EQUATIONAL REASONING FOR PROBABILISTIC PROGRAMMING

CHUNG-CHIEH SHAN, INDIANA UNIVERSITY

1. The fields

	Programming	Probability
Theoretical	Induction	Integral
Practical	Interpreter	Inference

2. The tasks

Whenever we're unsure about something, represent our uncertain knowledge as a distribution.

2.1. **The table game.** (Eddy 2004)

$$
\begin{aligned}
&\text{casino} \;:\; \mathbb{M}\,\text{Bool}\\
&\text{casino} \stackrel{\triangle}{=} \text{do}\;\{p \leftsquigarrow \text{uniform}\;0\;1;\\
&\qquad\qquad a_1 \leftsquigarrow \text{binomial}\;8\;p;\\
&\qquad\qquad () \leftsquigarrow \text{guard}\;(a_1 = 5);\\
&\qquad\qquad a_2 \leftsquigarrow \text{binomial}\;3\;p;\\
&\qquad\qquad \text{return}\;(a_2 \geq 1)\}
\end{aligned}
$$

Date: January 8, 2018.

2.2. **Inferring behavior from text-message data.** (Davidson-Pilon 2016)

$$
\begin{aligned}
&\text{texting} \;:\; \mathbb{M}\,(\overline{\mathbb{R}}_+ \times \overline{\mathbb{R}}_+)\\
&\text{texting} \stackrel{\triangle}{=} \text{do}\;\{r_1 \leftsquigarrow \text{exponential}\;37;\\
&\qquad\qquad r_2 \leftsquigarrow \text{exponential}\;42;\\
&\qquad\qquad t_0 \leftsquigarrow \text{counting}\;1\;70;\\
&\qquad\qquad \vec{c} \leftsquigarrow \text{mapM}\;\big(\lambda t.\,\text{poisson}\;(\text{if}\;t < t_0\;\text{then}\;r_1\;\text{else}\;r_2)\big)\\
&\qquad\qquad\qquad\qquad [1\,..\,70];\\
&\qquad\qquad () \leftsquigarrow \text{guard}\;(\vec{c} = [13, 24, ..]);\\
&\qquad\qquad \text{return}\;(r_1, r_2)\}
\end{aligned}
$$

2.3. **Observing a noisy draw from a normal distribution.**

$$
\begin{aligned}
&\text{helloWrong} &&: \mathbb{R} \to \mathbb{M}\,\mathbb{R}\\
&\text{helloWrong}\;y_0 &&\stackrel{\triangle}{=} \text{do}\;\{x \leftsquigarrow \text{normal}\;0\;1;\\
& &&\qquad y \leftsquigarrow \text{normal}\;x\;1;\\
& &&\qquad () \leftsquigarrow \text{guard}\;(y = y_0);\;\texttt{-- WRONG!}\\
& &&\qquad z \leftsquigarrow \text{normal}\;x\;1;\\
& &&\qquad \text{return}\;z\}\\
&\text{helloRight} &&: \mathbb{R} \to \mathbb{M}\,\mathbb{R}\\
&\text{helloRight}\;y_0 &&\stackrel{\triangle}{=} \text{do}\;\{x \leftsquigarrow \text{normal}\;0\;1;\\
& &&\qquad () \leftsquigarrow \text{factor}\;\frac{e^{-(y_0-x)^2/2}}{\sqrt{2\cdot\pi}};\\
& &&\qquad z \leftsquigarrow \text{normal}\;x\;1;\\
& &&\qquad \text{return}\;z\}\\
&\text{helloJoint} &&: \mathbb{M}\,(\mathbb{R} \times \mathbb{R})\\
&\text{helloJoint} &&\stackrel{\triangle}{=} \text{do}\;\{x \leftsquigarrow \text{normal}\;0\;1;\\
& &&\qquad y \leftsquigarrow \text{normal}\;x\;1;\\
& &&\qquad z \leftsquigarrow \text{normal}\;x\;1;\\
& &&\qquad \text{return}\;(y, z)\}\;\texttt{-- Ready to disintegrate}
\end{aligned}
$$

3. The equations

3.1. Nondeterminism and weights.

$$\begin{aligned}\text{binomial } 2\ p &= \big(p \odot p \odot \text{return } 2\big) \oplus \\ &\quad \big(p \odot (1-p) \odot \text{return } 1\big) \oplus \\ &\quad \big((1-p) \odot p \odot \text{return } 1\big) \oplus \\ &\quad \big((1-p) \odot (1-p) \odot \text{return } 0\big) \end{aligned} \tag{1}$$

$$\begin{aligned} &= \big(p^2 \odot \text{return } 2\big) \oplus \\ &\quad \big(2p(1-p) \odot \text{return } 1\big) \oplus \\ &\quad \big((1-p)^2 \odot \text{return } 0\big) \end{aligned} \tag{2}$$

3.2. From rejection sampling to importance sampling.

(MacKay 1998)

$$\begin{aligned}\text{casino} = \text{do } \{&p \leftsquigarrow \text{uniform } 0\ 1; \\ &a_1 \leftsquigarrow \text{binomial } 8\ p; \\ &() \leftsquigarrow \text{guard } (a_1 = 5); \\ &((1-(1-p)^3) \odot \text{return True}) \oplus \\ &\qquad ((1-p)^3 \odot \text{return False})\} \end{aligned} \tag{3}$$

$$\begin{aligned} = \text{do } \{&p \leftsquigarrow \text{uniform } 0\ 1; \\ &() \leftsquigarrow \text{factor } (56 \cdot p^5 \cdot (1-p)^3); \\ &((1-(1-p)^3) \odot \text{return True}) \oplus \\ &\qquad ((1-p)^3 \odot \text{return False})\} \end{aligned} \tag{4}$$

In general, if $m = r \mathrel{=\!\odot} n$, then we say that the function r is a *density* or *Radon-Nikodym derivative* of m with respect to n. If we know how to sample n, then r tells us how to importance-sample m using the proposal distribution n.

3.3. Density facts.

If r is a density of m with respect to n, then $r \circ f^{-1}$ is a density of fmap $f\ m$ with respect to fmap $f\ n$ whenever f is invertible.

If r is a density of m with respect to n, then recip $\circ\, r$ is a density of n with respect to m. Here recip is the reciprocal function $\lambda x.\, 1/x$, and a side condition is that the reciprocal must be defined almost everywhere:

$$\text{do } \{x \leftsquigarrow n;\ \text{guard } \neg(0 < r\ x < \infty)\} = \text{fail} \tag{5}$$

3.4. Conjugate prior and density recognition.

$$\begin{aligned}\text{casino} = (1/9) \odot \text{do } \{&p \leftsquigarrow \text{beta } 6\ 4; \\ &((1-(1-p)^3) \odot \text{return True}) \oplus \\ &\qquad ((1-p)^3 \odot \text{return False})\} \end{aligned} \tag{6}$$

$$\begin{aligned}\text{helloRight } y_0 = \frac{e^{-y_0^2/4}}{\sqrt{4\cdot\pi}} \odot \text{do } \{&x \leftsquigarrow \text{normal } (y_0/2)\ (1/\sqrt{2}); \\ &z \leftsquigarrow \text{normal } x\ 1; \\ &\text{return } z\} \end{aligned} \tag{7}$$

3.5. Variable elimination and integration.

$$\begin{aligned}\text{casino} = (1/9) \odot \big(&((10/11) \odot \text{return True}) \oplus \\ &((1/11) \odot \text{return False})\big) \end{aligned} \tag{8}$$

$$\text{helloRight } y_0 = \frac{e^{-y_0^2/4}}{\sqrt{4\cdot\pi}} \odot \text{normal } (y_0/2)\ \sqrt{3/2} \tag{9}$$

3.6. From density to disintegration.

(Shan and Ramsey 2017)

$$\text{helloJoint} = \text{lebesgue } (-\infty)\ \infty \mathrel{\otimes\!=} \text{helloRight} \tag{10}$$

Generalize helloRight to a *Kalman filter*, such as a function

$$f : \text{State} \to \mathbb{R} \to \text{State}$$

(where State $= \mathbb{R}_+ \times \mathbb{R} \times \mathbb{R}_+$) satisfying

$$\begin{aligned}\text{interpret } (f\ wcd\ y_0) = \text{do } \{&x \leftsquigarrow \text{interpret } wcd; \\ &\frac{e^{-(y_0-x)^2/2}}{\sqrt{2\cdot\pi}} \odot \text{normal } x\ 1\} \end{aligned} \tag{11}$$

(where interpret $(w, c, d) = w \odot \text{normal } c\ d$).

3.7. **Markov chain Monte Carlo.** (MacKay 1998; Tierney 1998; McElreath 2017)

Suppose we want samples from a given *target distribution*

$$p : \mathbb{M}\,\alpha$$

but it is inefficient or weighted as a sampler. To use Markov chain Monte Carlo, we seek a *transition kernel*

$$k : \alpha \to \mathbb{M}\,\alpha,$$

and iterate it to perform a random walk in the state space α. Our k should return a probability measure that is efficient and unweighted as a sampler. Moreover, it should satisfy *detailed balance*:

$$p \otimes\!\!= k = k =\!\!\otimes\, p \tag{12}$$

(Intuition: if $p \otimes\!\!= k = k =\!\!\otimes\, p$ then $p \gg\!= k = p$.)

Metropolis-Hastings is a way to construct k from a *proposal distribution*

$$q : \alpha \to \mathbb{M}\,\alpha.$$

Like the k we want, the q we provide should return a probability measure that is efficient and unweighted as a sampler, but q need not satisfy detailed balance (in other words, it is fine if $p \otimes\!\!= q \neq q =\!\!\otimes\, p$).

To use Metropolis-Hastings given q, first find a density r of $p \otimes\!\!= q$ with respect to $q =\!\!\otimes\, p$. That is, find

$$r : (\alpha \times \alpha) \to \overline{\mathbb{R}}_+$$

such that

$$p \otimes\!\!= q = r =\!\!\odot\, (q =\!\!\otimes\, p). \tag{13}$$

Then let

$$\alpha(x, y) = \min\{1, r(x, y)\} \tag{14}$$

$$\begin{aligned} k\ \text{old} = \text{do}\ \{&\text{new} \leftsquigarrow q\ \text{old};\\ &\text{accept} \leftsquigarrow \text{bernoulli}\ (\alpha(\text{new}, \text{old}));\\ &\text{return (if accept then new else old)}\} \end{aligned} \tag{15}$$

4. The interpretations: What are we equating?

4.1. **Denotational semantics.** (Culpepper and Cobb 2017; Staton 2017; Heunen et al. 2017; Ścibior et al. 2018)

Good for declaring what we want to compute. Equality is a congruence.

Measures are equivalent to integrators. Easy to understand as continuation-passing style.

4.2. **Operational semantics: samplers.** Good for implementing algorithms. But what kind of samplers?

4.2.1. *Randomized samplers (“Monte Carlo methods”) vs deterministic code.*

- Randomized samplers, such as Eddy’s (2004) samplers.
- Deterministic code, such as Eddy’s (2004) exact formula.

4.2.2. *Weighted vs unweighted results.*

- Weighted results, as from importance sampling, are superposed over time.
  Example use: histograms and other forms of expectation estimation.
- Unweighted results, as from rejection sampling, are used committally.
  Example use: deciding how to drive or where to visit next in a graph.
- Converting unweighted result stream to weighted is trivial.
- Converting weighted result stream to unweighted requires bound on weight.

4.2.3. *Efficient vs inefficient algorithms.*

- Sure we want samples fast, but slower samples can be more accurate.

5. The language

return		$: \alpha \to \mathbb{M}\,\alpha$		
$\gg=$	(>>=)	$: \mathbb{M}\,\alpha \to (\alpha \to \mathbb{M}\,\beta) \to \mathbb{M}\,\beta$	$m \gg= k$	$= \text{do}\ \{x \leftsquigarrow m;\ y \leftsquigarrow k\ x;\ \text{return}\ y\}$
$\gg$	(>>)	$: \mathbb{M}\,\alpha \to \mathbb{M}\,\beta \to \mathbb{M}\,\beta$	$m \gg n$	$\triangleq \text{do}\ \{\_ \leftsquigarrow m;\ y \leftsquigarrow n;\ \text{return}\ y\}$
$=\gg$	(=>>)	$: (\beta \to \mathbb{M}\,\alpha) \to \mathbb{M}\,\beta \to \mathbb{M}\,\beta$	$k =\gg n$	$\triangleq \text{do}\ \{y \leftsquigarrow n;\ \_ \leftsquigarrow k\ y;\ \text{return}\ y\}$
$\otimes=$	(<>=)	$: \mathbb{M}\,\alpha \to (\alpha \to \mathbb{M}\,\beta) \to \mathbb{M}\,(\alpha \times \beta)$	$m \otimes= k$	$\triangleq \text{do}\ \{x \leftsquigarrow m;\ y \leftsquigarrow k\ x;\ \text{return}\ (x, y)\}$
$\otimes$	(<>)	$: \mathbb{M}\,\alpha \to \mathbb{M}\,\beta \to \mathbb{M}\,(\alpha \times \beta)$	$m \otimes n$	$\triangleq \text{do}\ \{x \leftsquigarrow m;\ y \leftsquigarrow n;\ \text{return}\ (x, y)\} = \text{do}\ \{y \leftsquigarrow n;\ x \leftsquigarrow m;\ \text{return}\ (x, y)\}$
$=\otimes$	(=<>)	$: (\beta \to \mathbb{M}\,\alpha) \to \mathbb{M}\,\beta \to \mathbb{M}\,(\alpha \times \beta)$	$k =\otimes n$	$\triangleq \text{do}\ \{y \leftsquigarrow n;\ x \leftsquigarrow k\ y;\ \text{return}\ (x, y)\}$
fmap		$: (\alpha \to \beta) \to \mathbb{M}\,\alpha \to \mathbb{M}\,\beta$	fmap $f\ m$	$\triangleq m \gg= (\text{return} \circ f)$
$\oplus$	(<+>)	$: \mathbb{M}\,\alpha \to \mathbb{M}\,\alpha \to \mathbb{M}\,\alpha$		
fail		$: \mathbb{M}\,\alpha$		
guard		$: \text{Bool} \to \mathbb{M}\,\text{Unit}$	guard b	$\triangleq$ if b then return () else fail
factor		$: \overline{\mathbb{R}}_+ \to \mathbb{M}\,\text{Unit}$		
$\odot$	(*>)	$: \overline{\mathbb{R}}_+ \to \mathbb{M}\,\beta \to \mathbb{M}\,\beta$	$p \odot n$	$\triangleq \text{factor}\ p \gg n$
$=\odot$	(=*>)	$: (\beta \to \overline{\mathbb{R}}_+) \to \mathbb{M}\,\beta \to \mathbb{M}\,\beta$	$r =\odot n$	$\triangleq (\text{factor} \circ r) =\gg n$
bernoulli		$: [0, 1] \to \mathbb{M}\,\mathbb{N}$	bernoulli p	$= \big(p \odot \text{return}\ 1\big) \oplus \big((1 - p) \odot \text{return}\ 0\big)$
binomial		$: \mathbb{N} \to [0, 1] \to \mathbb{M}\,\mathbb{N}$	binomial $i\ p$	$= \text{fmap sum}\ \big(\text{replicateM}\ i\ (\text{bernoulli}\ p)\big)$
counting		$: \overline{\mathbb{N}} \to \overline{\mathbb{N}} \to \mathbb{M}\,\mathbb{N}$		
geometric		$: [0, 1] \to \mathbb{M}\,\mathbb{N}$	geometric p	$= \big(\lambda i.\,(1 - p) \cdot p^i\big) =\odot \text{counting}\ 0\ \infty$
poisson		$: \mathbb{R}_+ \to \mathbb{M}\,\mathbb{N}$	poisson r	$: \big(\lambda i.\, r^i / e^r / i!\big) =\odot \text{counting}\ 0\ \infty$
lebesgue		$: \overline{\mathbb{R}} \to \overline{\mathbb{R}} \to \mathbb{M}\,\mathbb{R}$		
uniform		$: \mathbb{R} \to \mathbb{R} \to \mathbb{M}\,\mathbb{R}$	uniform $x\ y$	$= \big(1/(y - x)\big) \odot \text{lebesgue}\ x\ y$
beta		$: \mathbb{R}_+ \to \mathbb{R}_+ \to \mathbb{M}\,[0, 1]$	beta $a\ b$	$= \big(\lambda p.\, p^{a-1} \cdot (1 - p)^{b-1} / \mathrm{B}(a, b)\big) =\odot \text{lebesgue}\ 0\ 1$
exponential		$: \mathbb{R}_+ \to \mathbb{M}\,\mathbb{R}_+$	exponential l	$= \text{do}\ \{x \leftsquigarrow \text{lebesgue}\ 0\ \infty;\ e^{-x} \odot \text{return}\ (l \cdot x)\}$
normal		$: \mathbb{R} \to \mathbb{R}_+ \to \mathbb{M}\,\mathbb{R}$	normal $c\ d$	$= \text{do}\ \{x \leftsquigarrow \text{lebesgue}\ (-\infty)\ \infty;\ (e^{-x^2/2} / \sqrt{2 \cdot \pi}) \odot \text{return}\ (c + d \cdot x)\}$

6. The laws

6.1. $\gg=$ and return form a commutative monad.

$$\text{return}\ x \gg= k = k\ x \tag{16}$$
$$m \gg= \text{return} = m \tag{17}$$
$$(m \gg= k) \gg= l = m \gg= \lambda x.\,(k\ x \gg= l) \tag{18}$$
$$\text{do}\ \{x \leftsquigarrow m;\ y \leftsquigarrow n;\ k\ x\ y\} = \text{do}\ \{y \leftsquigarrow n;\ x \leftsquigarrow m;\ k\ x\ y\} \tag{19}$$

(Generalize $\otimes\!=$ to countable products?)

6.2. $\oplus$ and fail form a commutative monoid.

$$\text{fail} \oplus m = m \tag{20}$$
$$= m \oplus \text{fail} \tag{21}$$
$$(m \oplus n) \oplus o = m \oplus (n \oplus o) \tag{22}$$
$$m \oplus n = n \oplus m \tag{23}$$

(Generalize $\oplus$ to countable sums?)

6.3. $\oplus$ and fail distribute over $\gg=$.

$$(m \oplus n) \gg= k = (m \gg= k) \oplus (n \gg= k) \tag{24}$$
$$m \gg= \lambda x.\,(k\ x \oplus l\ x) = (m \gg= k) \oplus (m \gg= l) \tag{25}$$
$$\text{fail} \gg= k = \text{fail} \tag{26}$$
$$m \gg \text{fail} = \text{fail} \tag{27}$$

6.4. factor is an isomorphism between $\overline{\mathbb{R}}_+$ and $\mathbb{M}$ Unit.

$$\text{factor}\ p \gg \text{factor}\ q = \text{factor}\ (p \cdot q) \tag{28}$$
$$\text{factor}\ p \oplus \text{factor}\ q = \text{factor}\ (p + q) \tag{29}$$
$$\text{return}\ () = \text{factor}\ 1 \tag{30}$$
$$\text{fail} = \text{factor}\ 0 \tag{31}$$

(Treat $\overline{\mathbb{R}}_+$ as synonym for $\mathbb{M}$ Unit?)

6.5. Conjugate priors. (derived by algebra)

$$\left(\lambda p.\, p^{a'} \cdot (1-p)^{b'}\right) =\!\odot\ \text{beta}\ a\ b \quad = \quad \frac{\mathrm{B}(a+a', b+b')}{\mathrm{B}(a,b)} \odot \text{beta}\ (a+a')\ (b+b') \tag{32}$$
$$\left(\lambda x.\, \frac{e^{-(x-c')^2/d'^2/2}}{\sqrt{2\cdot\pi}\cdot d'}\right) =\!\odot\ \text{normal}\ c\ d = \frac{e^{-(c-c')^2/(d^2+d'^2)/2}}{\sqrt{2\cdot\pi\cdot(d^2+d'^2)}} \odot \text{normal}\ \frac{c\cdot d^{-2} + c'\cdot d'^{-2}}{d^{-2}+d'^{-2}}\ \frac{1}{\sqrt{d^{-2}+d'^{-2}}} \tag{33}$$

6.6. Probability measures. (derived by integral calculus)

$$\text{bernoulli}\ p \gg n = n \tag{34}$$
$$\text{geometric}\ p \gg n = n \tag{35}$$
$$\text{poisson}\ r \gg n = n \tag{36}$$
$$\text{uniform}\ x\ y \gg n = n \tag{37}$$
$$\text{beta}\ a\ b \gg n = n \tag{38}$$
$$\text{exponential}\ l \gg n = n \tag{39}$$
$$\text{normal}\ c\ d \gg n = n \tag{40}$$

6.7. Change of variables. (derived by integral calculus)

$$\text{fmap}\ (\lambda x.\,-\log x)\ (\text{uniform}\ 0\ 1) = \text{exponential}\ 1 \tag{41}$$
$$\text{fmap}\ (\lambda x.\, c + d\cdot x)\ (\text{lebesgue}\ a\ b) = (1/d) \odot \text{lebesgue}\ (c + d\cdot a)\ (c + d\cdot b) \tag{42}$$

References

Culpepper, Ryan, and Andrew Cobb. 2017. Contextual equivalence for probabilistic programs with continuous random variables and scoring. In *Programming languages and systems: Proceedings of ESOP 2017, 26th European symposium on programming*, ed. Yang Hongseok, 368–392. Lecture Notes in Computer Science 10201, Berlin: Springer.

Davidson-Pilon, Cameron. 2016. *Bayesian methods for hackers: Probabilistic programming and Bayesian inference*. Boston: Addison-Wesley.

Eddy, Sean R. 2004. What is Bayesian statistics? *Nature Biotechnology* 22(9):1177–1178.

Heunen, Chris, Ohad Kammar, Sam Staton, and Hongseok Yang. 2017. A convenient category for higher-order probability theory. In *LICS 2017: Proceedings of the 32nd symposium on logic in computer science*, 1–12. Washington, DC: IEEE Computer Society Press.

MacKay, David J. C. 1998. Introduction to Monte Carlo methods. In *Learning and inference in graphical models*, ed. Michael I. Jordan. Dordrecht: Kluwer. Paperback: *Learning in Graphical Models*, MIT Press.

McElreath, Richard. 2017. Markov chains: Why walk when you can flow? `http://elevanth.org/blog/2017/11/28/build-a-better-markov-chain/`.

Ścibior, Adam, Ohad Kammar, Matthijs Vákár, Sam Staton, Hongseok Yang, Yufei Cai, Klaus Ostermann, Sean K. Moss, Chris Heunen, and Zoubin Ghahramani. 2018. Denotational validation of higher-order Bayesian inference. In *POPL '18: Conference record of the annual ACM symposium on principles of programming languages*. New York: ACM Press.

Shan, Chung-chieh, and Norman Ramsey. 2017. Exact Bayesian inference by symbolic disintegration. In *POPL '17: Conference record of the annual ACM symposium on principles of programming languages*, 130–144. New York: ACM Press.

Staton, Sam. 2017. Commutative semantics for probabilistic programming. In *Programming languages and systems: Proceedings of ESOP 2017, 26th European symposium on programming*, ed. Yang Hongseok, 855–879. Lecture Notes in Computer Science 10201, Berlin: Springer.

Tierney, Luke. 1998. A note on Metropolis-Hastings kernels for general state spaces. *The Annals of Applied Probability* 8(1): 1–9.