Semantics of Higher-Order Probabilistic Programs with Continuous Distributions

Hongseok Yang
University of Oxford

Based on work with or by Chris Heunen, Ohad Kammar, Sam Staton, and Frank Wood
Learning outcome

• Can explain what one can do with higher-order probabilistic programming language.

• Can use measure theory to interpret prob. prog. with continuous distributions.

• Can use quasi-Borel space to interpret higher-order prob. prog. with conditioning.
References

1. A convenient category for higher-order probability theory. Heunen et al. LICS’17.

2. Commutative semantics for probabilistic programs. Staton. ESOP’17.
What is probabilistic programming?
(Bayesian) probabilistic modelling of data

1. Develop a new probabilistic (generative) model.
2. Design an inference algorithm for the model.
3. Using the algo., fit the model to the data.
(Bayesian) probabilistic modelling of data

in a prob. prog. language

1. Develop a new probabilistic (generative) model.
2. Design an inference algorithm for the model.
3. Using the algo., fit the model to the data.
(Bayesian) probabilistic modelling of data in a prob. prog. language as a program

1. Develop a new probabilistic (generative) model.
2. Design an inference algorithm for the model.
3. Using the algo., fit the model to the data.
(Bayesian) probabilistic modelling of data in a prob. prog. language as a program

1. Develop a new probabilistic (generative) model.

2. Design an inference algorithm for the model.

3. Using the algo., fit the model to the data.

a generic inference algo. of the language
Line fitting

Graph showing points plotted on a coordinate system.
Line fitting

\[ f(x) = s \cdot x + b \]
Bayesian generative model

\[ Y_i \quad b \quad S \]

\[ i = 0 \ldots 6 \]
Bayesian generative model

\[ f(x) = s \times x + b \]

Where \( i = 1 \ldots 5 \)

\[ y_i \sim \text{normal}(f(i), 1) \]

\[ s \sim \text{normal}(0, 2) \]

\[ b \sim \text{normal}(0, 6) \]
Bayesian generative model

\[
\begin{align*}
  s & \sim \text{normal}(0, 2) \\
  b & \sim \text{normal}(0, 6) \\
  f(x) &= s \cdot x + b \\
  y_i & \sim \text{normal}(f(i), 0.5) \\
  \text{where } i &= 0 \ldots 6
\end{align*}
\]
Bayesian generative model

\[ s \sim \text{normal}(0, 2) \]
\[ b \sim \text{normal}(0, 6) \]
\[ f(x) = s \times x + b \]
\[ y_i \sim \text{normal}(f(i), 0.5) \]
\[ \text{where } i = 0 \ldots 6 \]

Q: posterior of \((s, b)\) given \(y_0 = 0.6, \ldots, y_6 = 8.4\)?
Posterior of s and b given $y_i$'s

$$P(s, b \mid y_0, \ldots, y_6) = \frac{P(y_0, \ldots, y_6 \mid s, b) \times P(s, b)}{P(y_0, \ldots, y_6)}$$
Posterior of $s$ and $b$ given $y_i$'s

$$P(s, b \mid y_0, \ldots, y_6) = \frac{P(y_0, \ldots, y_6 \mid s, b) \times P(s, b)}{P(y_0, \ldots, y_6)}$$
Posterior of s and b given $y_i$'s

\[ P(s, b \mid y_0, \ldots, y_6) = \frac{P(y_0, \ldots, y_6 \mid s, b) \times P(s, b)}{P(y_0, \ldots, y_6)} \]
Posterior of s and b given $y_i$'s

$$P(s, b \mid y_0, \ldots, y_6) = \frac{P(y_0, \ldots, y_6 \mid s, b) \times P(s, b)}{P(y_0, \ldots, y_6)}$$
Posterior of s and b given $y_i$'s

\[ P(s, b \mid y_0, \ldots, y_6) = \frac{P(y_0, \ldots, y_6 \mid s, b) \times P(s, b)}{P(y_0, \ldots, y_6)} \]
Posterior of s and b given \( y_i \)'s

\[
P(s, b | y_0, ..., y_6) = \frac{P(y_0, ..., y_6 | s, b) \times P(s, b)}{P(y_0, ..., y_6)}
\]
(almost) Anglican program

(let [s (sample (normal 0 2))
     b (sample (normal 0 6))
     f (fn [x] (+ (* s x) b))]
    (observe (normal (f 0) .5) .6)
    (observe (normal (f 1) .5) .7)
    (observe (normal (f 2) .5) 1.2)
    (observe (normal (f 3) .5) 3.2)
    (observe (normal (f 4) .5) 6.8)
    (observe (normal (f 5) .5) 8.2)
    (observe (normal (f 6) .5) 8.4)
    (predict :sb [s b])
(almost) Anglican program

(let [s (sample (normal 0 2))
     b (sample (normal 0 6))
     f (fn [x] (+ (* s x) b))]

(observe (normal (f 0) .5) .6)
(observe (normal (f 1) .5) .7)
(observe (normal (f 2) .5) 1.2)
(observe (normal (f 3) .5) 3.2)
(observe (normal (f 4) .5) 6.8)
(observe (normal (f 5) .5) 8.2)
(observe (normal (f 6) .5) 8.4)
(almost) Anglican program

(let [s (sample (normal 0 2))
     b (sample (normal 0 6))
     f (fn [x] (+ (* s x) b))]
  
  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4)

[s b])

NB: (predict :sb [s b]) should be used instead of [s b] in Anglican
Samples from prior
Samples from posterior
Semantic challenges
(let [s (sample (normal 0 2))
    b (sample (normal 0 6))
    f (fn [x] (+ (* s x) b))]

  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4)

  [s b])
(let [s (sample (normal 0 2))
    b (sample (normal 0 6))
    f (fn [x] (+ (* s x) b))]

  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4)

  [s b])

1. Continuous distributions.
Challenge 1: Continuous distributions

- Need care for handling continuous distributions on $\mathbb{R}$, to avoid paradoxes.
- Something like measure theory needed.
- Complex math.
(let [s (sample (normal 0 2))
b (sample (normal 0 6))
f (fn [x] (+ (* s x) b))]

(observe (normal (f 0) .5) .6)
(observe (normal (f 1) .5) .7)
(observe (normal (f 2) .5) 1.2)
(observe (normal (f 3) .5) 3.2)
(observe (normal (f 4) .5) 6.8)
(observe (normal (f 5) .5) 8.2)
(observe (normal (f 6) .5) 8.4)

[s b])

1. Continuous distributions.
2. Higher-order functions.
(let [s (sample (normal 0 2))
     b (sample (normal 0 6))
     f (fn [x] (+ (* s x) b))]

  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4)

1. Continuous distributions.
2. Higher-order functions.
\[
\begin{align*}
(\text{let } [F \ (fn \ [] \\
\quad (\text{let } [s \ (\text{sample} \ (\text{normal} \ 0 \ 2)) \\
\quad \quad b \ (\text{sample} \ (\text{normal} \ 0 \ 6))] \\
\quad \quad (fn \ [x] \ (+ \ (\ast \ s \ x) \ b))))) \\
\text{f} \ (F)] \\
(\text{observe} \ (\text{normal} \ (f \ 0) \ .5) \ .6) \\
(\text{observe} \ (\text{normal} \ (f \ 1) \ .5) \ .7) \\
(\text{observe} \ (\text{normal} \ (f \ 2) \ .5) \ 1.2) \\
(\text{observe} \ (\text{normal} \ (f \ 3) \ .5) \ 3.2) \\
(\text{observe} \ (\text{normal} \ (f \ 4) \ .5) \ 6.8) \\
(\text{observe} \ (\text{normal} \ (f \ 5) \ .5) \ 8.2) \\
(\text{observe} \ (\text{normal} \ (f \ 6) \ .5) \ 8.4)
\end{align*}
\]

1. Continuous distributions.
2. Higher-order functions.
(let [F (fn []
    (let [s (sample (normal 0 2))
          b (sample (normal 0 6))]
      (fn [x] (+ (* s x) b))))
  f (add-change-points F 0 6))]
(observe (normal (f 0) .5) .6)
(observe (normal (f 1) .5) .7)
(observe (normal (f 2) .5) 1.2)
(observe (normal (f 3) .5) 3.2)
(observe (normal (f 4) .5) 6.8)
(observe (normal (f 5) .5) 8.2)
(observe (normal (f 6) .5) 8.4)

1. Continuous distributions.
2. Higher-order functions.
Samples from posterior
(let [F (fn []
  (let [s (sample (normal 0 2))
        b (sample (normal 0 6))]
    (fn [x] (+ (* s x) b)))]
  f (add-change-points F 0 6)]
  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4)
)

1. Continuous distributions.
2. Higher-order functions.
(let [\[F (fn []
    (let [s (sample (normal 0 2))
        b (sample (normal 0 6))]
      (fn [x] (+ (* s x) b))))
  f (add-change-points F 0 6))]
  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4))

1. Continuous distributions.
2. Higher-order functions.
Samples from posterior
Challenge 2: Higher-order functions

Measure theory doesn’t support HO fns well.

\[ \text{ev} : (\mathbb{R} \rightarrow_m \mathbb{R}) \times \mathbb{R} \rightarrow \mathbb{R}, \quad \text{ev}(f,x) = f(x). \]

[Aumann 61] ev is not measurable no matter which σ-algebra is used for \( \mathbb{R} \rightarrow_m \mathbb{R} \).

[Cor] The category of measurable spaces is not cartesian closed.
1. Continuous distributions.
2. Higher-order functions.

(let [F (fn []
  (let [s (sample (normal 0 2))
       b (sample (normal 0 6))]
      (fn [x] (+ (* s x) b))))
  f (add-change-points F 0 6)]
  (observe (normal (f 0) .5) .6)
  (observe (normal (f 1) .5) .7)
  (observe (normal (f 2) .5) 1.2)
  (observe (normal (f 3) .5) 3.2)
  (observe (normal (f 4) .5) 6.8)
  (observe (normal (f 5) .5) 8.2)
  (observe (normal (f 6) .5) 8.4)
1. Continuous distributions.
2. Higher-order functions.
3. Conditioning and prog. eqs.
Challenge 3: Conditioning and prog. eqs

\[
\begin{bmatrix} e: \text{real} \end{bmatrix} \in M(\mathbb{R})
\]

- M should model prob. computations.
- M should validate equations from statistics.
- M should be commutative.
- Difficult to find such M due to conditioning.
Challenge 3: Conditioning and prog. eqs

\[ \{ e: real \} \in M(\mathbb{R}) \]

- M should model prob. computations.
- M should validate equations from statistics.
- M should be commutative.
- Difficult to find such M due to conditioning.

only certain measures
nonfinite measures
Challenge 3: Conditioning and prog. eqs

- $[e: \text{real}] \in M(\mathbb{R})$
- nearly-finite measures
- nonfinite measures

- M should model prob. computations.
- M should validate equations from statistics.
- M should be commutative.
- Difficult to find such M due to conditioning.
(let [F (fn []
    (let [s (sample (normal 0 2))
          b (sample (normal 0 6))]
        (fn [x] (+ (* s x) b))))
  f (add-change-points F 0 6))
(observe (normal (f 0) .5) .6)
(observe (normal (f 1) .5) .7)
(observe (normal (f 2) .5) 1.2)
(observe (normal (f 3) .5) 3.2)
(observe (normal (f 4) .5) 6.8)
(observe (normal (f 5) .5) 8.2)
(observe (normal (f 6) .5) 8.4)

1. Continuous distributions.
2. Higher-order functions.
3. Conditioning and prog. eqs.
(let [F (fn []
   (let [s (sample (normal 0 2))
        b (sample (normal 0 6))]
      (fn [x] (+ (* s x) b))))
  f (add-change-points F 0 6))
(observe (normal (f 0) .5) .6)
(observe (normal (f 1) .5) .7)
(observe (normal (f 2) .5) 1.2)
(observe (normal (f 3) .5) 3.2)
(observe (normal (f 4) .5) 6.8)
(observe (normal (f 5) .5) 8.2)
(observe (normal (f 6) .5) 8.4)

1. Continuous distributions.
2. Higher-order functions.
3. Conditioning and prog. eqs.

Quasi-Borel space (QBS)
Big picture 1: Extend measure theory using category theory.
1. Continuous distr.
2. Higher-order fns.
1. Continuous distr.
2. Higher-order fns.

Meas_B
1. Continuous distr.
2. Higher-order fns.

$\text{Meas}_B$

Yoneda embedding

$[\text{Meas}_B^{\text{op}}, \text{Set}]_{\Pi}$
1. Continuous distr.
2. Higher-order fns.

$\text{Meas}_B$

Yoneda embedding

$[\text{Meas}_B^{op}, \text{Set}]_\Pi$

Preserves nearly all the structures
1. Continuous distr.
2. Higher-order fns.

Meas_B

Yoneda embedding

[Meas_B^{op}, Set]_{\Pi}

Enough structure for function types
1. Continuous distr.
2. Higher-order fns.

\[
\text{Meas}_B \\
\downarrow \text{Yoneda embedding} \\
[\text{Meas}_{B^\text{op}}, \text{Set}]_\Pi
\]
1. Continuous distr.
2. Higher-order fns.

\[ \text{Meas}_B \xrightarrow{\text{Yoneda embedding}} \text{Meas}_B^{\text{op}} \text{, Set} \text{, } \Pi \xrightarrow{\text{QBS}} \text{Full subcat. of separated functors} \]
1. Continuous distr.
2. Higher-order fns.

Function spaces (CCC).
Concrete (extensional).

Meas\textsubscript{B} \longrightarrow [\text{Meas}_{B^{\text{op}}, \text{Set}}]_{\Pi} \longrightarrow \text{QBS}

Yoneda embedding

Full subcat. of separated functors
1. Continuous distr.
2. Higher-order fns.

$\text{Meas}_B$ \hspace{2cm} $\text{QBS}$

- Yoneda embedding
- Full subcat. of separated functors

Strong monad of s-finite kernels

$\text{SFinKer}$
1. Continuous distr.
2. Higher-order fns.

\[ \text{Meas}_B \rightarrow [\text{Meas}_B^{\text{op}}, \text{Set}]_\Pi \rightarrow \text{QBS} \rightarrow \text{SFinKer} \]

- Yoneda embedding
- Full subcat. of separated functors
Big picture 2: Random element first.
Random element $\alpha$ in $X$
Random element $\alpha$ in $X$

$\alpha : \Omega \rightarrow X$

- $X$ - set of values.
- $\Omega$ - set of random seeds.
- Random seed generator.
Random element $\alpha$ in $X$

in measure theory

$\alpha : \Omega \to X$

- $X$ - set of values.
- $\Omega$ - set of random seeds.
- Random seed generator.
Random element $\alpha$ in $X$

in measure theory

$\alpha : \Omega \rightarrow X$

• $X$ - set of values.
• $\Omega$ - set of random seeds.
• Random seed generator.

1. $\Sigma \subseteq 2^\Omega$, $\Theta \subseteq 2^X$
Random element $\alpha$ in $X$

in measure theory

$\alpha : \Omega \rightarrow X$

- $X$ - set of values.
- $\Omega$ - set of random seeds.
- Random seed generator.

1. $\Sigma \subseteq 2^\Omega$, $\Theta \subseteq 2^X$
2. $\mu : \Sigma \rightarrow [0,1]$
Random element $\alpha$ in $X$ in measure theory

$\alpha : \Omega \to X$ is a random element if $\alpha^{-1}(A) \in \Sigma$ for all $A \in \Theta$

- $X$ - set of values.
- $\Omega$ - set of random seeds.
- Random seed generator.

1. $\Sigma \subseteq 2^\Omega$, $\Theta \subseteq 2^X$
2. $\mu : \Sigma \to [0, 1]$
Random element $\alpha$ in $X$

$\alpha : \Omega \rightarrow X$

- $X$ - set of values.
- $\Omega$ - set of random seeds.
- Random seed generator.
Random element $\alpha$ in $X$

in quasi-Borel spaces

$\alpha : \Omega \rightarrow X$

- $X$ - set of values.
- $\Omega$ - set of random seeds.
- Random seed generator.
Random element $\alpha$ in $X$

in quasi-Borel spaces

$\alpha : \mathbb{R} \rightarrow X$

- $X$ - set of values.
- $\mathbb{R}$ - set of random seeds.
- Random seed generator.

1. $\mathbb{R}$ as random source
2. Borel subsets $\mathcal{B} \subseteq 2^\mathbb{R}$
Random element $\alpha$ in $X$

in quasi-Borel spaces

$\alpha : \mathbb{R} \rightarrow X$

- $X$ - set of values.
- $\mathbb{R}$ - set of random seeds.
- Random seed generator.

1. $\mathbb{R}$ as random source
2. Borel subsets $\mathcal{B} \subseteq 2^\mathbb{R}$
Random element $\alpha$ in $X$

in quasi-Borel spaces

$\alpha : \mathbb{R} \to X$

- $X$ - set of values.
- $\mathbb{R}$ - set of random seeds.
- Random seed generator.

1. $\mathbb{R}$ as random source
2. Borel subsets $\mathcal{B} \subseteq 2^\mathbb{R}$
3. $M \subseteq [\mathbb{R} \rightarrow X]$
Random element $\alpha$ in $X$

in quasi-Borel spaces

$\alpha : \mathbb{R} \rightarrow X$ is a random variable if $\alpha \in M$

- $X$ - set of values.
- $\mathbb{R}$ - set of random seeds.
- Random seed generator.

1. $\mathbb{R}$ as random source
2. Borel subsets $\mathcal{B} \subseteq 2^\mathbb{R}$
3. $M \subseteq [\mathbb{R} \rightarrow X]$
• Measure theory:

• Measurable space \((X, \Theta \subseteq 2^X)\).

• Random element is an induced concept.

• QBS:

• Quasi-Borel space \((X, M \subseteq [\mathbb{R} \rightarrow X])\).

• \(M\) is the set of random elements.
Rest of this tutorial

1. Baby measure theory.
   PL with cont. distribution.

2. Quasi-Borel space (QBS).
   PL with cont. distr. & HO fns.

3. SFinKer monad on QBS.
   PL with cont. distr., HO fns & conditioning.
Rest of this tutorial

1. Baby measure theory.
   PL with cont. distribution.

2. Quasi-Borel space (QBS).
   PL with cont. distr. & HO fns.

3. SFinKer monad on QBS.
   PL with cont. distr., HO fns & conditioning.
Programming language

• Will be sloppy about its syntax.

• Higher-order call-by-value probabilistic PL.

\[ t ::= \text{bool} \mid \text{real} \mid t \times t \mid t \rightarrow t \]

\[ e ::= \ldots \]
Baby measure theory
How to specify prob. $\mu$?
How to specify prob. $\mu$?

$X = \{0, 1, 2\}$.

Define $\mu : X \rightarrow [0,1]$. E.g., $\mu = [0.4, 0.4, 0.2]$.

Lifted $\mu : 2^X \rightarrow [0,1]$ by $\mu(A) = \sum_{x \in A} \mu(x)$. 
How to specify prob. $\mu$?

$X = \mathbb{R}$.

Define $\mu : X \to [0,1]$.

Lifted $\mu : 2^X \to [0,1]$ by $\mu(A) = \sum_{x \in A} \mu(x)$. 
How to specify prob. \( \mu \)?

\[ X = \mathbb{R}. \]

Define \( \mu : X \rightarrow [0, 1] \).

Lifted \( \mu : 2^X \rightarrow [0, 1] \) by \( \mu(A) = \sum_{x \in A} \mu(x) \).

Uncountable sum. Typically \( \infty \).
How to specify prob. $\mu$?

$X = \mathbb{R}$.

Define $\mu : X \rightarrow [0,1]$.

Lifted $\mu : 2^X \rightarrow [0,1]$ by $\mu(A) = \sum_{x \in A} \mu(x)$.

Define
How to specify prob. \( \mu \)?

\[ \mathcal{X} = \mathbb{R}. \]

Define \( \mu : \mathcal{X} \to [0,1] \)

Lifted \( \mu : 2^\mathcal{X} \to [0,1] \) by \( \mu(A) = \sum_{x \in A} \mu(x) \)

Define a good collection \( \Sigma \subset 2^\mathcal{X} \).
Define \( \mu : \Sigma \to [0,1] \) with some care.
How to specify prob. $\mu$?

$X = \mathbb{R}$.

Define $\mu : X \to [0,1]$.

Lifted $\mu : 2^X \to [0,1]$ by $\mu(A) = \sum_{x \in A} \mu(x)$.

Define $\sigma$-algebra

Pick a good collection $\Sigma \subseteq 2^X$.

Define $\mu : \Sigma \to [0,1]$ with some care.

probability measure
Let $\Sigma \subseteq 2^X$.

$\Sigma$ is a $\sigma$-algebra if it contains $X$, and is closed under countable union and set subtraction.

$(X, \Sigma)$ is a measurable space if $\Sigma$ is a $\sigma$-algebra.
Let $\Sigma \subseteq 2^X$.

$\Sigma$ is a $\sigma$-algebra if it contains $X$, and is closed under countable union and set subtraction.

$(X, \Sigma)$ is a measurable space if $\Sigma$ is a $\sigma$-algebra.

$\mu : \Sigma \rightarrow [0,1]$ is a probability measure if $\mu(X) = 1$ and $\mu(\biguplus_{n \in \mathbb{N}} A_n) = \sum_{n \in \mathbb{N}} \mu(A_n)$ for all disjoint $A_n$'s.

$(X, \Sigma, \mu)$ is a probability space if …
[Q] What are not measurable spaces?

1. \((\mathcal{B}, 2^\mathcal{B})\).
2. \((\mathcal{B} \times \mathcal{B}, \{ A \times B \mid A \in 2^\mathcal{B} \text{ and } B \in 2^\mathcal{B} \})\).
3. \((\mathbb{R}, \{ A \subseteq \mathbb{R} \mid A \text{ or } (\mathbb{R} - A) \text{ countable} \})\).
4. \((\mathbb{R}, \{ (r,s] \mid r < s \})\).
[Q] What are not measurable spaces?

1. \((\mathcal{B}, 2^\mathcal{B})\).

2. \((\mathcal{B} \times \mathcal{B}, \{A \times B \mid A \in 2^\mathcal{B} \text{ and } B \in 2^\mathcal{B}\})\).

3. \((\mathbb{R}, \{A \subseteq \mathbb{R} \mid A \text{ or } (\mathbb{R} - A) \text{ countable}\})\).

4. \((\mathbb{R}, \{(r, s] \mid r < s\})\).
[Q] Convert them to measurable spaces.

1. \((\mathcal{B}, 2^\mathcal{B})\).

2. \((\mathcal{B} \times \mathcal{B}, \{ A \times B | A \in 2^\mathcal{B} \text{ and } B \in 2^\mathcal{B} \})\).

3. \((\mathbb{R}, \{ A \subseteq \mathbb{R} | A \text{ or } (\mathbb{R} - A) \text{ countable} \})\).

4. \((\mathbb{R}, \{ (r,s] | r<s \})\).
[Q] Convert them to measurable spaces.

1. \((\mathcal{B}, 2^\mathcal{B})\).

2. \((\mathcal{B} \times \mathcal{B}, \{ A \times B \mid A \in 2^\mathcal{B} \text{ and } B \in 2^\mathcal{B} \})\).

3. \((\mathbb{R}, \{ A \subseteq \mathbb{R} \mid A \text{ or } (\mathbb{R} - A) \text{ countable} \})\).

4. \((\mathbb{R}, \{ [r,s] \mid r < s \})\).

Closure exists.

\(\sigma(\Pi)\) smallest \(\sigma\)-algebra containing \(\Pi\).
$(X, \Sigma), (Y, \Theta)$ - mBle spaces.

Product $\sigma$-algebra: $\Sigma \otimes \Theta = \sigma\{A \times B \mid A \in \Sigma, B \in \Theta\}$.

Product space: $(X, \Sigma) \times_m (Y, \Theta) = (X \times Y, \Sigma \otimes \Theta)$.

Borel $\sigma$-algebra on $\mathbb{R}$: $\mathcal{B} = \sigma\{(r, s] \mid r < s\}$.

Borel space: $(\mathbb{R}, \mathcal{B})$. 
Types mean mBle spaces
Types mean mBle spaces

\[ [\text{bool}] = (\mathbb{B}, 2^\mathbb{B}) \]

\[ [\text{real}] = (\mathbb{R}, \mathbb{B}) \]
Types mean mBle spaces

\[\begin{align*}
\text{[bool]} &= (\mathbb{B}, 2^\mathbb{B}) \\
\text{[real]} &= (\mathbb{R}, \mathbb{B}) \\
\text{[t \times t']} &= \text{[t]} \times_m \text{[t']} \\
\text{[x_1:t_1, \ldots, x_n:t_n]} &= \text{[t_1]} \times_m \ldots \times_m \text{[t_n]}
\end{align*}\]
(X, Σ), (Y, Θ) - mBle spaces.

f: X → Y is **measurable** (denoted f: X → m Y) if f⁻¹(A) ∈ Σ for all A ∈ Θ.
\((X, \Sigma), (Y, \Theta)\) - mBle spaces.

\(f: X \rightarrow Y\) is **measurable** (denoted \(f: X \rightarrow _mY\)) if \(f^{-1}(A) \in \Sigma\) for all \(A \in \Theta\).

\(k: X \times \Theta \rightarrow [0, 1]\) is a **prob. kernel** if \(k(x, -)\) is a prob. measure and \(k(-, A)\) is measurable for all \(x, A\).
Terms mean prob. kernels

\[ \Gamma \vdash e : t \] is a prob. kernel from \( \lbrack \Gamma \rbrack \) to \( \lbrack t \rbrack \).
Terms mean prob. kernels

\[
\Gamma \vdash e : t \] is a prob. kernel from \( \Gamma \) to \( t \).

\[
y : \text{real} \vdash y + \text{sample}(\text{norm}(0,1)) : \text{real}
\]
Terms mean prob. kernels

\[ \Gamma \vdash e : t \] is a prob. kernel from \( \Gamma \) to \( t \).

\[ [y : \text{real} \vdash y + \text{sample}(\text{norm}(0,1)) : \text{real}] \](r,A)
Terms mean prob. kernels

\[ \Gamma \vdash e : t \] is a prob. kernel from \( \Gamma \) to \( t \).

\[ y : \text{real} \vdash y + \text{sample}(\text{norm}(0,1)) : \text{real} \] \((r,A)\)

\[ = \int_A \text{density-norm}(s \mid r,1) \, ds. \]
Rest of this tutorial

1. Baby measure theory.
   PL with cont. distribution.

2. Quasi-Borel space (QBS).
   PL with cont. distr. & HO fns.

3. SFinKer monad on QBS.
   PL with cont. distr., HO fns & conditioning.
Quasi-Borel space
Quasi-Borel space - set with random elements.
Quasi-Borel space - set with random elements.

\[(X, M \subseteq \mathbb{R} \rightarrow X)\]

such that \(M\) has enough random elements.
Quasi-Borel space - set with random elements.

\((X, M \subseteq [\mathbb{R} \to X])\)

such that \(M\) has enough random elements.
Quasi-Borel space - set with random elements.

\[(X, M \subseteq [\mathbb{R} \to X])\]

such that \(M\) has enough random elements.
Quasi-Borel space - set with random elements.

\[(X, M \subseteq [\mathbb{R} \to X])\]

such that \(M\) has enough random elements.

1. \(M\) contains all constant functions.
Quasi-Borel space - set with random elements.

\[(X, M \subseteq [\mathbb{R} \to X])\]

such that \(M\) has enough random elements.

1. \(M\) contains all constant functions.

2. \((\alpha \circ \beta) \in M\) for all \(\alpha \in M\) and mBle \(\beta : \mathbb{R} \to \mathbb{R}\).
Quasi-Borel space - set with random elements.

\((X, \mathcal{M} \subseteq [\mathbb{R} \to X])\)

such that \(\mathcal{M}\) has enough random elements.

1. \(\mathcal{M}\) contains all constant functions.

2. \((\alpha \circ \beta) \in \mathcal{M}\) for all \(\alpha \in \mathcal{M}\) and mBle \(\beta : \mathbb{R} \to \mathbb{R}\).

3. If \(\mathbb{R} = \bigcup_{i \in \mathbb{N}} R_i\) with \(R_i \in \mathcal{B}\) and \(\alpha_1, \alpha_2, \ldots \in \mathcal{M}\),
   then \((\alpha_i \text{ when } R_i)_{i \in \mathbb{N}} \in \mathcal{M}\).

Here \((\alpha_i \text{ when } R_i)_{i \in \mathbb{N}}(r) = \alpha_i(r)\) for all \(r \in R_i\).
[Q] Pick a non-QBS.

1. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is a constant function}\})\).

2. \((\mathbb{R}, [\mathbb{R} \to \mathbb{R}])\).

3. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ measurable wrt. } \mathcal{B}\})\).
[Q] Pick a non-QBS.

1. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is a constant function}\})\).

2. \((\mathbb{R}, [\mathbb{R} \to \mathbb{R}])\).

3. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ measurable wrt. } \mathcal{B}\})\).
[Q] Turn it into a QBS.

1. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is a constant function}\})\).

2. \((\mathbb{R}, [\mathbb{R} \to \mathbb{R}])\).

3. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ measurable wrt. } \mathcal{B}\})\).
[Q] Turn it to a QBS.

1. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is a constant function}\})\).

2. \((\mathbb{R}, [\mathbb{R} \to \mathbb{R}])\).

3. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ measurable wrt. } \mathcal{B}\})\).
[Q] Turn it to a QBS.

Standard way of converting a set to a QBS.

1. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is a constant function}\})\).

2. \((\mathbb{R}, [\mathbb{R} \to \mathbb{R}])\).

3. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ measurable wrt. } \mathcal{B}\})\).
[Q] Turn it to a QBS.

Standard way of converting a set to a QBS.

1. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is a constant function}\})\).

2. \((\mathbb{R}, [\mathbb{R} \to \mathbb{R}])\).

3. \((\mathbb{R}, \{\alpha: \mathbb{R} \to \mathbb{R} \mid \alpha \text{ is measurable wrt. } \mathcal{B}\})\).

Standard way of converting a mBle space to a QBS.
(QBS) morphism

$(X,M), \ (Y,N) - $ QBSes.

$f : X \rightarrow Y$ is a **morphism** if $(fo\alpha) \in N$ for all $\alpha \in M$.

Maps random elements to random elements.

We will write $f : X \rightarrow_q Y$. 
[Th] QBSes form a cartesian closed category. So, they provide good product and function spaces.
[Q] What are the sets of random elements?

1. Product: \((X,M) \times_q (Y,N) = (Z,O)\).
   - \(Z = X \times Y, \quad \pi_1(x,y) = x, \quad \pi_2(x,y) = y.\)
   - \(O = ???\)

2. Fn space: \([ (X,M) \rightarrow_q (Y,N) ] = (Z,O) \]
   - \(Z = \{ f \mid f : X \rightarrow_q Y \}, \quad \text{ev}(f,x) = f(x)\)
   - \(O = ???\)
[Q] What are the sets of random elements?

1. Product: \((X,M) \times_q (Y,N) = (Z,O)\).
   - \(Z = X \times Y, \quad \pi_1(x,y) = x, \quad \pi_2(x,y) = y\).
   - \(O = ???\)

2. Fn space: \([ (X,M) \rightarrow_q (Y,N) ] = (Z,O)\)
   - \(Z = \{ f \mid f : X \rightarrow_q Y \}, \quad \text{ev}(f,x) = f(x)\).
   - \(O = ???\)
[Q] What are the sets of random elements?

1. Product: 
   \((X,M) \times \_q (Y,N) = (Z,O)\).
   - \(Z = X \times Y\), \(\pi_1(x,y) = x\), \(\pi_2(x,y) = y\).
   - \(O = \{ <\alpha,\beta> \mid \alpha \in M \text{ and } \beta \in N \}\).

2. Fn space: 
   \[\{(X,M) \rightarrow \_q (Y,N)\} = (Z,O)\]
   - \(Z = \{ f \mid f : X \rightarrow \_q Y \}\), \(\text{ev}(f,x) = f(x)\)
   - \(O = \{ \text{curry}(g) \mid g : \mathbb{R} \times \_q X \rightarrow \_q Y \}\).
Why works?

[NO] ev : \((\mathbb{R} \rightarrow_m \mathbb{R}) \times_m \mathbb{R} \rightarrow_m \mathbb{R}\)

vs

[YES] ev : \((\mathbb{R} \rightarrow_q \mathbb{R}) \times_q \mathbb{R} \rightarrow_q \mathbb{R}\)

Because the QBS product is more permissive.
Why works?

[NO] \( ev : (\mathbb{R} \rightarrow_{\text{mode}} \mathbb{R}) \times_{\text{mode}} \mathbb{R} \rightarrow_{\text{mode}} \mathbb{R} \)

vs

[YES] \( ev : (\mathbb{R} \rightarrow_{\text{QBS}} \mathbb{R}) \times_{\text{QBS}} \mathbb{R} \rightarrow_{\text{QBS}} \mathbb{R} \)

Because the QBS product is more permissive.
Types mean QBSes

\[
[\text{bool}] = \text{MStoQBS}(\mathbb{B}, 2^\mathbb{B})
\]

\[
[\text{real}] = \text{MStoQBS}(\mathbb{R}, \mathbb{B})
\]
Types mean QBSes

\[ [\text{bool}] = \text{MStoQBS}(\mathbb{B}, 2^\mathbb{B}) \]
\[ [\text{real}] = \text{MStoQBS}(\mathbb{R}, \mathbb{B}) \]
\[ [t \times t'] = [t] \times_q [t'] \]

\[ [x_1:t_1, \ldots, x_n:t_n] = [t_1] \times_q \ldots \times_q [t_n] \]
Types mean QBSes

\[
\begin{align*}
\llbracket \text{bool} \rrbracket &= \text{MStoQBS}(\mathbb{B}, 2^\mathbb{B}) \\
\llbracket \text{real} \rrbracket &= \text{MStoQBS}(\mathbb{R}, \mathbb{B}) \\
\llbracket t \times t' \rrbracket &= \llbracket t \rrbracket \times_q \llbracket t' \rrbracket \\
\llbracket t \to t' \rrbracket &= \llbracket t \rrbracket \to_q \text{Monad}(\llbracket t' \rrbracket) \\
\llbracket x_1:t_1, \ldots, x_n:t_n \rrbracket &= \llbracket t_1 \rrbracket \times_q \ldots \times_q \llbracket t_n \rrbracket
\end{align*}
\]
Terms mean morphisms almost

\[ \Gamma \vdash e : t \] is a morphism \( \Gamma \rightarrow_{q} \text{Monad}[t] \).
Probability measure on Quasi-Borel space
A **probability measure** on a QBS \((X, M)\) is a pair 
\((\alpha, \mu)\) of \(\alpha \in M\) and a prob. measure \(\mu\) on \((\mathbb{R}, \mathcal{B})\).
A probability measure on a QBS \((X, M)\) is a pair \((\alpha, \mu)\) of \(\alpha \in M\) and a prob. measure \(\mu\) on \((\mathbb{R}, \mathcal{B})\). random seed generator
A probability measure on a QBS \((X,M)\) is a pair \((\alpha, \mu)\) of \(\alpha \in M\) and a prob. measure \(\mu\) on \((\mathbb{R}, \mathcal{B})\).

seed convertor
A probability measure on a QBS \((X,M)\) is a pair \((\alpha,\mu)\) of \(\alpha \in M\) and a prob. measure \(\mu\) on \((\mathbb{R},\mathcal{B})\).
A probability measure on a QBS \((X,M)\) is a pair \((\alpha, \mu)\) of \(\alpha \in M\) and a prob. measure \(\mu\) on \((\mathbb{R}, \mathcal{B})\).

E.g.

\[
(X,M) = \text{MStoQBS}(\mathcal{B}, 2^\mathcal{B})
\]

\[
\mu = \text{uniform}(0,1], \quad \alpha(r) = \text{if } (r < 0.5) \text{ true false}
\]
A probability measure on a QBS \((X,M)\) is a pair \((\alpha,\mu)\) of \(\alpha \in M\) and a prob. measure \(\mu\) on \((\mathbb{R},\mathcal{B})\).

E.g.

\[
(X,M) = \text{MStoQBS}(\mathcal{B}, 2^\mathcal{B})
\]

\(\mu = \text{uniform}(0,1]\), \(\alpha(r) = \text{if } (r < 0.5) \text{ true false}\)

\(\mu' = \text{uniform}(0,2]\)/2, \(\alpha'(r) = \text{if } (r < 1) \text{ true false}\)
A probability measure on a QBS $(X,M)$ is a pair $(\alpha, \mu)$ of $\alpha \in M$ and a prob. measure $\mu$ on $(\mathbb{R}, \mathcal{B})$.

Quotient prob. measures by the smallest $\sim$ s.t.

$$(\alpha, \mu) \sim (\beta, \nu)$$

if $\alpha \circ f = \beta$ and $\nu \circ f^{-1} = \mu$ for some $f: \mathbb{R} \to_{m} \mathbb{R}$.

$[\alpha, \mu]$ - equivalence class.
QBS of prob. measures

\[ \text{Prob}(X,\mathcal{M}) = (Y,N) \]

\( Y = \{ [\alpha,\mu] \mid (\alpha,\mu) \text{ is a prob. meas. on } (X,\mathcal{M}) \} \).

\( N = \{ \lambda r.[\alpha,k(r)] \mid \alpha \in \mathcal{M} \text{ and } k : \mathbb{R} \times \mathcal{B} \rightarrow [0,1] \text{ is a prob. kernel} \} \).

[Lem] Prob is a strong monad.
Completing the definitions

\[ [[t \rightarrow t']] = [[t]] \rightarrow_q \text{Prob}([[t']]) \]

\[ [[\Gamma \vdash e : t]] \text{ is a morphism } [[\Gamma]] \rightarrow_q \text{Prob}[[t]] \]
Rest of this tutorial

1. Baby measure theory.
   PL with cont. distribution.

2. Quasi-Borel space (QBS).
   PL with cont. distr. & HO fns.

3. SFinKer monad on QBS.
   PL with cont. distr., HO fns & conditioning.
Conditioning and SFinKer monad on QBS
\([\Gamma \vdash e : t]\) is a morphism \([\Gamma] \rightarrow_q \text{Monad}[t]\)
\( \llbracket \Gamma \vdash e : t \rrbracket \) is a morphism \( \llbracket \Gamma \rrbracket \rightarrow \_q \text{Monad}[t] \)

Bayes's rule:

\[
p(o \mid h) \times p(h) = p(h \mid o) \times p(o)
\]
\([\Gamma \vdash e : t] \) is a morphism \(\Gamma \rightarrow_q \text{Monad}[t]\)

Bayes’s rule:

\[
p(o \mid h) \times p(h) = p(h \mid o) \times p(o)
\]

1. \(\text{Monad}(\_ ) = \text{Prob}(\_ ). \)
\([\Gamma \vdash e : t] \) is a morphism \([\Gamma] \rightarrow q \text{Monad}[t]\)

Bayes’s rule:

\[
p(o \mid h) \times p(h) = p(h \mid o) \times p(o)
\]

1. \text{Monad}(\_)= \text{Prob}(\_). Sometimes undefined.
\[ [\Gamma \vdash e : t] \text{ is a morphism } [\Gamma] \rightarrow_q \text{Monad}[t] \]

Bayes’s rule:

\[ p(o \mid h) \times p(h) = p(h \mid o) \times p(o) \]

1. Monad(\_\_) = Prob(\_\_). Sometimes undefined.

2. Monad(\_\_) = Prob([0, \infty) \times_q \_\_).
⟦Γ ⊢ e : t⟧ is a morphism ⟦Γ⟧ → Monad⟦t⟧

Bayes’s rule:

\[ p(o \mid h) \times p(h) = p(h \mid o) \times p(o) \]

1. Monad(_: ) = Prob(_). Sometimes undefined.

2. Monad(_: ) = Prob([0,∞) x q _). Failed eqs.
Failed conjugate-prior equation from statistics

\[
\begin{aligned}
&\text{let } x = \text{sample}(\beta(1, 1)) \text{ in } \\
&\text{observe}(\text{flip}(x), \text{true}); \\
&x \\
\end{aligned}
\neq

\begin{aligned}
&\text{observe}(\text{flip}(0.5), \text{true}); \\
&\text{sample}(\beta(2, 1))
\end{aligned}\]
\[[\Gamma \vdash e : t]\] is a morphism \[\Gamma \rightarrow_q \text{Monad}[t]\]

Bayes’s rule:

\[p(o \mid h) \times p(h) = p(h \mid o) \times p(o)\]

1. \(\text{Monad}(\_ ) = \text{Prob}(\_ )\). Sometimes undefined.

2. \(\text{Monad}(\_ ) = \text{Prob}([0, \infty) \times_q \_ )\). Failed eqs.
$\Gamma \vdash e : t$ is a morphism $\Gamma \rightarrow_q \text{Monad}[t]$

Bayes’s rule:

$$p(o \mid h) \times p(h) = p(h \mid o) \times p(o) = p(o, h)$$

1. Monad(_) = Prob(_). Sometimes undefined.

2. Monad(_) = Prob([0, \infty) \times_q _)\). Failed eqs.

3. Monad(_) = Meas(_).
\([\Gamma \vdash e : t] \) is a morphism \([\Gamma] \to \text{Monad}[t] \)

Bayes’s rule:

\[
p(o \mid h) \times p(h) = p(h \mid o) \times p(o) = p(o, h)
\]

1. \(\text{Monad}(\_\_) = \text{Prob}(\_\_).\) Sometimes undefined.

2. \(\text{Monad}(\_\_) = \text{Prob}([0, \infty) \times q \_\_).\) Failed eqs.

3. \(\text{Monad}(\_\_) = \text{Meas}(\_\_).\) No commutativity.
Failed commutativity

\[
\begin{align*}
\left[ \begin{array}{c}
\text{let } x = e \ \text{in} \\
\text{let } y = e' \ \text{in} \\
\text{e''}
\end{array} \right] \neq \left[ \begin{array}{c}
\text{let } y = e' \ \text{in} \\
\text{let } x = e \ \text{in} \\
\text{e''}
\end{array} \right]
\end{align*}
\]

if x doesn’t occur in e’ and y doesn’t occur in e
⟦Γ ⊢ e : t⟧ is a morphism ⟦Γ⟧ →_q Monad⟦t⟧

Bayes’s rule:

\[ p(o \mid h) \times p(h) = p(h \mid o) \times p(o) = p(o, h) \]

1. Monad(_) = Prob(_). Sometimes undefined.

2. Monad(_) = Prob([0, ∞) x_q _). Failed eqs.

3. Monad(_) = Meas(_). No commutativity.
$\Gamma \vdash e : t \]$ is a morphism $\Gamma \to q \text{Monad}[t]$

Bayes’s rule:

$$p(o \mid h) \times p(h) = p(h \mid o) \times p(o) = p(o, h)$$


2. Monad(_$) = Prob([0, \infty) \times_q _)$. Failed eqs.


4. Monad(_$) = SFinKer(_$).
QBS of prob. measures

\[ \text{Prob}(X,M) = (Y,N) \]

\[ Y = \{ [\alpha, \mu] \mid \alpha \in M, \mu \text{ prob. meas. on } (\mathbb{R}, \mathcal{B}) \}. \]

\[ N = \{ \lambda r.[\alpha, k(r)] \mid \alpha \in M, k : \mathbb{R} \times \mathcal{B} \to [0,1] \text{ prob. kernel} \}. \]
QBS of prob. measures
s-finite kernels

\[ \text{Prob}(X,M) = (Y,N) \]

\[ Y = \{ [\alpha, \mu] \mid \alpha \in M, \mu \text{ prob. meas. on } (\mathbb{R}, \mathscr{B}) \} \]

\[ N = \{ \lambda r.[\alpha, k(r)] \mid \alpha \in M, \quad k : \mathbb{R} \times \mathscr{B} \rightarrow [0,1] \text{ prob. kernel} \} \]
QBS of prob. measures

s-finite kernels

\[ \text{Prob}(X,M) = (Y,N) \]

\[ Y = \{ [\alpha, \mu] \mid \alpha \in M, \mu \text{ prob. meas. on } (\mathbb{R}, \mathcal{B}) \} \]

\[ N = \{ \lambda r.[\alpha, k(r)] \mid \alpha \in M, k : \mathbb{R} \times \mathcal{B} \to [0,1] \text{ prob. kernel} \} \]

\( \mu \text{ finite if like prob. measure but just } \mu(\mathbb{R}) < \infty. \)

\( \mu \text{ s-finite if countable sum of finite measures.} \)
QBS of prob. measures

\[ \text{Prob}(X, M) = (Y, N) \]

\[ Y = \{ [\alpha, \mu] \mid \alpha \in M, \mu \text{ prob. meas. on } (\mathbb{R}, \mathcal{B}) \} \]

\[ N = \{ \lambda r.[\alpha, k(r)] \mid \alpha \in M, k : \mathbb{R} \times \mathcal{B} \to [0, 1] \text{ prob. kernel} \} \]

\[ \mu \text{ finite if like prob. measure but just } \mu(\mathbb{R}) < \infty. \]

\[ \mu \text{ s-finite if countable sum of finite measures.} \]
[Th] SFinKer can be used to define the semantics of prob. PL with conditioning. (i.e., strong monad.)

[Th] It validates commutativity of programs and prog. eqs from statistics.
\[
\begin{align*}
\llbracket \text{bool} \rrbracket &= \text{MStoQBS}(\mathbb{B}, 2^\mathbb{B}) \\
\llbracket \text{real} \rrbracket &= \text{MStoQBS}(\mathbb{R}, \mathbb{B}) \\
\llbracket t \times t' \rrbracket &= \llbracket t \rrbracket \times_q \llbracket t' \rrbracket \\
\llbracket t \rightarrow t' \rrbracket &= \llbracket [t] \rightarrow_q \text{SFinKer}(\llbracket t' \rrbracket) \rrbracket \\
\llbracket x_1: t_1, \ldots, x_n: t_n \rrbracket &= \llbracket t_1 \rrbracket \times_q \ldots \times_q \llbracket t_n \rrbracket \\
\llbracket \Gamma \vdash e : t \rrbracket \text{ is a morphism } \llbracket \Gamma \rrbracket \rightarrow_q \text{SFinKer}[t]
\end{align*}
\]
\[[\text{bool}]\] = MStoQBS(\(\mathbb{B}, 2^\mathbb{B}\))

\[[\text{real}]\] = MStoQBS(\(\mathbb{R}, \mathbb{B}\))

\[[t \times t']\] = \([t]\) x_q \([t']\)

\[[t \rightarrow t']\] = \([[[t]] \rightarrow_q SFinKer([t'])\])

\[[x_1:t_1, \ldots, x_n:t_n]\] = \([t_1]\) x_q \ldots x_q \([t_n]\)

\([\Gamma \vdash e : t]\) is a morphism \([\Gamma] \rightarrow_q SFinKer[t]\)
\[[\text{bool}]\] = MStoQBS(\(\mathbb{B}, 2^\mathbb{B}\))

\[[\text{real}]\] = MStoQBS(\(\mathbb{R}, \mathbb{B}\))

\[[t \times t']\] = \([t]\) \(x_q\) \([t']\)

\[[t \rightarrow t']\] = \([[t] \rightarrow_q SFinKer([[t']])\])

\[[x_1: t_1, \ldots, x_n: t_n]\] = \([t_1]\) \(x_q\) \(\ldots\) \(x_q\) \([t_n]\)

\[[\Gamma \vdash e : t]\] is a morphism \([[\Gamma] \rightarrow_q SFinKer([t])]\)
\[ [\text{bool}] = \text{MStoQBS}(\mathbb{B}, 2^\mathbb{B}) \]
\[ [\text{real}] = \text{MStoQBS}(\mathbb{R}, \mathbb{B}) \]
\[ [t \times t'] = [t] \times_q [t'] \]
\[ [t \to t'] = [[t] \to_q \text{SFinKer}([t'])] \]
\[ [x_1 : t_1, \ldots, x_n : t_n] = [t_1] \times_q \ldots \times_q [t_n] \]
\[ [\Gamma \vdash e : t] \text{ is a morphism } [[\Gamma] \to_q \text{SFinKer}[t]] \]
\[[\text{bool}]\] = \text{MStoQBS}(\mathbb{B}, 2^{\mathbb{B}}) \\
\[[\text{real}]\] = \text{MStoQBS}(\mathbb{R}, \mathbb{B}) \\
\[[t \times t']\] = \[[t]\] x_q [[t']]
\[[t \rightarrow t']\] = \[[[t] \rightarrow_q \text{SFinKer}([[t']])\]

\[[x_1:t_1, \ldots, x_n:t_n]\] = \[[t_1]\] x_q \ldots x_q [[t_n]]

[[\Gamma \vdash e : t]] is a morphism \[[\Gamma] \rightarrow_q \text{SFinKer}[[t]]\]
References

1. A convenient category for higher-order probability theory. Heunen et al. LICS’17.
2. Commutative semantics for probabilistic programs. Staton. ESOP’17.
References

1. A convenient category for higher-order probability theory. Heunen et a. LICS’17.

2. Commutative semantics for probabilistic programs. Staton. ESOP’17.