

1 Intro

A walkthrough of [1] and [2], mainly focusing on the presentation in [2]. We first briefly review measures and kernels because they show up in the semantics.

2 Measures

A **σ -algebra** Σ_X on a set X is a collection of subsets of X that contains \emptyset and is closed under complements and countable unions. A **measurable space** is a pair (X, Σ_X) of a set and a σ -algebra on it. The elements of Σ_X , which are themselves sets of elements in X , are called **measurable sets**. For example, the Borel sets are the smallest σ -algebra on \mathbb{R} that contains the intervals. This is the usual σ -algebra for \mathbb{R} . For any countable set, you can always start off with the individual elements of the set: satisfying complements and unions this gives you the powerset σ -algebra.

A **measure** on a measurable space (X, Σ_X) is a function $\mu : \Sigma_X \rightarrow [0, \infty]$ into the set $[0, \infty]$ of extended non-negative reals that takes countable disjoint unions to sums, i.e. $\mu(\emptyset) = 0$ and $\mu(\bigcup_{n \in \mathbb{N}} U_n) = \sum_{n \in \mathbb{N}} \mu(U_n)$ for any \mathbb{N} -indexed sequence of disjoint measurable sets U_n . A **probability measure** is a measure μ such that $\mu(X) = 1$. For example, the Lebesgue measure λ on \mathbb{R} is generated by $\lambda((a, b)) = b - a$. For any $x \in X$, the Dirac measure $\delta_x(U) = 1$ if $x \in U$ and 0 otherwise.

3 Kernels

A **kernel** k from X to Y , notated $k : X \rightsquigarrow Y$, is a function $k : X \times \Sigma_Y \rightarrow [0, \infty]$ such that

- for fixed x , $k(x, -) : \Sigma_Y \rightarrow [0, \infty]$ is a measure
- for fixed dy , $k(-, dy) : X \rightarrow [0, \infty]$ is a measurable function.

An intuitive example of kernels at work is conditional probability. Take $k(x, -)$ to be a probability measure $\mu_Y : \Sigma_Y \rightarrow [0, 1]$ on Y given a particular value $X = x$: $\sum_{dy \in \Sigma_Y} k(x, dy) = 1$. Note that $\sum_{x \in X} k(x, dy) \neq 1$ in general.

Let X, Y, Z be measurable spaces and let $k^1 : X \times Y \rightsquigarrow Z$ and $k^2 : X \rightsquigarrow Y$ be s-finite kernels (**TODO**: explain s-finiteness). Then we can define the composition $(k^1 \star k^2) : X \rightsquigarrow Z$ as

$$(k^1 \star k^2)(x, U) = \int_Y k^2(x, dy) k^1(x, y, U)$$

For intuition, consider $k(x, -) : \Sigma_Y \rightarrow [0, 1]$ to be the probability measure that tells you how likely it is to start off at location x and end up in the interval dy . Then the composition $(k^1 \star k^2)(x, dz) = \int_Y k^2(x, dy) k^1(x, y, dz)$ can be taken to represent the probability of starting off at x and ending up in the interval dz , where we take a step in-between to land on y , but average out across all intervals dy that we can land in when jumping from x .

4 Types and Semantics

The two papers [1,2] don't actually define the set of terms, but they are implied to be the typical ones in a language like Haskell. The 2018 POPL paper on semantics for higher-order languages [3] does formally introduce a kind and type system and a set of terms.

$$\mathbb{A}, \mathbb{B} ::= \mathbb{R} | P(\mathbb{A}) | 1 | \mathbb{A} \times \mathbb{B} | \sum_i \mathbb{A}_{i \in I}$$

where I is countable and non-empty. As we will see, types are interpreted as measurable spaces $[[\mathbb{A}]]$. We have sum and product types. We have a type that denotes the Reals. We have a type that denotes the set of probability measures over a space denoted by \mathbb{A} . We have the type 1 that denotes a singleton set. For booleans, we can just take $(1+1)$ and $P(1+1)$ is the type for distributions over booleans. For the natural numbers, we can use $\sum_{i \in \mathbb{N}} 1$.

Typing judgements: $\Gamma \vdash_d$ for deterministic judgements and $\Gamma \vdash_p$ for probabilistic judgements. Having the two typing judgements is mainly for notational clarity: it helps us to define interpretation differently for deterministic and probabilistic terms.

4.1 Typing Judgements and Interpretations for Deterministic Terms

Deterministic terms $\Gamma \vdash_d t : \mathbb{A}$ are interpreted as measurable functions $[[t]] : [[\Gamma]] \rightarrow [[\mathbb{A}]]$, where $[[t]](\gamma) = x$ is an element of the underlying set $[[\mathbb{A}]]$ rather than a measurable set in $\Sigma_{[[\mathbb{A}]]}$.

- $[[x]](\gamma :: d :: \gamma') = d$
- $[[(i, t)]](\gamma) = (i, [[t]](\gamma))$
- $[[\text{case } t \text{ of } \{(i, x) \rightarrow u_i\}_{i \in I}]](\gamma) = [[u_i]](\gamma, d)$ if $[[t]](\gamma) = (i, d)$
- $[[()]](\gamma) = ()$
- $[[(t_0, t_1)]](\gamma) = ([[t_0]](\gamma), [[t_1]](\gamma))$
- $[[\pi_j(t)]](\gamma) = d_j$ if $[[t]](\gamma) = (d_0, d_1)$

where the **case** expression above has a deterministic continuation u_i . We will come back to the probabilistic continuation case. Now the semantics for sequencing.

4.2 Typing Judgements and Interpretations for Probabilistic Terms

Probabilistic terms $\Gamma \vdash_p t : \mathbb{A}$ are interpreted as s-finite kernels $[[t]] : [[\Gamma]] \rightsquigarrow [[\mathbb{A}]]$.

- **return**(t)

$$\frac{\Gamma \vdash_d t : \mathbb{A}}{\Gamma \vdash_p \text{return}(t) : \mathbb{A}} \quad [[\text{return}(t)]](\gamma, da) = \delta_{[[t]](\gamma)}(da)$$

this corresponds to a Dirac Delta measure sitting at a point $[[t]]_\gamma$.

- **let $x = t$ in u**

$$\frac{\Gamma \vdash_p t : \mathbb{A} \quad \Gamma, x : \mathbb{A} \vdash_p u : \mathbb{B}}{\Gamma \vdash_p \text{let } x = t \text{ in } u : \mathbb{B}} \quad [[\text{let } x = t \text{ in } u]](\gamma, db) = \int_{x, dx \in [[\mathbb{A}]}} [[u]](\gamma, x, db) [[t]](\gamma, dx)$$

This one takes careful reading. Consider the kernel composition definition above. Take k_1 to be $[[u]]$ and take k_2 to be $[[t]]$. Then $(k^1 \star k^2)(\gamma, db) = ([[u]] \star [[t]])(\gamma, db)$

- **probabilistic case**

$$\frac{\Gamma \vdash_d t : \sum_{i \in I} \mathbb{A}_i \quad (\Gamma, x : \mathbb{A}_i \vdash_p u_i : \mathbb{B})_{i \in I}}{\Gamma \vdash_p (\text{case } t \text{ of } \{(i, x) \rightarrow u_i\}_{i \in I}) : \mathbb{B}}$$

$$[[\text{case } t \text{ of } \{(i, x) \rightarrow u_i\}_{i \in I}]](\gamma, db) = [[u_i]](\gamma :: d, db) \text{ if } [[t]](\gamma) = (i, d)$$

- **sample(t)**

$$\frac{\Gamma \vdash_d t : P(\mathbb{A})}{\Gamma \vdash_p \text{sample}(t) : \mathbb{A}} \quad [[\text{sample}(t)]](\gamma, da) = ([[t]](\gamma))(da)$$

- **score(t)**

$$\frac{\Gamma \vdash_d t : \mathbb{R}}{\Gamma \vdash_p \text{score}(t) : 1}$$

$$[[\text{score}(t)]](\gamma, du) = \begin{cases} \text{abs}([t](\gamma)), & \text{if } du = \{()\} \\ 0, & \text{if } du = \emptyset \end{cases}$$

4.3 Typing Judgements and Interpretation for Normalize

$$\frac{\Gamma \vdash_p t : \mathbb{A}}{\Gamma \vdash_d \text{normalize}(t) : \mathbb{R} \times P(\mathbb{A}) + 1 + 1}$$

To give it a semantics, we must find the normalizing constant to divide by. Consider $\Gamma \vdash_p t : \mathbb{A}$ and let $\text{evidence}_t = [[t]]_{\gamma, [[\mathbb{A}]}}$. Then:

$$[[\text{normalize}(t)]](\gamma) = \begin{cases} (0, (\text{evidence}_t, \frac{[[t]](\gamma, (-))}{\text{evidence}_t})), & \text{if } \text{evidence}_t \in (0, \infty) \\ (1, ()), & \text{if } \text{evidence}_t = 0 \\ (2, ()), & \text{if } \text{evidence}_t = \infty \end{cases}$$

5 Example Programs

...

6 Commutativity

...

7 Higher Order

...

8 Citation

1. Staton et al. Semantics for probabilistic programming: higher-order functions, continuous distributions, and soft constraints. LICS 2016. <https://arxiv.org/pdf/1601.04943.pdf>
2. Staton. Commutative semantics for probabilistic programming. ESOP 2017. ‘<http://www.cs.ox.ac.uk/people/samuel.staton/papers/esop2017.pdf>
3. Scibior et al. Denotational validation of Bayesian inference. POPL 2018. <https://arxiv.org/pdf/1711.03219.pdf>