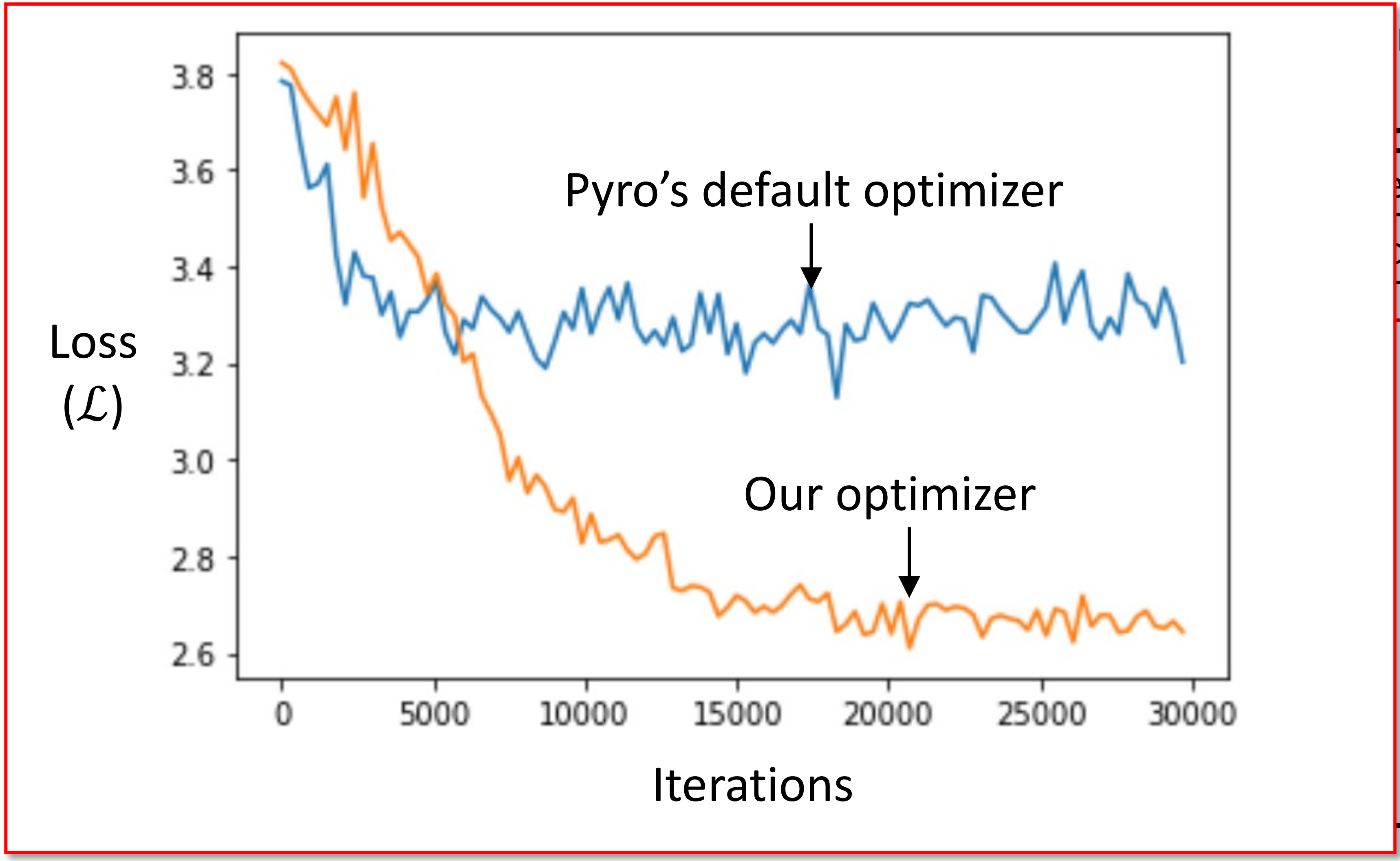
of "optimized" random var's

Example in Pyro	LoC	Our optimizer: # rv (Sound)	Pyro's default optimizer: # rv (Sound)	Pyro's default optimizer: # rv (Unsound)
Splitting normal	16	1	1	1
⋯ (7 examples omitted)	⋯	⋯	⋯	⋯
Deep exponential family	105	6	6	0
Deep Markov model	112	1	1	0
Hidden Markov model	137	2	2	0
Single-cell annotation	147	3	3	0
Attend-infer-repeat	174	1	1	1
Conditional VAE	205	1	1	0

due to branching

due to div-by-0

Evaluation Results



ndom var’s

efault optimizer
d) # rv (Unsound)
1
…
0
0
0
0
1
0

due to branching

due to div-by-0

High-Level Messages

- It is subtle to do smoothness analysis properly (automatically, soundly, precisely enough).

  One reason: Make assumptions on target smoothness, which are easily violated.

- There are some PL research opportunities for ML (which are less explored).

  This work: Static analysis for automatic planning of inference algorithms.

Thanks for your attention!

Our Approach: Soundness

<u>Theorem</u> Our ϕ-smoothness analysis is sound if ϕ satisfies five assumptions:

- Strictness: $\forall f = \lambda x.\bot : \mathbb{R}^n \rightharpoonup \mathbb{R}^m.\quad f \in \phi.$
- Projection: $\forall f = \mathrm{proj}_{n\to m} : \mathbb{R}^n \rightharpoonup \mathbb{R}^m.\quad f \in \phi.$
- Restriction: $\forall f: \mathbb{R}^n \rightharpoonup \mathbb{R}^m, x \in \mathbb{R}^k (k \le n).\ f \in \phi \implies f(x, -) \in \phi.$
- Composition: $\forall f: \mathbb{R}^n \rightharpoonup \mathbb{R}^m, g: \mathbb{R}^m \rightharpoonup \mathbb{R}^l.\quad f, g \in \phi \implies (g \circ f) \in \phi.$
- Pairing: $\forall f: \mathbb{R}^n \rightharpoonup \mathbb{R}^m, g: \mathbb{R}^n \rightharpoonup \mathbb{R}^l.\quad f, g \in \phi \implies (f, g) \in \phi.$

Target smoothness property	A3 (proj.)	A4 (pair.)	A5 (rest.)	A6 (comp.)	A7 (stri.)
cont. (C^0)	o	o	o	o	o
locally Lipschitz (= $\phi^{(l)}$)	o	o	o	o	o
uniformly cont.	o	o	o	o	o
Lipschitz cont.	o	o	o	o	o
jointly diff.	o	o	o	o	o
continuously diff. (C^1)	o	o	o	o	o
smooth (C^∞)	o	o	o	o	o
real analytic (C^ω)	o	o	o	o	o
partially cont. (= $\phi^{(pc)}$)	o	o	o	×	o
partially diff. (= $\phi^{(pd)}$)	o	o	o	×	o
almost-everywhere cont.	o	o	×	×	o

Our Approach: Soundness

Target smoothness property	A3 (proj.)	A4 (pair.)	A5 (rest.)	A6 (comp.)	A7 (stri.)
cont. (C^0)	○	○	○	○	○
locally Lipschitz ($= \phi^{(l)}$)	○	○	○	○	○
uniformly cont.	○	○	○	○	○
Lipschitz cont.	○	○	○	○	○
diff. ($= \phi^{(d)}$)	○	○	○	○	○
continuously diff. (C^1)	○	○	○	○	○
smooth (C^∞)	○	○	○	○	○
real analytic (C^ω)	○	○	○	○	○
partially cont. ($= \phi^{(pc)}$)	○	○	○	×	○
partially diff. ($= \phi^{(pd)}$)	○	○	○	×	○
almost-everywhere cont.	○	○	×	×	○
almost-everywhere diff.	○	○	×	×	○
coordinatewise non-decreasing	○	○	○	○	○
locally bounded	○	○	○	○	○
bounded	×	○	○	○	○
Borel measurable	○	○	○	○	○
locally integrable	○	○	×	×	○
integrable	×	○	×	×	○

Evaluation Results

Reparametrization trick is incorrect in presence of discontinuity #2277

Open | 3 tasks | **fritzo** opened this issue on Jan 22, 2020 · 0 comments

fritzo commented on Jan 22, 2020 • edited | Member

This issue was raised by **@hongseok-yang**.

In models where likelihoods discontinuously depend on a latent variable, it is incorrect to use the parameterization trick for that variable in the guide. A correct workaround is to substitute the guide distribution with a non-reparameterized copy. For example:

```
def model(data):
    z = pyro.sample("z", dist.Normal(0,1))
    p = z.round().clamp(min=0.2, max=0.8)
    pyro.sample("x", dist.Bernoulli(p), obs=tensor(0.))

def guide(data):
    loc = pyro.param("loc", torch.tensor(0.))
    scale = pyro.param("scale", torch.tensor(1.), constraint=positive)
-   pyro.sample("z", dist.Normal(loc, 1))
+   pyro.sample("z", NonreparameterizedNormal(loc, scale))
```

Assignees

No one assigned

Labels

bug usability

Projects

None yet

Milestone

No milestone

Development

No branches or pull r