ωPAP Spaces
reasoning denotationally about differentiable & probabilistic programs

Mathieu Huot*, **Alexander Lew***, Vikash Mansinghka, Sam Staton




LICS ‘23

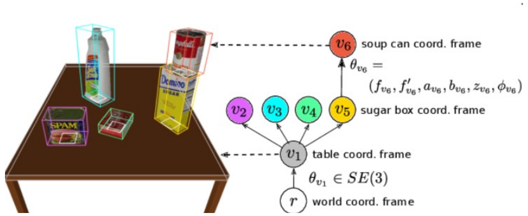
* Equal Contribution

Differentiable Programming

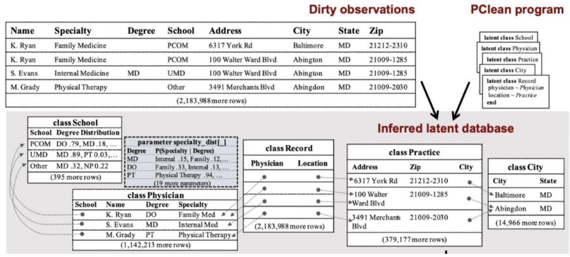
Torch TensorFlow Enzyme JAX Taichi MXNet

Probabilistic Programming

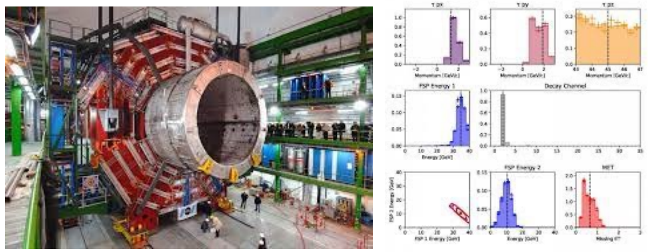
Gen SPPL Birch Stan ProbTorch Pyro



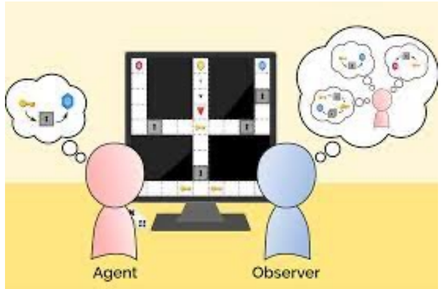
3D Scene Understanding
(Gothoskar et al. 2021)



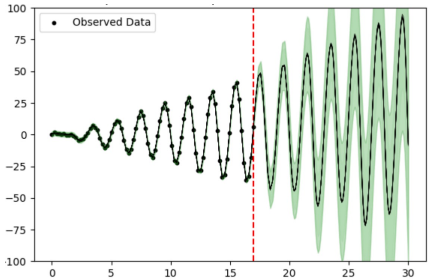
Data Deduplication and Cleaning
(Lew et al. 2021)



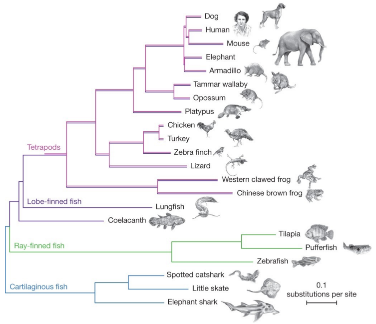
LHC Particle Mass Inference
(Baydin et al. 2019)



Online Goal Inference
(Zhi-Xuan et al. 2020)



Time Series Forecasting
(Saad et al. 2023)



Statistical Phylogenetics
(Ronquist et al. 2021)

AI / Robotics

Data Analysis

Natural Sciences

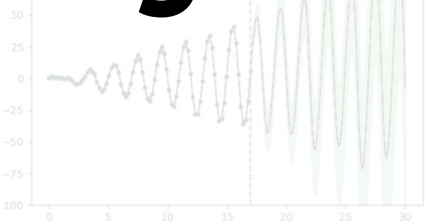
3D Scene Understanding
(Gothoskar et al. 2021)

Data Deduplication and Cleaning
(Lew et al. 2021)

LHC Particle Mass Inference
(Baydin et al. 2019)

Modeling Languages

Online Goal Inference
(Zhi-Xuan et al. 2020)



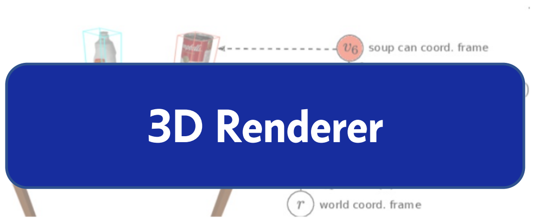
Time Series Forecasting
(Saad et al. 2023)

Statistical Phylogenetics
(Ronquist et al. 2021)

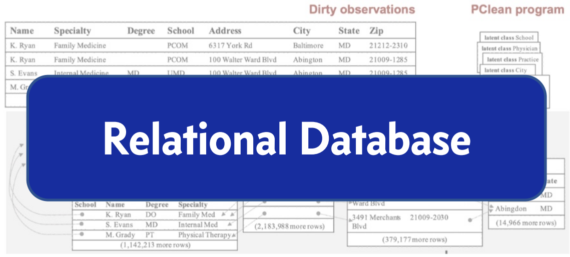
AI / Robotics

Data Analysis

Natural Sciences



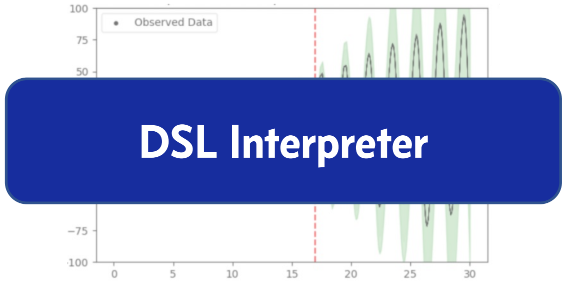
3D Scene Understanding
(Gothoskar et al. 2021)



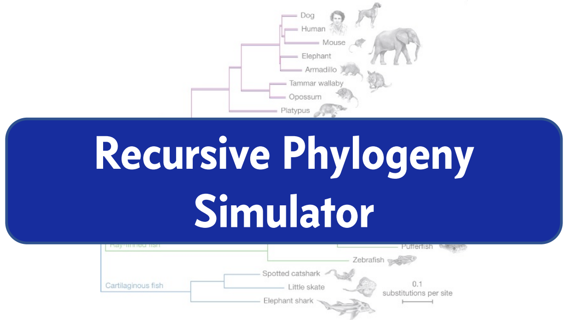
Data Deduplication and Cleaning
(Lew et al. 2021)

LHC Particle Mass Inference
(Baydin et al. 2019)

Online Goal Inference
(Zhi-Xuan et al. 2020)



Time Series Forecasting
(Saad et al. 2023)



Statistical Phylogenetics
(Ronquist et al. 2021)

AI / Robotics

Data Analysis

Natural Sciences

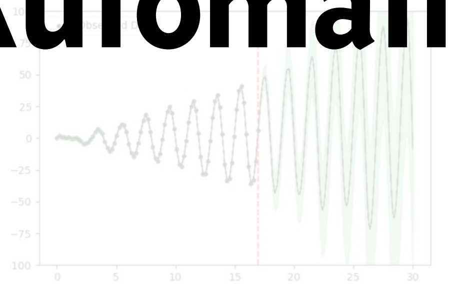
3D Scene Understanding
(Gothoskar et al. 2021)

Data Deduplication and Cleaning
(Lew et al. 2021)

LHC Particle Mass Inference
(Baydin et al. 2019)

Automation

Online Goal Inference
(Zhi-Xuan et al. 2020)



Time Series Forecasting
(Saad et al. 2023)

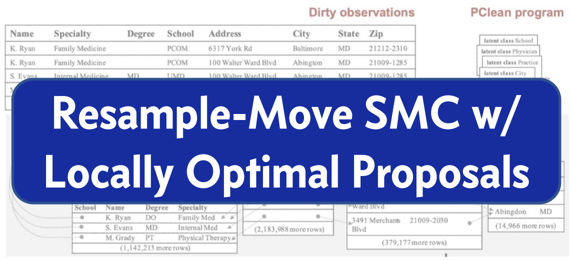
Statistical Phylogenetics
(Ronquist et al. 2021)

AI / Robotics

Data Analysis

Natural Sciences

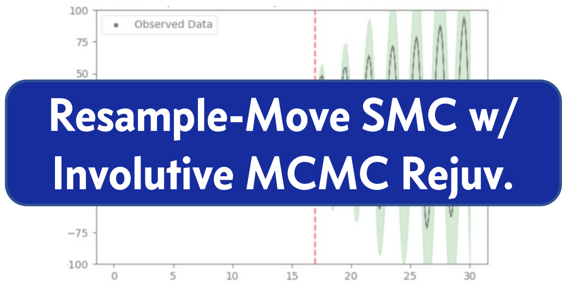
3D Scene Understanding
(Gothoskar et al. 2021)



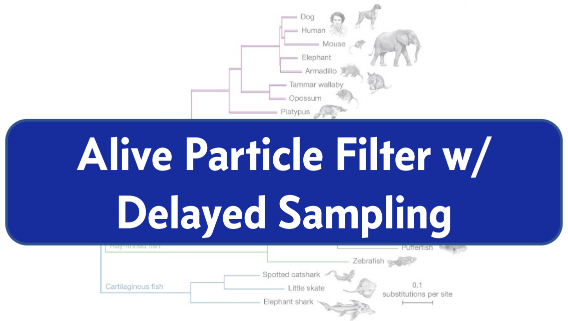
Data Deduplication and Cleaning
(Lew et al. 2021)

LHC Particle Mass Inference
(Baydin et al. 2019)

Online Goal Inference
(Zhi-Xuan et al. 2020)



Time Series Forecasting
(Saad et al. 2023)



Statistical Phylogenetics
(Ronquist et al. 2021)

AI / Robotics

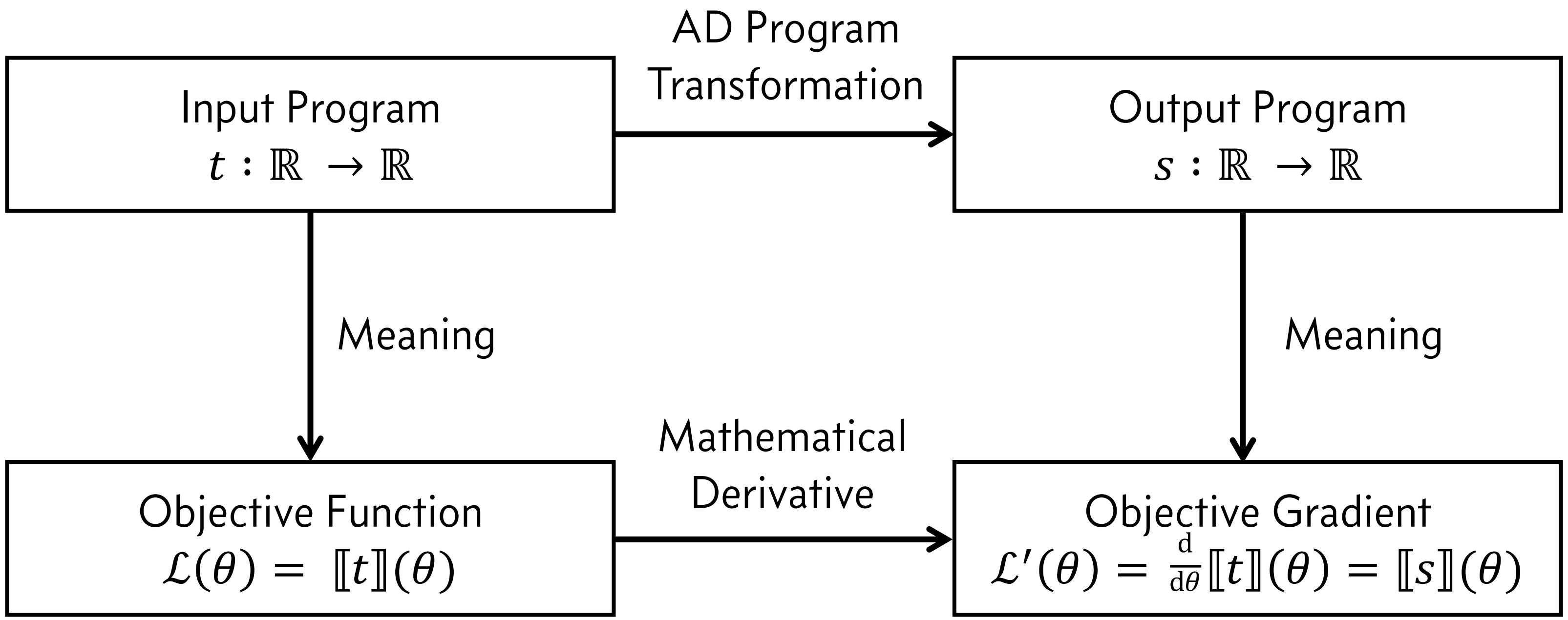
Data Analysis

Natural Sciences

Outline

- **What do we need semantics for?**
- Key challenge
- Our approach
- Main results

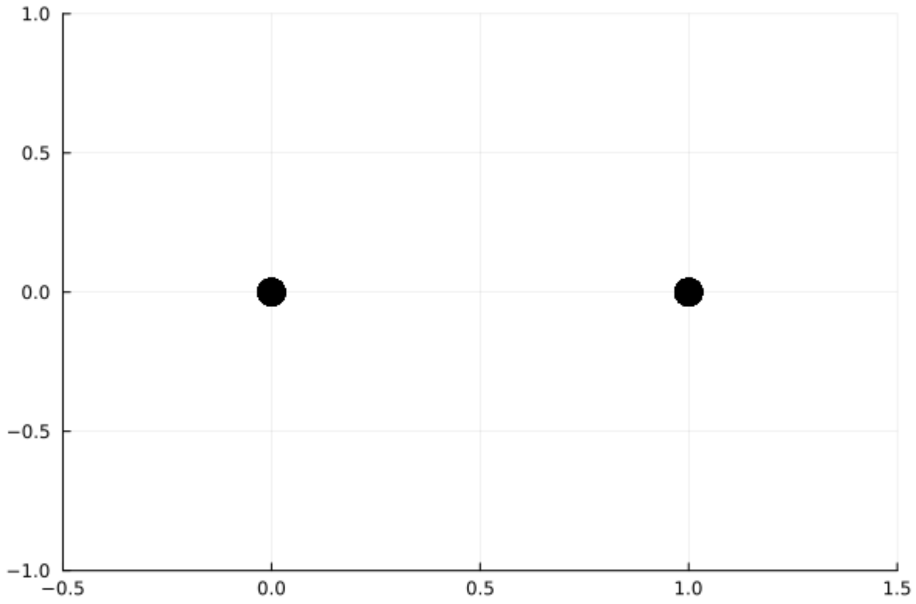
Correctness of Automation



Reasoning about Coverage

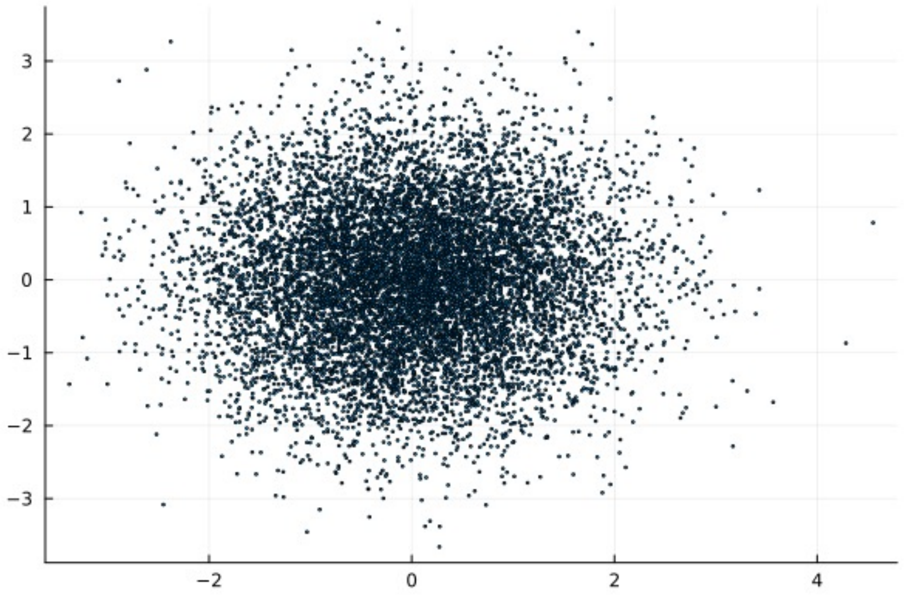
Reasoning about Coverage

```
fair_flip = do
  u <- sample
  if u < 0.5 then
    return 1
  else
    return 0
```



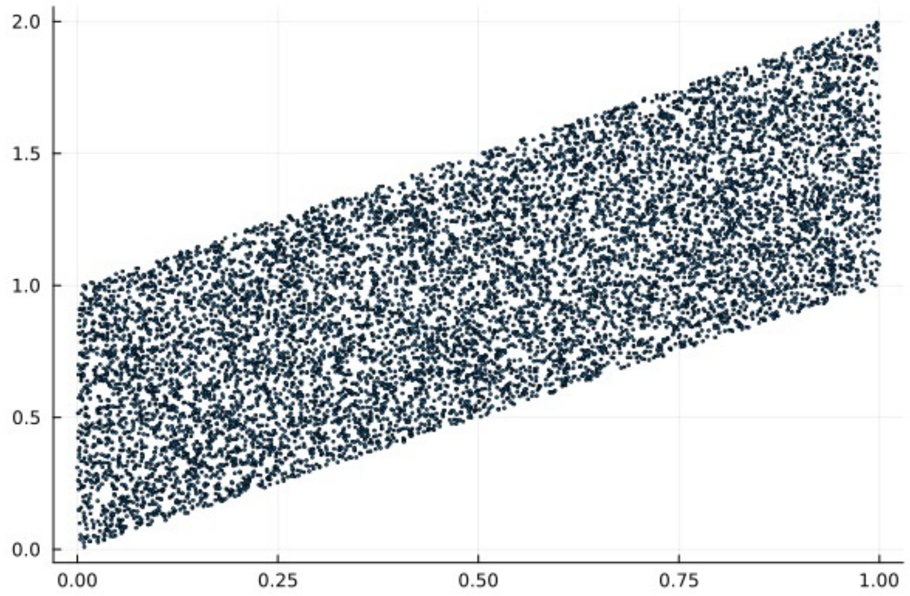
Reasoning about Coverage

```
std_normal = do
  u1 <- sample
  u2 <- sample
  let r=sqrt(-2*ln(u1))
  let t=2*pi*u2
  return (r*cos(t),
          r*sin(t))
```



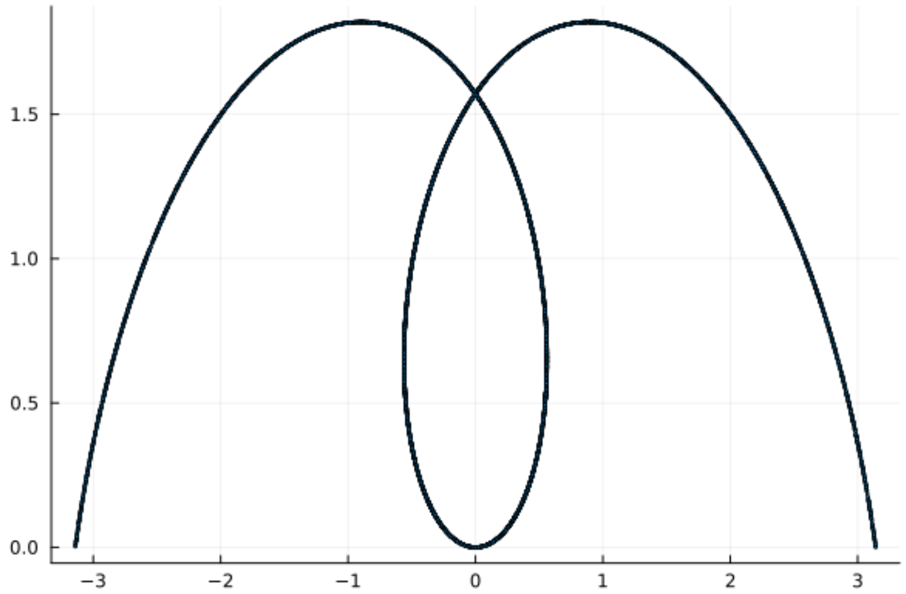
Reasoning about Coverage

```
uniform_band = do
  x <- sample
  y <- sample
  return (x, x + y)
```



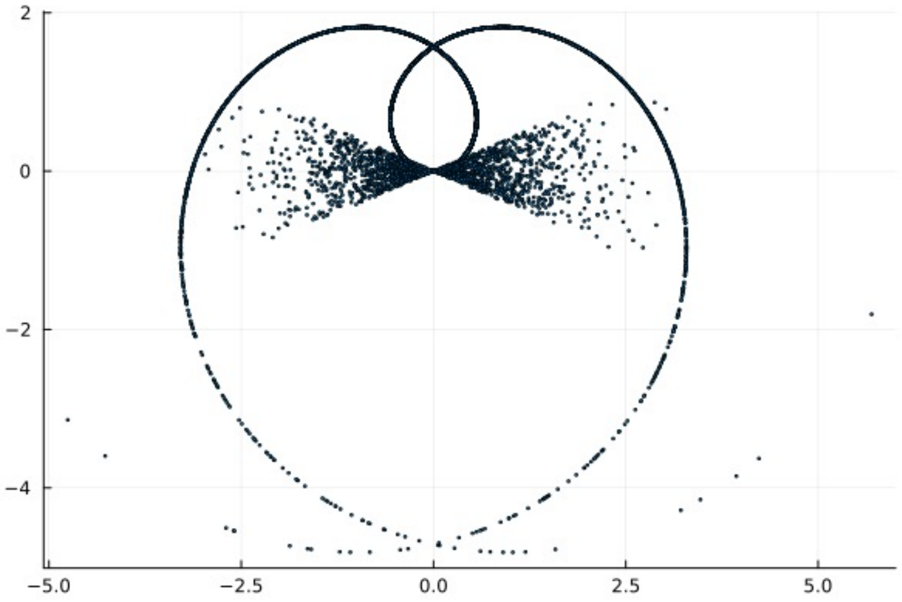
Reasoning about Coverage

```
mcdonalds = do
  x <- sample
  let t = 2*pi*x-pi
  return (t*cos(t),
          t*sin(t))
```

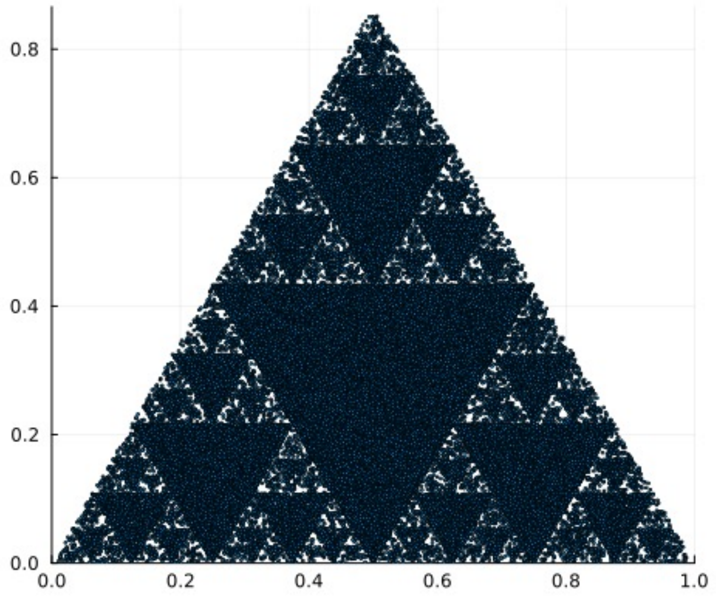


Reasoning about Coverage

```
bowtie_heart = do
  x <- std_normal
  let t = x * pi / 2
  r <- if |t| < 0.4
       then
         std_normal
       else
         return t
  return (r*cos(t), r*sin(t))
```

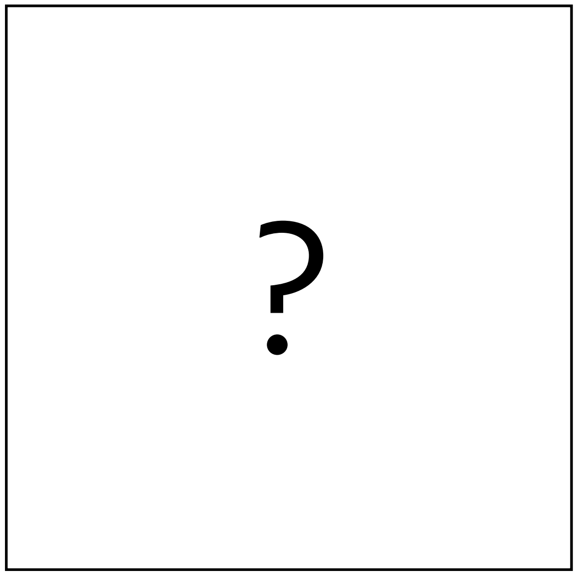


Reasoning about Coverage



$\mathcal{S}^c$

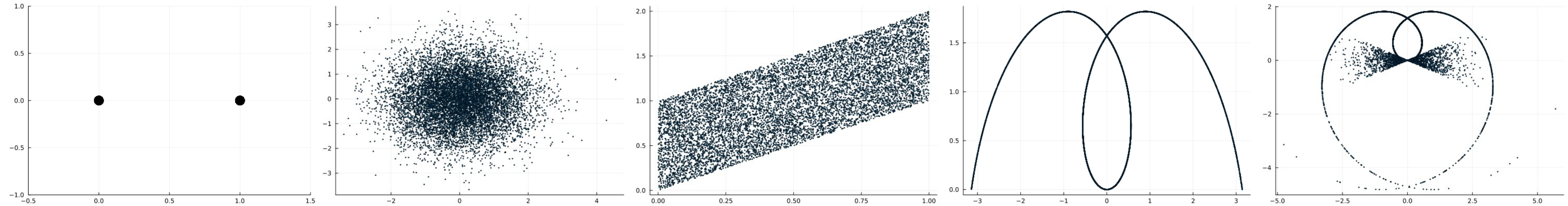
Complement of Sierpinski Triangle



$\mathcal{S}$

Sierpinski Triangle

Reasoning about Coverage



…

Can my language express all models users care about?

Do my interfaces and automation algorithms support any program the user could throw at me?

Reasoning about Coverage

We can use semantics to reason about the **definable measures**

$$\{[\![e]\!] \mid e : M\ \mathbb{R}^n\}$$

Outline

- What do we need semantics for?
- **Key challenge**
- Our approach
- Main results

Key challenge: the Goldilocks problem

ω**QBS**

Admits pathological examples

ω**Diff**

Can't model non-smooth primitives

ω**PAP**

This Work: Just Right?

Outline

- What do we need semantics for?
- Key challenge
- **Our approach**
- Main results

Concrete categories and higher-order recursion

With applications including probability, differentiability, and full abstraction

Cristina Matache
University of Oxford
Department of Computer Science
Oxford, UK

Sean Moss
University of Oxford
Department of Computer Science
Oxford, UK

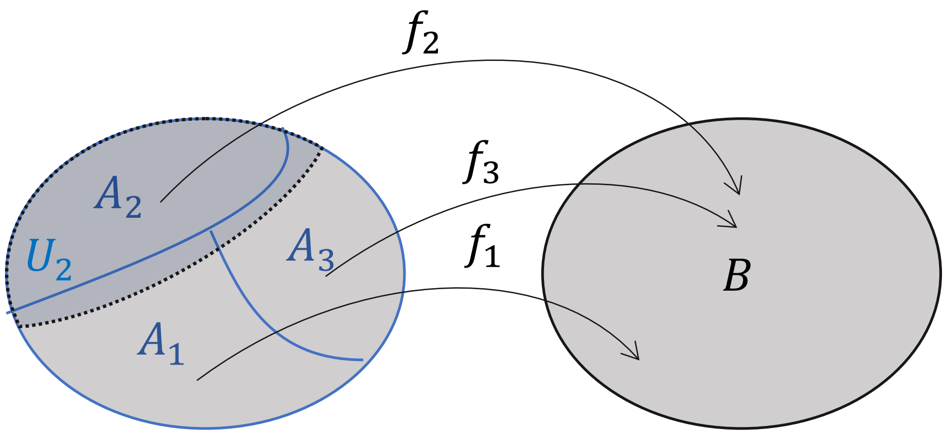
Sam Staton
University of Oxford
Department of Computer Science
Oxford, UK

Smooth $\mathbb{R} \to \mathbb{R}$ *functions*	*Measurable* $\mathbb{R} \to \mathbb{R}$ *functions*	*PAP* $\mathbb{R} \to \mathbb{R}$ *functions*
$\downarrow$	$\downarrow$	$\downarrow$
$\omega\mathbf{Diff}$	$\omega\mathbf{QBS}$	$\omega\mathbf{PAP}$

PAP functions

Piecewise analytic under analytic partition (PAP)
Lee et al. (2020)

A function is **PAP** if its domain can be partitioned into countably many **analytic sets**, restricted to each of which it is equal to some **analytic function**.



$$f(x) = \begin{cases} f_1(x) & \textbf{if } x \in A_1 \\ f_2(x) & \textbf{if } x \in A_2 \\ f_3(x) & \textbf{if } x \in A_3 \\ \vdots & \end{cases}$$

real-analytic functions defined on open $U_i \supseteq A_i$

analytic sets (finite intersections of sets of the form $\{x \in U | g(x) \leq 0\}$, for U open and g analytic) that **partition** A

Intuition: what functions are PAP?

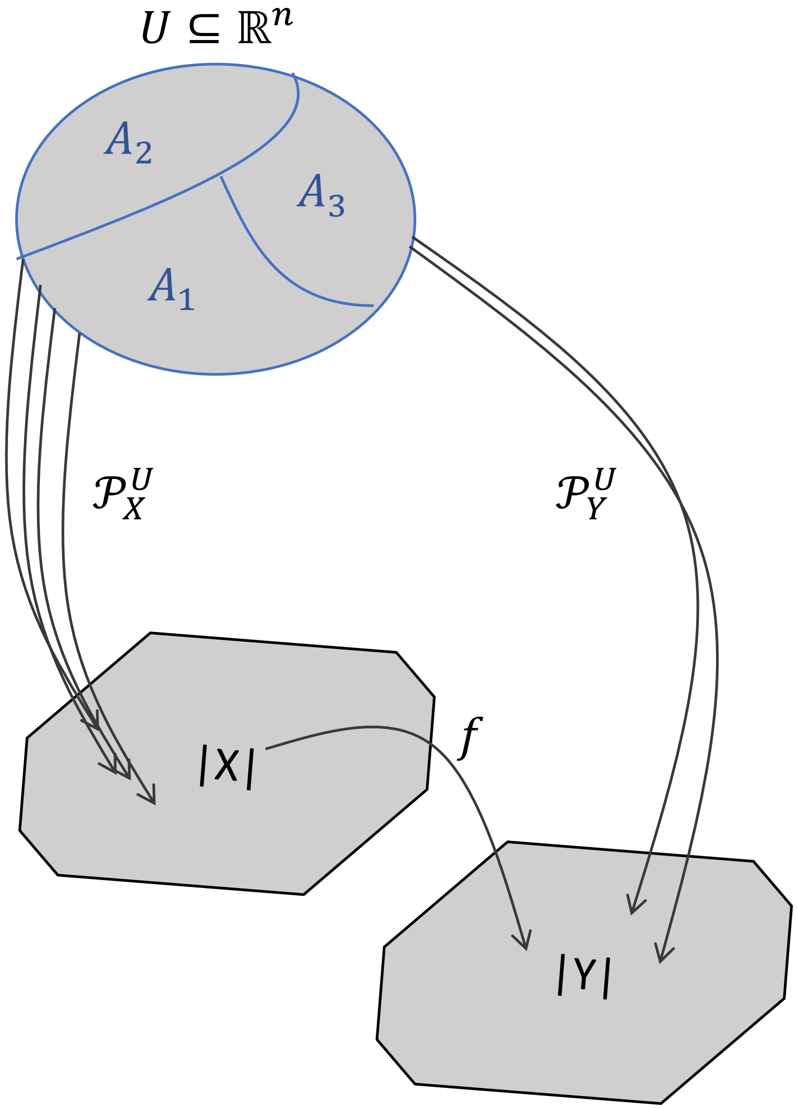
Includes (for example):

- All primitives in **numpy**
  arithmetic, trigonometry, comparison, floor, ceil, …
- All functions in **SpecialFunctions.jl**
- All functions in Haskell's numeric typeclasses

Does not include (for example):

- Indicator function for Cantor set
- Measurable bijections betw. $\mathbb{R}$, $\mathbb{R}^2$
- Hilbert curve

PAP spaces

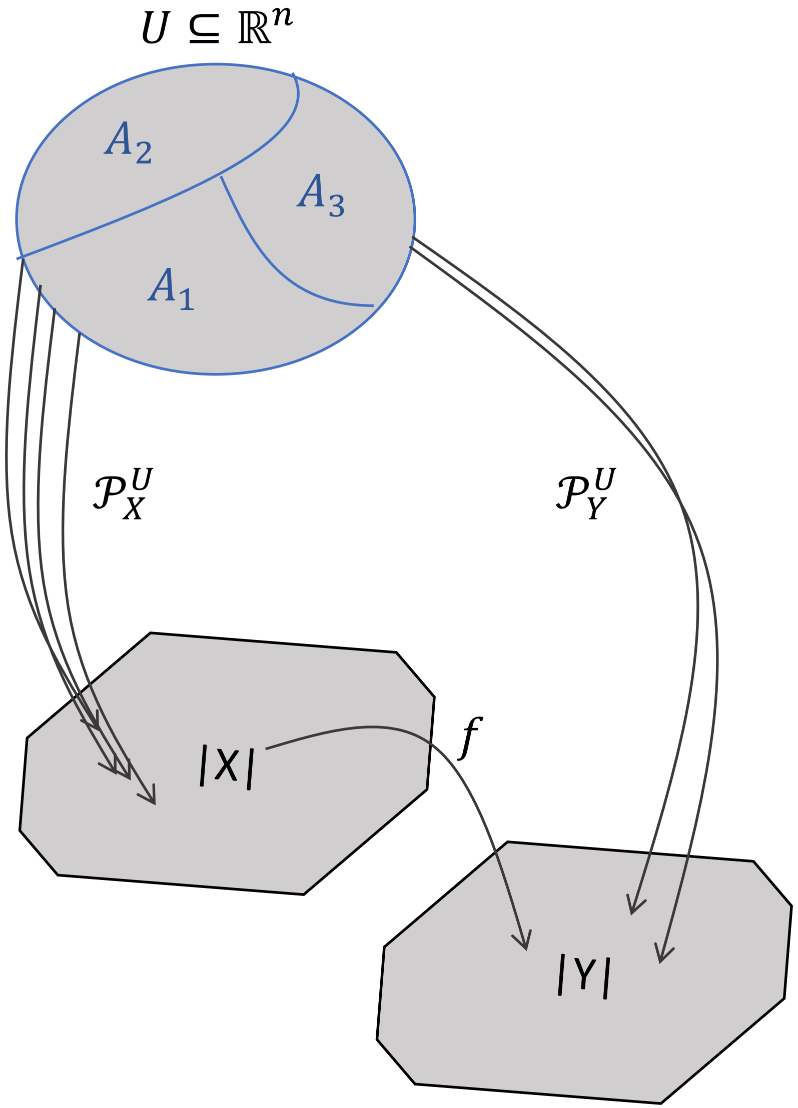


A **PAP space** X is:

- a set $|X|$, together with
- for each n and each set $U \in \mathbb{R}^n = \bigcup_{i \in \mathbb{N}} A_i$, a set $\mathcal{P}_X^U \subseteq |X|^U$ of *plots in* X, satisfying several closure properties.

A function $f : |X| \to |Y|$ is a **PAP morphism** if, whenever $\phi \in \mathcal{P}_X^U$, the composition $f \circ \phi \in \mathcal{P}_Y^U$.

ωPAP spaces



An **ωPAP space** X is:

- an $\boldsymbol{\omega}$**cpo** $|X|$,
- for each n and each set $U \in \mathbb{R}^n = \bigcup_{i\in\mathbb{N}} A_i$, a set $\mathcal{P}_X^U \subseteq |X|^U$ of **Scott-continuous** *plots in* X, satisfying several closure properties

A Scott-continuous function $f : |X| \to |Y|$ is an **ωPAP morphism** if, whenever $\phi \in \mathcal{P}_X^U$, the composition $f \circ \phi \in \mathcal{P}_Y^U$.

Language (CBV PCF with ℝ & probability)

$\tau ::= \mathbf{1} \mid \mathbb{R}^k \mid \mathbb{B} \mid \tau_1 \times \tau_2 \mid \tau_1 \to \tau_2 \mid M\,\tau$

$e ::= c \mid x \mid e_1\ e_2 \mid \textbf{if}\ e_1\ \textbf{then}\ e_2\ \textbf{else}\ e_3 \mid (e_1, e_2) \mid \pi_1 e \mid \pi_2 e \mid \textbf{return}\ e$
$\quad \mid \lambda x{:}\tau.\,e \mid \mu f{:}\tau_1 \to \tau_2.\,\lambda x{:}\tau_1.\,e \mid \textbf{sample} \mid \textbf{score}\ e \mid \textbf{do}\ \{m\}$

$m ::= e \mid x \leftarrow e; m$

ωQBS

Admits pathological examples—many interesting properties not provable

ωDiff

Can't model non-smooth primitives

ωPAP

This Work: Just Right?

Outline

- What do we need semantics for?
- Key challenge
- Our approach
- **Main results**

… for deterministic programs

New proof

Deterministic programs are almost-everywhere differentiable

Theorem. If $\vdash e : \mathbb{R}^m \to \mathbb{R}^n$, then $[\![e]\!]$ is differentiable at almost every input on which it is defined.

New proof

AD is almost everywhere correct

Theorem. If $\vdash e : \mathbb{R}^m \to \mathbb{R}^n$, then $[\![e]\!]$ is defined on the same inputs as $[\![AD\{e\}]\!]$, and for almost all such inputs x, $[\![AD\{e\}]\!](x)$ is the derivative of $[\![e]\!]$ at x.

Randomly initialized gradient descent converges, even when using AD

New result

Theorem. Let $\vdash e : \mathbb{R}^m \to \mathbb{R}$, with $[\![e]\!]$ total, L-smooth, and bounded below. Let $(\epsilon, x_0) \sim \mu$ for any μ supported on $\left(0, \frac{2}{L}\right) \times \mathbb{R}^m$. Then letting $x_t \coloneqq x_{t-1} - \epsilon \cdot [\![AD\{e\}]\!](x_{t-1})$,

$\lim_{t\to\infty} \nabla[\![e]\!](x_t) = 0$ with probability 1.

Gradient descent with pre-set learning rate ϵ may diverge when using AD

New result

```
# Recursive helper
def g(x, n):
    if x > 0:
        return ((x - n)*(x - n))/(lr * 2)
    if x == 0:
        return x / lr + n*n / (2*lr)
    else:
        return g(x+1, n+1)

def P(x):
  return g(x,0)
```

$$[\![P]\!] = \lambda x. \frac{x^2}{2\epsilon}$$

Gradient descent diverges to $-\infty$

… for probabilistic programs

New proof

Almost-surely terminating probabilistic programs have almost-everywhere differentiable weight functions

Theorem. If $\vdash e : M\ X$ and $[\![e]\!]$ almost surely halts, then $\mathbf{wt}[\![e]\!]$ is differentiable almost everywhere.

New result

All probabilistic programs on $\mathbb{R}^n$ are supported on a countable union of manifolds

Theorem. If $\vdash e : M\ \mathbb{R}^n$, then $[\![e]\!]$ is supported on a countable union of smooth submanifolds of $\mathbb{R}^n$.