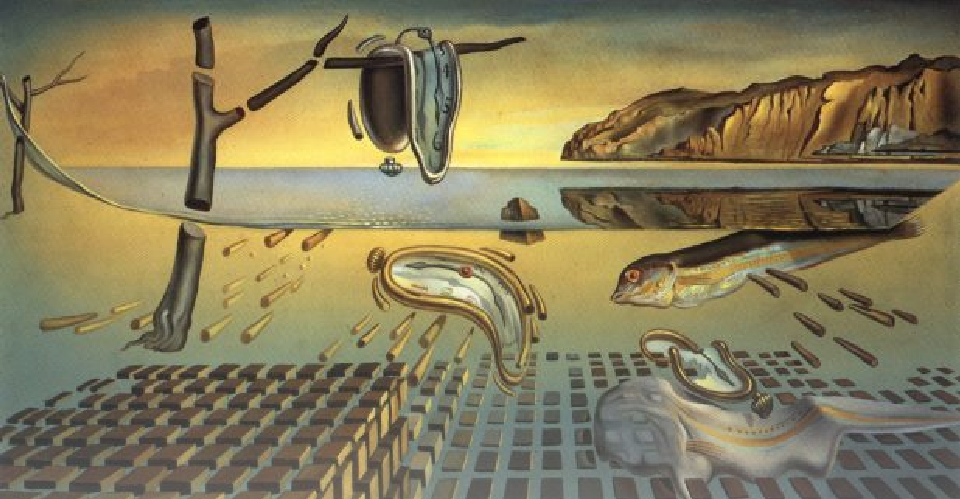


Modular probabilistic inference by program transformations

Chung-chieh Shan · QAPL, 2–3 April 2016

Programming Techniques

R. Morris
Editor

On the Criteria To Be Used in Decomposing Systems into Modules

D.L. Parnas
Carnegie-Mellon University

This paper discusses modularization as a mechanism for improving the flexibility and comprehensibility of a system while allowing the shortening of its development time. The effectiveness of a "modularization" is dependent upon the criteria used in dividing the system into modules. A system design problem is presented and both a conventional and unconventional decomposition are described. It is shown that the unconventional decompositions have distinct advantages for the goals outlined. The criteria used in arriving at the decompositions are discussed. The unconventional decomposition, if implemented with the conventional assumption that a module consists of one or more subroutines, will be less efficient in most cases. An alternative approach to implementation which does not have this effect is sketched.

Introduction

A lucid statement of the philosophy of modular programming can be found in a 1970 textbook on the design of system programs by Gouthier and Pont [1, ¶10.23], which we quote below:[1]

> A well-defined segmentation of the project effort ensures system modularity. Each task forms a separate, distinct program module. At implementation time each module and its inputs and outputs are well-defined, there is no confusion in the intended interface with other system modules. At checkout time the integrity of the module is tested independently; there are few scheduling problems in synchronizing the completion of several tasks before checkout can begin. Finally, the system is maintained in modular fashion; system errors and deficiencies can be traced to specific system modules, thus limiting the scope of detailed error searching.

Usually nothing is said about the criteria to be used


TYO
TPE
SFO
IND
ITH
NYC
BOS
AAR
AMS
BZO

Probabilistic Programm ×

www.darpa.mil/program/probabilistic-programming-for-advancing-machine-Learning

EXPLORE TOPICS

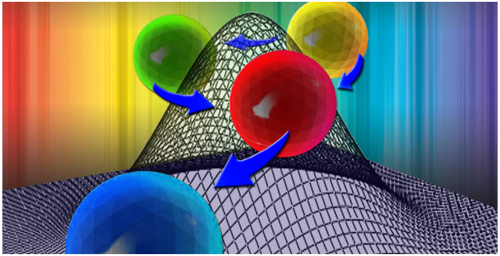
DARPA DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

ABOUT US / OUR RESEARCH / NEWS / EVENTS / WORK WITH US /

Defense Advanced Research Projects Agency > Program Information > Probabilistic Programming for Advancing Machine Learning

Probabilistic Programming for Advancing Machine Learning (PPAML)

Dr. Suresh Jagannathan



Machine learning – the ability of computers to understand data, manage results and infer insights from uncertain information – is the force behind many recent revolutions in computing. Email spam filters, smartphone personal assistants and self-driving vehicles are all based on research advances in machine learning. Unfortunately, even as the demand for these capabilities is accelerating, every new application requires a Herculean effort. Teams of hard-to-find experts must build expensive, custom tools that are often painfully slow and can perform unpredictably against large, complex data sets.

RESOURCES

- PPAML FAQ
- PPAML Proposers Day
- PPAML Proposers Day Attendee List
- Proposers Day Opening Remarks (Video)
- Proposers Day Contracts (Video)
- Proposers Day Security (Video)
- Proposers Day PPAML Program (Video)
- Proposers Day Q and A (Video)

Major features

- BNT supports many types of **conditional probability distributions** (nodes), and it is easy to add more.
  - Tabular (multinomial)
  - Gaussian
  - Softmax (logistic/ sigmoid)
  - Multi-layer perceptron (neural network)
  - Noisy-or
  - Deterministic

- BNT supports **decision and utility nodes**, as well as chance nodes, i.e., influence diagrams as well as Bayes nets.

- BNT supports static and dynamic BNs (useful for modelling dynamical systems and sequence data).

- BNT supports many different **inference algorithms**, and it is easy to add more.
  - Exact inference for static BNs:
    - junction tree
    - variable elimination
    - brute force enumeration (for discrete nets)

Sorting analogy

Ordering	+	Sorting technique	=	Sorting procedure

Ordering
- Numeric
- Alphabetical
- Case folding
- Reverse
- ...

Sorting technique
- Merge sort
- Quick sort
- Insertion sort
- Radix sort
- ...

Distribution	+	Inference technique	=	Inference procedure
Generative story		Interpreter/compiler		Probabilistic programming

parametric Bayesian extension of the HMM with an infinite number of hidden states. Exact Bayesian inference for the iHMM is intractable. Specifically, given a particular setting of the parameters the forward-backward algorithm cannot be applied since the number of states K is infinite, while with the parameters marginalized out all hidden state variables will be coupled and the forward-backward algorithm cannot be applied either. Currently the only approximate inference algorithm available is Gibbs sampling, where individual hidden state variables are resampled conditioned on all other variables (Teh et al., 2006). Unfor-
...tely convergence of ...s sampling is noto...
...he HMM s... the strong...
...co... ...ft...
Fina... ...ove... ...ers
and γ. ...n solution, w... not have strong belie... about the hyperpara...ers, is to use gamma hyperpriors: $\alpha \sim \mathsf{Gamma}(a_\alpha, b_\alpha)$ and $\gamma \sim \mathsf{Gamma}(a_\gamma, b_\gamma)$. (Teh et al., 2006) describe how these hyperparameters can be sampled efficiently, and we will use this in the experiments to follow.

3. The Gibbs Sampler

The Gibbs sampler was the first sampling algorithm for the iHMM that converges to the true posterior. One proposal builds on the direct assignment sampling scheme for the HDP in (Teh et al., 2006) by marginalizing out the hidden variables $\boldsymbol{\pi}, \boldsymbol{\phi}$ from (2), (3) and ignoring the ordering of states implicit in $\boldsymbol{\beta}$. Thus we only need to sample the hidden trajectory $\boldsymbol{s}$, the base

2. The Infinite Hidden Markov Model

We start this section by describing the finite HMM, then taking the infinite limit to obtain an intuition for the infinite HMM, followed by a more precise definition. A finite HMM consists of a hidden state sequence $\boldsymbol{s} = (s_1, s_2, \ldots, s_T)$ and a corresponding observation sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$. Each state variable s_t can take on a finite number of states, say $1 \ldots K$. Transitions between states are governed by Markov dynamics parameterized by the transition matrix $\boldsymbol{\pi}$, where $\pi_{ij} = p(s_t = j | s_{t-1} = i)$, while the initial state prob...ilities are $\pi_{0i} = p(s_1 = i)$. For each state $s_t \in$ ... there is a p... which
...aram... ...tion l... ...tate:
...ina... ...iv...
...cause large... ...$\boldsymbol{s}$ to be
... now introduce t... ...am sampler which
not suffer from this slow mixing behavior by sampling the whole sequence $\boldsymbol{s}$ in one go.

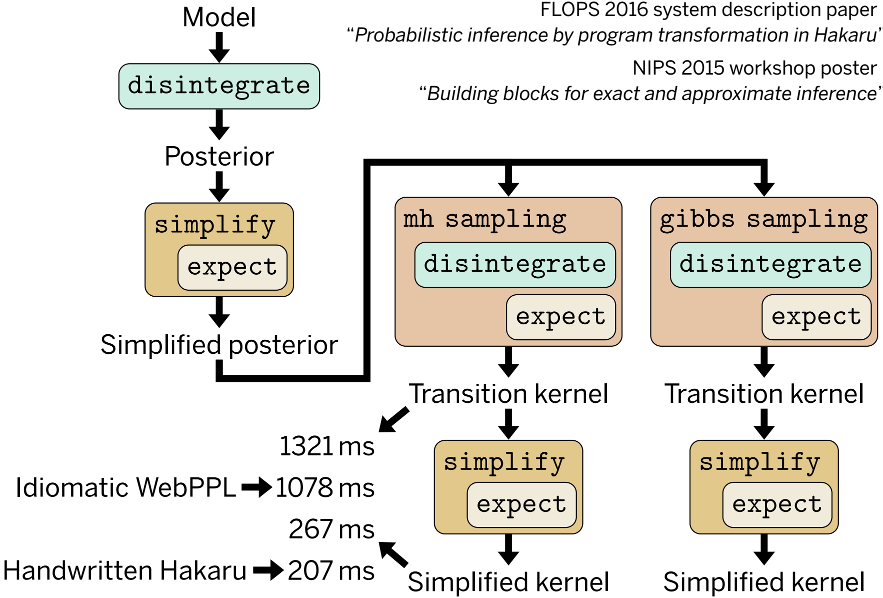
4. The Beam Sampler

The forward-backward algorithm does not apply to the iHMM because the number of states, and hence the number of potential state trajectories, are infinite. The idea of beam sampling is to introduce auxiliary variables $\boldsymbol{u}$ such that conditioned on $\boldsymbol{u}$ the number of trajectories with positive probability is finite. Now dynamic programming can be used to compute the conditional probabilities of each of these trajectories and thus sample *whole* trajectories efficiently. These

Hakaru: meaningful and reusable, from clear to fast

FLOPS 2016 system description paper
"*Probabilistic inference by program transformation in Hakaru*"

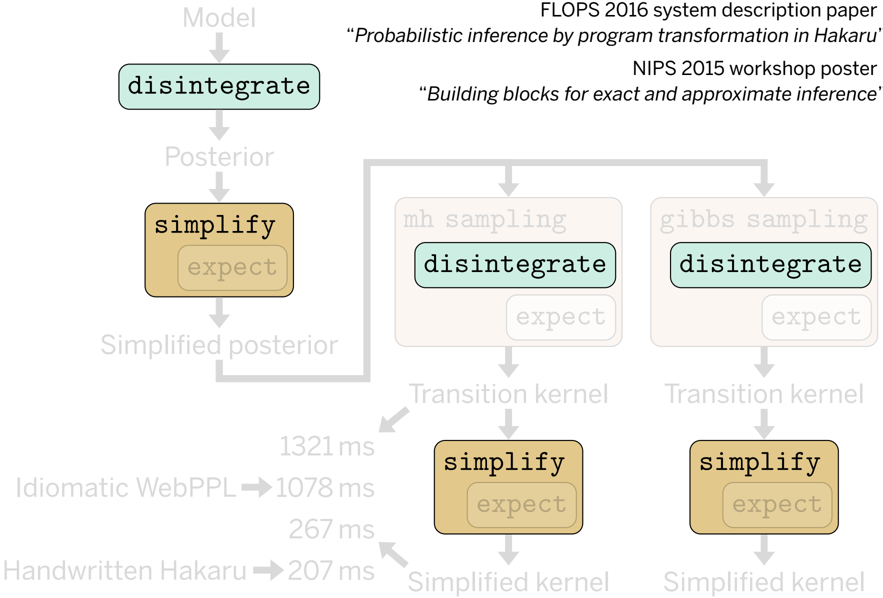
NIPS 2015 workshop poster
"*Building blocks for exact and approximate inference*"



Hakaru: meaningful and reusable, from clear to fast

FLOPS 2016 system description paper
"Probabilistic inference by program transformation in Hakaru"

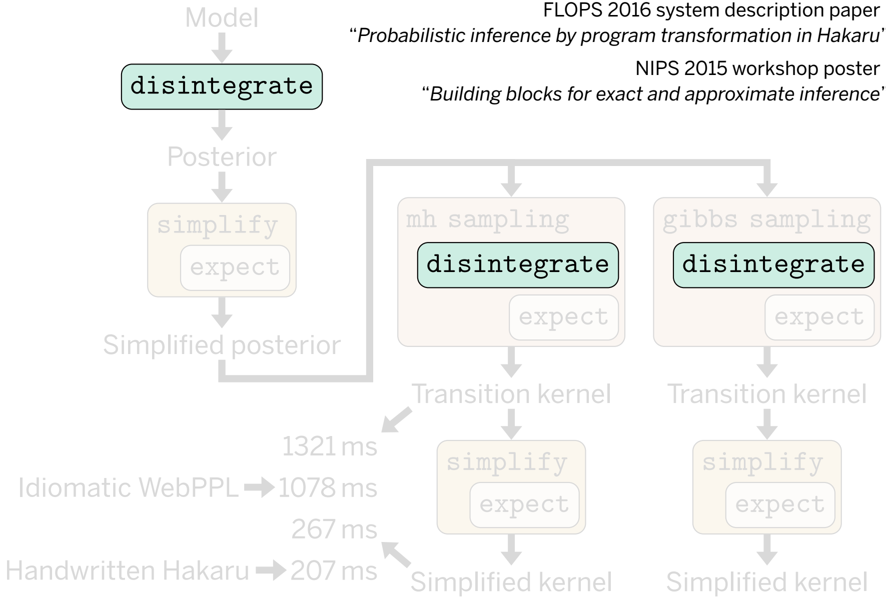
NIPS 2015 workshop poster
"Building blocks for exact and approximate inference"



Hakaru: meaningful and reusable, from clear to fast

FLOPS 2016 system description paper
"*Probabilistic inference by program transformation in Hakaru*"

NIPS 2015 workshop poster
"*Building blocks for exact and approximate inference*"



Disintegration for medical diagnosis

Diseases A and B are equally prevalent.

Disease A causes one of symptoms 1, 2, 3 with equal probability.

Disease B causes one of symptoms 1, 2 with equal probability.

$$\wr A \mapsto {}^{1}\!/_{2}, B \mapsto {}^{1}\!/_{2} \int \qquad : \mathbb{M}\ \text{Disease}$$

Disintegration for medical diagnosis

Diseases A and B are equally prevalent.

Disease A causes one of symptoms 1, 2, 3 with equal probability.

Disease B causes one of symptoms 1, 2 with equal probability.

$$
\begin{array}{l}
\textbf{do}\ \{\mathit{disease} \leftsquigarrow \lbag A \mapsto 1/2, B \mapsto 1/2 \rbag; \\
\qquad \mathit{symptom} \leftsquigarrow \textbf{case}\ \mathit{disease}\ \textbf{of} \\
\qquad\qquad\qquad\qquad A \to \lbag 1 \mapsto 1/3, 2 \mapsto 1/3, 3 \mapsto 1/3 \rbag \\
\qquad\qquad\qquad\qquad B \to \lbag 1 \mapsto 1/2, 2 \mapsto 1/2 \rbag; \\
\qquad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\text{Symptom} \times \text{Disease})
\end{array}
$$

Disintegration for medical diagnosis

Diseases A and B are equally prevalent.
Disease A causes one of symptoms 1, 2, 3 with equal probability.
Disease B causes one of symptoms 1, 2 with equal probability.

$$\begin{aligned}
&\textbf{do}\ \{\mathit{disease} \leftsquigarrow \wr A \mapsto 1/2, B \mapsto 1/2 \int; \\
&\qquad \mathit{symptom} \leftsquigarrow \textbf{case}\ \mathit{disease}\ \textbf{of} \\
&\qquad\qquad\qquad A \to \wr 1 \mapsto 1/3, 2 \mapsto 1/3, 3 \mapsto 1/3 \int \\
&\qquad\qquad\qquad B \to \wr 1 \mapsto 1/2, 2 \mapsto 1/2 \int; \\
&\qquad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\mathrm{Symptom} \times \mathrm{Disease})
\end{aligned}$$

$$\begin{aligned}
= \wr &(1, A) \mapsto 1/6, (2, A) \mapsto 1/6, (3, A) \mapsto 1/6, \\
&(1, B) \mapsto 1/4, (2, B) \mapsto 1/4 \int
\end{aligned}$$

$$\begin{array}{c|ccc}
A & 1/6 & 1/6 & 1/6 \\
B & 1/4 & 1/4 & 0 \\
\hline
 & 1 & 2 & 3
\end{array}$$

Disintegration for medical diagnosis

Diseases A and B are equally prevalent.
Disease A causes one of symptoms 1, 2, 3 with equal probability.
Disease B causes one of symptoms 1, 2 with equal probability.

$$\begin{array}{l}\textbf{do}\ \{\mathit{disease} \leftsquigarrow \wr A \mapsto 1/2, B \mapsto 1/2 \int;\\ \qquad \mathit{symptom} \leftsquigarrow \textbf{case}\ \mathit{disease}\ \textbf{of}\\ \qquad\qquad A \to \wr 1 \mapsto 1/3, 2 \mapsto 1/3, 3 \mapsto 1/3 \int\\ \qquad\qquad B \to \wr 1 \mapsto 1/2, 2 \mapsto 1/2 \int;\\ \qquad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\,(\text{Symptom} \times \text{Disease})\end{array}$$

$$= \wr (1, A) \mapsto 1/6, (2, A) \mapsto 1/6, (3, A) \mapsto 1/6, (1, B) \mapsto 1/4, (2, B) \mapsto 1/4 \int$$

	1	2	3
A	1/6	1/6	1/6
B	1/4	1/4	0

model

posterior

$$\begin{array}{l}\lambda \mathit{symptom}.\ \textbf{case}\ \mathit{symptom}\ \textbf{of}\\ \qquad 1 \to \wr A \mapsto 1/6, B \mapsto 1/4 \int\\ \qquad 2 \to \wr A \mapsto 1/6, B \mapsto 1/4 \int\\ \qquad 3 \to \wr A \mapsto 1/6 \int \qquad : \text{Symptom} \to \mathbb{M}\,\text{Disease}\end{array}$$

Disintegration on a zero-probability observation

Diseases A and B are equally prevalent.
Disease A causes a symptom chosen uniformly from $[0,3] \subset \mathbb{R}$.
Disease B causes a symptom chosen uniformly from $[0,2] \subset \mathbb{R}$.

$$\wr A \mapsto 1/2, B \mapsto 1/2 \int \qquad : \mathbb{M}\ \text{Disease}$$

Disintegration on a zero-probability observation

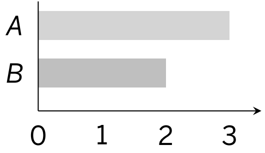
Diseases *A* and *B* are equally prevalent.
Disease *A* causes a symptom chosen uniformly from $[0, 3] \subset \mathbb{R}$.
Disease *B* causes a symptom chosen uniformly from $[0, 2] \subset \mathbb{R}$.

$$
\begin{array}{l}
\textbf{do}\ \{\mathit{disease} \leftarrow\!\!\sim \wr A \mapsto 1/2, B \mapsto 1/2 \wr;\\
\quad \mathit{symptom} \leftarrow\!\!\sim \textbf{case}\ \mathit{disease}\ \textbf{of}\\
\qquad\qquad A \to \textbf{uniform}\ 0\ 3\\
\qquad\qquad B \to \textbf{uniform}\ 0\ 2;\\
\quad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\mathrm{Symptom} \times \mathrm{Disease})
\end{array}
$$

Disintegration on a zero-probability observation

Diseases A and B are equally prevalent.
Disease A causes a symptom chosen uniformly from $[0, 3] \subset \mathbb{R}$.
Disease B causes a symptom chosen uniformly from $[0, 2] \subset \mathbb{R}$.

$$
\begin{array}{l}
\textbf{do } \{\mathit{disease} \leftsquigarrow \lbag A \mapsto 1/2, B \mapsto 1/2 \rbag; \\
\qquad \mathit{symptom} \leftsquigarrow \textbf{case } \mathit{disease} \textbf{ of} \\
\qquad\qquad\qquad A \to \textbf{uniform } 0\ 3 \\
\qquad\qquad\qquad B \to \textbf{uniform } 0\ 2; \\
\qquad \textbf{return } (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\text{Symptom} \times \text{Disease})
\end{array}
$$

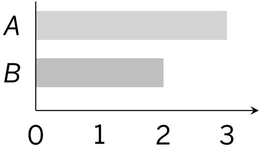


Disintegration on a zero-probability observation

Diseases A and B are equally prevalent.
Disease A causes a symptom chosen uniformly from $[0,3] \subset \mathbb{R}$.
Disease B causes a symptom chosen uniformly from $[0,2] \subset \mathbb{R}$.

$$\begin{array}{l} \textbf{do}\ \{\mathit{disease} \leftsquigarrow \wr A \mapsto 1/2, B \mapsto 1/2 \wr; \\ \quad \mathit{symptom} \leftsquigarrow \textbf{case}\ \mathit{disease}\ \textbf{of} \\ \qquad\qquad A \to \textbf{uniform}\ 0\ 3 \\ \qquad\qquad B \to \textbf{uniform}\ 0\ 2; \\ \quad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\text{Symptom} \times \text{Disease}) \end{array}$$

$\Downarrow$ model / posterior



$$\begin{array}{l} \lambda \mathit{symptom}.\ \textbf{if}\ \mathit{symptom} \leq 2 \\ \qquad \textbf{then}\ \wr A \mapsto 1/6, B \mapsto 1/4 \wr \\ \qquad \textbf{else}\ \ \wr A \mapsto 1/6 \wr \qquad : \text{Symptom} \to \mathbb{M}\ \text{Disease} \end{array}$$

Disintegration on a zero-probability observation

Choose *disease* uniformly from $[1, 3] \subset \mathbb{R}$.

Choose *symptom* uniformly from $[0, \textit{disease}] \subset \mathbb{R}$.

$$\textbf{uniform}\ 1\ 3 \qquad : \mathbb{M}\ \text{Disease}$$

Disintegration on a zero-probability observation

Choose *disease* uniformly from $[1, 3] \subset \mathbb{R}$.

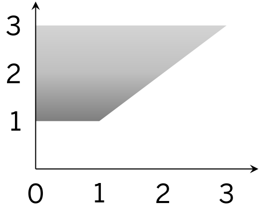
Choose *symptom* uniformly from $[0, \mathit{disease}] \subset \mathbb{R}$.

$$
\begin{aligned}
&\textbf{do}\ \{\mathit{disease} \leftsquigarrow \textbf{uniform}\ 1\ 3;\\
&\qquad \mathit{symptom} \leftsquigarrow \textbf{uniform}\ 0\ \mathit{disease};\\
&\qquad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\mathrm{Symptom} \times \mathrm{Disease})
\end{aligned}
$$

Disintegration on a zero-probability observation

Choose *disease* uniformly from $[1, 3] \subset \mathbb{R}$.
Choose *symptom* uniformly from $[0, \mathit{disease}] \subset \mathbb{R}$.

$$\begin{aligned}&\textbf{do}\ \{\mathit{disease} \leftsquigarrow \textbf{uniform}\ 1\ 3;\\ &\qquad \mathit{symptom} \leftsquigarrow \textbf{uniform}\ 0\ \mathit{disease};\\ &\qquad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\ (\text{Symptom} \times \text{Disease})\end{aligned}$$

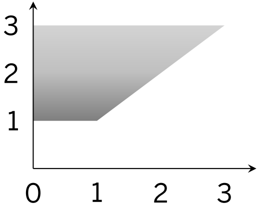


Disintegration on a zero-probability observation

Choose *disease* uniformly from $[1,3] \subset \mathbb{R}$.
Choose *symptom* uniformly from $[0, \mathit{disease}] \subset \mathbb{R}$.

$$\begin{aligned}&\textbf{do}\ \{\mathit{disease} \leftsquigarrow \textbf{uniform}\ 1\ 3;\\ &\qquad \mathit{symptom} \leftsquigarrow \textbf{uniform}\ 0\ \mathit{disease};\\ &\qquad \textbf{return}\ (\mathit{symptom}, \mathit{disease})\} \qquad : \mathbb{M}\,(\mathrm{Symptom} \times \mathrm{Disease})\end{aligned}$$

model
$\Downarrow$
posterior



$$\begin{aligned}&\lambda \mathit{symptom}.\ \textbf{do}\ \{\mathit{disease} \leftsquigarrow \textbf{uniform}\ 1\ 3;\\ &\qquad \textbf{if}\ 0 \le \mathit{symptom} \le \mathit{disease}\\ &\qquad\quad \textbf{then}\ \lbag \mathit{disease} \mapsto {}^1\!/_{\mathit{disease}} \rbag\\ &\qquad\quad \textbf{else}\ \lbag\rbag\} \qquad : \mathrm{Symptom} \to \mathbb{M}\ \mathrm{Disease}\end{aligned}$$

Measure semantics

$$[\![\mathbb{M}\ \alpha]\!] \quad = \quad (\alpha \to \mathbb{R}) \to \mathbb{R}$$

Measure semantics

$$
\begin{aligned}
[\![\mathbb{M}\ \alpha]\!] &= (\alpha \to \mathbb{R}) \to \mathbb{R} \\
[\![\wr A \mapsto 1/2, B \mapsto 1/2 \wr]\!](f) &= \frac{f(A)}{2} + \frac{f(B)}{2} \\
[\![\textbf{return}\ (\mathit{symptom}, \mathit{disease})]\!](f) &= f(\mathit{symptom}, \mathit{disease}) \\
[\![\textbf{uniform}\ 1\ 3]\!](f) &= \int_1^3 \frac{f(x)}{2}\,\mathrm{d}x \\
[\![\textbf{lebesgue}]\!](f) &= \int_{-\infty}^{\infty} f(x)\,\mathrm{d}x \\
[\![\textbf{do}\ \{x \leftsquigarrow m;\ k\}]\!](f) &= [\![m]\!](\lambda x.\ [\![k]\!]f)
\end{aligned}
$$

Measure semantics

$$\begin{aligned}
[\![\mathbb{M}\ \alpha]\!] &= (\alpha \to \mathbb{R}) \to \mathbb{R} \\
[\![\wr A \mapsto 1/2, B \mapsto 1/2 \wr]\!](f) &= \frac{f(A)}{2} + \frac{f(B)}{2} \\
[\![\mathbf{return}\ (symptom, disease)]\!](f) &= f(symptom, disease) \\
[\![\mathbf{uniform}\ 1\ 3]\!](f) &= \int_1^3 \frac{f(x)}{2}\,\mathrm{d}x \\
[\![\mathbf{lebesgue}]\!](f) &= \int_{-\infty}^{\infty} f(x)\,\mathrm{d}x \\
[\![\mathbf{do}\ \{x \leftsquigarrow m;\ k\}]\!](f) &= [\![m]\!](\lambda x.\ [\![k]\!]f) \\
\left[\!\!\left[\begin{array}{l}\mathbf{do}\ \{d \leftsquigarrow \mathbf{uniform}\ 1\ 3; \\ \qquad s \leftsquigarrow \mathbf{uniform}\ 0\ d; \\ \qquad \mathbf{return}\ (s, d)\}\end{array}\right]\!\!\right](f) &= \underbrace{\int_1^3 \int_0^d \frac{f(s,d)}{2 \cdot d}\,\mathrm{d}s\,\mathrm{d}d}_{\textit{Patently linear in } f}
\end{aligned}$$

Disintegration specification

$$[\![m]\!] = [\![\textbf{do}\ \{s \leftsquigarrow \textbf{lebesgue};\ d \leftsquigarrow k;\ \textbf{return}\ (s,d)\}]\!]$$

model

posterior

Disintegration specification

$$[\![m]\!] = [\![\mathbf{do}\ \{s \leftsquigarrow \mathbf{lebesgue};\ d \leftsquigarrow k;\ \mathbf{return}\ (s, d)\}]\!]$$

model

posterior

$$\begin{aligned} m = \mathbf{do}\ \{&d \leftsquigarrow \mathbf{uniform}\ 1\ 3; \\ &s \leftsquigarrow \mathbf{uniform}\ 0\ d; \\ &\mathbf{return}\ (s, d)\} \end{aligned}$$

$$\begin{aligned} k = \mathbf{do}\ \{&d \leftsquigarrow \mathbf{uniform}\ 1\ 3; \\ &\mathbf{if}\ 0 \le s \le d \\ &\quad \mathbf{then}\ \wr d \mapsto {}^{1}\!/_{d} \int \\ &\quad \mathbf{else}\ \ \wr\int\} \end{aligned}$$

Disintegration specification

$$[\![m]\!] = [\![\textbf{do}\ \{s \leftsquigarrow \textbf{lebesgue};\ d \leftsquigarrow k;\ \textbf{return}\ (s,d)\}]\!]$$

model posterior

$$\begin{aligned} m = \textbf{do}\ \{&d \leftsquigarrow \textbf{uniform}\ 1\ 3;\\ &s \leftsquigarrow \textbf{uniform}\ 0\ d;\\ &\textbf{return}\ (s,d)\} \end{aligned}$$

$$\begin{aligned} k = \textbf{do}\ \{&d \leftsquigarrow \textbf{uniform}\ 1\ 3;\\ &\textbf{if}\ 0 \le s \le d\\ &\quad \textbf{then}\ \wr d \mapsto {}^1\!/_d \int\\ &\quad \textbf{else}\ \ \wr\int\} \end{aligned}$$

$$[\![k]\!](f) = \int_1^3 \frac{\textbf{if}\ 0 \le s \le d\ \textbf{then}\ f(d)/d\ \textbf{else}\ 0}{2}\,\mathrm{d}d$$

$$\begin{aligned} &[\![\textbf{do}\ \{s \leftsquigarrow \textbf{lebesgue};\ d \leftsquigarrow k;\ \textbf{return}\ (s,d)\}]\!](f)\\ &\quad = \int_{-\infty}^{\infty}\int_1^3 \frac{\textbf{if}\ 0 \le s \le d\ \textbf{then}\ f(s,d)/d\ \textbf{else}\ 0}{2}\,\mathrm{d}d\,\mathrm{d}s\\ &\quad = \int_1^3\int_0^d \frac{f(s,d)}{2\cdot d}\,\mathrm{d}s\,\mathrm{d}d = [\![m]\!] \end{aligned}$$

Useful but unspecified and thus unautomated before

1. Initialise $x_{0,1:n}$.
2. For $i = 0$ to $N - 1$
   - Sample $x_1^{(i+1)} \sim p(x_1 | x_2^{(i)}, x_3^{(i)}, \ldots, x_n^{(i)})$.
   - Sample $x_2^{(i+1)} \sim p(x_2 | x_1^{(i+1)}, x_3^{(i)}, \ldots, x_n^{(i)})$.

     $\vdots$
   - Sample $x_j^{(i+1)} \sim p(x_j | x_1^{(i+1)}, \ldots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, \ldots, x_n^{(i)})$.

     $\vdots$
   - Sample $x_n^{(i+1)} \sim p(x_n | x_1^{(i+1)}, x_2^{(i+1)}, \ldots x_{n-1}^{(i+1)})$.

Figure 12. Gibbs sampler.

(Borel paradox)

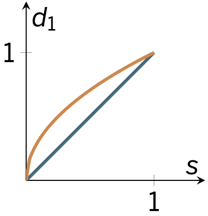
Determinism requires inversion

$$\mathbf{do}\ \{d \leftsquigarrow \mathbf{uniform}\ 0\ 1;\\ s \leftsquigarrow \mathbf{return}\ (2 \cdot d);\\ \mathbf{return}\ (s, d)\}$$

Determinism requires inversion

$$\begin{aligned}&\textbf{do}\ \{d \leftsquigarrow \textbf{uniform}\ 0\ 1;\\ &\quad s \leftsquigarrow \textbf{return}\ (2 \cdot d);\\ &\quad \textbf{return}\ (s, d)\}\end{aligned}$$

$$\begin{aligned}&\textbf{do}\ \{d_1 \leftsquigarrow \textbf{uniform}\ 0\ 1;\\ &\quad d_2 \leftsquigarrow \lbrace\!\!\lbrace 1 \mapsto 1/2, 2 \mapsto 1/2 \rbrace\!\!\rbrace;\\ &\quad s \leftsquigarrow \textbf{return}\ d_1^{d_2};\\ &\quad \textbf{return}\ (s, (d_1, d_2))\}\end{aligned}$$



(Deterministic observable)

Radio Yerevan

Question: Is it correct that Grigori Grigorievich Grigoriev won a luxury car at the All-Union Championship in Moscow?

Answer: In principle, yes.

But first of all it was not Grigori Grigorievich Grigoriev, but Vassili Vassilievich Vassiliev.

Second, it was not at the All-Union Championship in Moscow, but at a Collective Farm Sports Festival in Smolensk.

Third, it was not a car, but a bicycle.

And fourth he didn't win it, but rather it was stolen from him.

Automatic disintegrator

Question: Is it correct that our disintegrator is a lazy evaluator?

Answer: In principle, yes.

$$\text{evaluate} : \lceil \alpha \rceil \to H \to (\alpha \times H)$$

Automatic disintegrator

Question: Is it correct that our disintegrator is a lazy evaluator?

Answer: In principle, yes.

But first of all it is not an evaluator, but a partial evaluator.

$$\text{evaluate} : \lceil \alpha \rceil \to H \to (\lfloor \alpha \rfloor \times H)$$

Automatic disintegrator

Question: Is it correct that our disintegrator is a lazy evaluator?

Answer: In principle, yes.

But first of all it is not an evaluator, but a partial evaluator.

Second, it not only evaluates terms, but also performs random choices.

$$\text{evaluate} : \lceil \quad \alpha \rceil \to H \to (\lfloor \alpha \rfloor \to H \to \lfloor \mathbb{M}\ \gamma \rfloor) \to \lfloor \mathbb{M}\ \gamma \rfloor$$

$$\text{perform} : \lceil \mathbb{M}\ \alpha \rceil \to H \to (\lfloor \alpha \rfloor \to H \to \lfloor \mathbb{M}\ \gamma \rfloor) \to \lfloor \mathbb{M}\ \gamma \rfloor$$

Automatic disintegrator

Question: Is it correct that our disintegrator is a lazy evaluator?

Answer: In principle, yes.

But first of all it is not an evaluator, but a partial evaluator.

Second, it not only evaluates terms, but also performs random choices.

Third, it not only produces outcomes and values, but also constrains them.

$$
\begin{aligned}
\text{evaluate} &: \lceil \quad \alpha \rceil \to H \to (\lfloor \alpha \rfloor \to H \to \lfloor \mathbb{M}\ \gamma \rfloor) \to \lfloor \mathbb{M}\ \gamma \rfloor \\
\text{perform} &: \lceil \mathbb{M}\ \alpha \rceil \to H \to (\lfloor \alpha \rfloor \to H \to \lfloor \mathbb{M}\ \gamma \rfloor) \to \lfloor \mathbb{M}\ \gamma \rfloor \\
\text{constrain-value} &: \lceil \quad \alpha \rceil \to \lfloor \alpha \rfloor \to H \to (H \to \lfloor \mathbb{M}\ \gamma \rfloor) \to \lfloor \mathbb{M}\ \gamma \rfloor \\
\text{constrain-outcome} &: \lceil \mathbb{M}\ \alpha \rceil \to \lfloor \alpha \rfloor \to H \to (H \to \lfloor \mathbb{M}\ \gamma \rfloor) \to \lfloor \mathbb{M}\ \gamma \rfloor
\end{aligned}
$$

Automatic disintegrator

Question: Is it correct that our disintegrator is a lazy evaluator?

Answer: In principle, yes.

But first of all it is not an evaluator, but a partial evaluator.

Second, it not only evaluates terms, but also performs random choices.

Third, it not only produces outcomes and values, but also constrains them.

And fourth it doesn't produce one term, but searches for a random variable to constrain.

$$\begin{aligned}
\text{evaluate} &: \lceil \phantom{\mathbb{M}}\ \alpha \rceil \to H \to (\lfloor \alpha \rfloor \to H \to \{\lfloor \mathbb{M}\ \gamma \rfloor\}) \to \{\lfloor \mathbb{M}\ \gamma \rfloor\} \\
\text{perform} &: \lceil \mathbb{M}\ \alpha \rceil \to H \to (\lfloor \alpha \rfloor \to H \to \{\lfloor \mathbb{M}\ \gamma \rfloor\}) \to \{\lfloor \mathbb{M}\ \gamma \rfloor\} \\
\text{constrain-value} &: \lceil \phantom{\mathbb{M}}\ \alpha \rceil \to \lfloor \alpha \rfloor \to H \to (H \to \{\lfloor \mathbb{M}\ \gamma \rfloor\}) \to \{\lfloor \mathbb{M}\ \gamma \rfloor\} \\
\text{constrain-outcome} &: \lceil \mathbb{M}\ \alpha \rceil \to \lfloor \alpha \rfloor \to H \to (H \to \{\lfloor \mathbb{M}\ \gamma \rfloor\}) \to \{\lfloor \mathbb{M}\ \gamma \rfloor\}
\end{aligned}$$

Automatic disintegrator in action

$$\left\lceil \text{perform}\;(\textbf{do}\;\{d \leftsquigarrow \textbf{uniform}\;1\;3;\; s \leftsquigarrow \textbf{uniform}\;0\;d;\; \textbf{return}\;(s,d)\})\right.^{[]}$$

$$[d' \leftsquigarrow \textbf{uniform}\;1\;3]$$

$$\text{perform}\;(\textbf{do}\;\{s \leftsquigarrow \textbf{uniform}\;0\;d';\; \textbf{return}\;(s,d')\})$$

$$[d' \leftsquigarrow \textbf{uniform}\;1\;3;\; s' \leftsquigarrow \textbf{uniform}\;0\;d']$$

$$\text{perform}\;(\textbf{return}\;(s',d'))$$

$$\left\lfloor \text{evaluate}\;(s',d') \Rightarrow (s',d') \right.$$

$$\left\lceil \text{constrain-value}\;s'\;s \right.$$

$$\left\lceil \text{constrain-outcome}\;(\textbf{uniform}\;0\;d')\;s \right.$$

Automatic disintegrator in action

$$\text{perform}\ (\textbf{do}\ \{d \leftsquigarrow \textbf{uniform}\ 1\ 3;\ s \leftsquigarrow \textbf{uniform}\ 0\ d;\ \textbf{return}\ (s,d)\})^{[]}$$

$$[d' \leftsquigarrow \textbf{uniform}\ 1\ 3]$$

$$\text{perform}\ (\textbf{do}\ \{s \leftsquigarrow \textbf{uniform}\ 0\ d';\ \textbf{return}\ (s,d')\})$$

$$[d' \leftsquigarrow \textbf{uniform}\ 1\ 3;\ s' \leftsquigarrow \textbf{uniform}\ 0\ d']$$

$$\text{perform}\ (\textbf{return}\ (s',d'))$$

$$\text{evaluate}\ (s',d') \Rightarrow (s',d')$$

$$\text{constrain-value}\ s'\ s$$

$$\text{constrain-outcome}\ (\textbf{uniform}\ 0\ d')\ s$$

nondeterminism

$$\text{evaluate}\ 0 \Rightarrow 0$$

$$\text{evaluate}\ d'$$

$$\text{perform}\ (\textbf{uniform}\ 1\ 3)$$

$$\textbf{do}\ \{d'' \leftsquigarrow \textbf{uniform}\ 1\ 3;\ \square\}$$

$$\Rightarrow d''$$

$$[\textbf{let}\ d' = d'';\ s' \leftsquigarrow \textbf{uniform}\ 0\ d']$$

$$\Rightarrow d''$$

$$\textbf{if}\ 0 \le s \le d''\ \textbf{then do}\ \{() \leftsquigarrow \wr () \mapsto 1/d'' \int;\ \square\}\ \textbf{else}\ \wr\int$$

$$[\textbf{let}\ d' = d'';\ \textbf{let}\ s' = s]$$

Automatic disintegrator in action

$$\left[\begin{array}{l}
\text{perform}\ (\textbf{do}\ \{d \leftsquigarrow \textbf{uniform}\ 1\ 3;\ s \leftsquigarrow \textbf{uniform}\ 0\ d;\ \textbf{return}\ (s,d)\})^{[]} \\
\hfill [d' \leftsquigarrow \textbf{uniform}\ 1\ 3] \\
\text{perform}\ (\textbf{do}\ \{s \leftsquigarrow \textbf{uniform}\ 0\ d';\ \textbf{return}\ (s,d')\}) \\
\hfill [d' \leftsquigarrow \textbf{uniform}\ 1\ 3;\ s' \leftsquigarrow \textbf{uniform}\ 0\ d'] \\
\text{perform}\ (\textbf{return}\ (s',d')) \\
\text{evaluate}\ (s',d') \Rightarrow (s',d')
\end{array}\right.$$

$$\left[\begin{array}{l}
\text{constrain-value}\ s'\ s \\
\left[\begin{array}{l}
\text{constrain-outcome}\ (\textbf{uniform}\ 0\ d')\ s \qquad\qquad \text{nondeterminism} \\
\left[\text{evaluate}\ 0 \Rightarrow 0\right. \\
\left[\begin{array}{l}
\text{evaluate}\ d' \\
\left[\begin{array}{l}
\text{perform}\ (\textbf{uniform}\ 1\ 3) \\
\Rightarrow d''
\end{array}\right. \\
\Rightarrow d''
\end{array}\right.
\end{array}\right.
\end{array}\right.$$

$$\textbf{do}\ \{d'' \leftsquigarrow \textbf{uniform}\ 1\ 3;\ \Box\}$$

$$[\textbf{let}\ d' = d'';\ s' \leftsquigarrow \textbf{uniform}\ 0\ d']$$

$$\textbf{if}\ 0 \le s \le d''\ \textbf{then do}\ \{() \leftsquigarrow \lbag () \mapsto 1/d'' \rbag;\ \Box\}\ \textbf{else}\ \lbag\rbag$$

$$[\textbf{let}\ d' = d'';\ \textbf{let}\ s' = s]$$

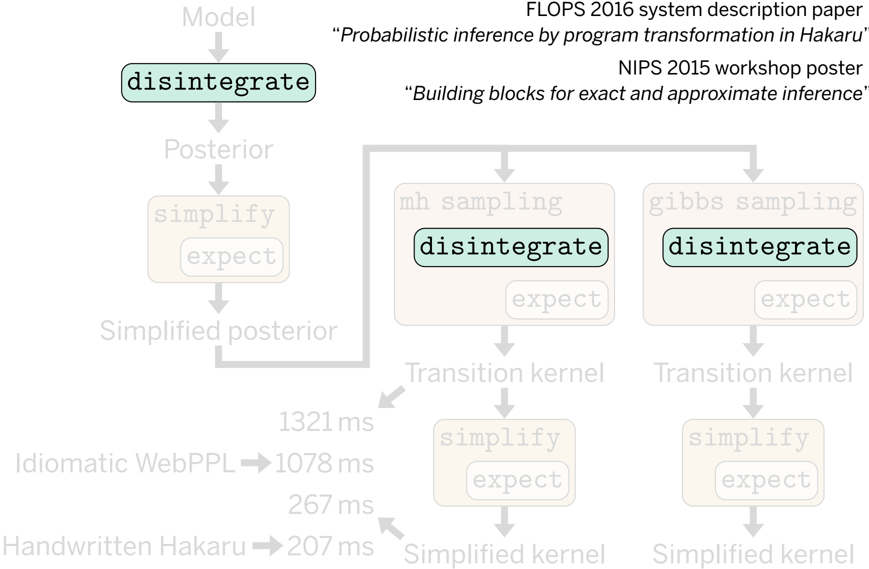
Interim summary

- Generate observed symptoms from hidden causes
- **First exact inference algorithm** for continuous distributions
- **Enable modular composition** of inference techniques
- **Lessons for language design** and reasoning
- Ongoing work: arrays
  more dominating measures
  more deterministic observables
  prove correctness

Hakaru: meaningful and reusable, from clear to fast

FLOPS 2016 system description paper
"*Probabilistic inference by program transformation in Hakaru*"

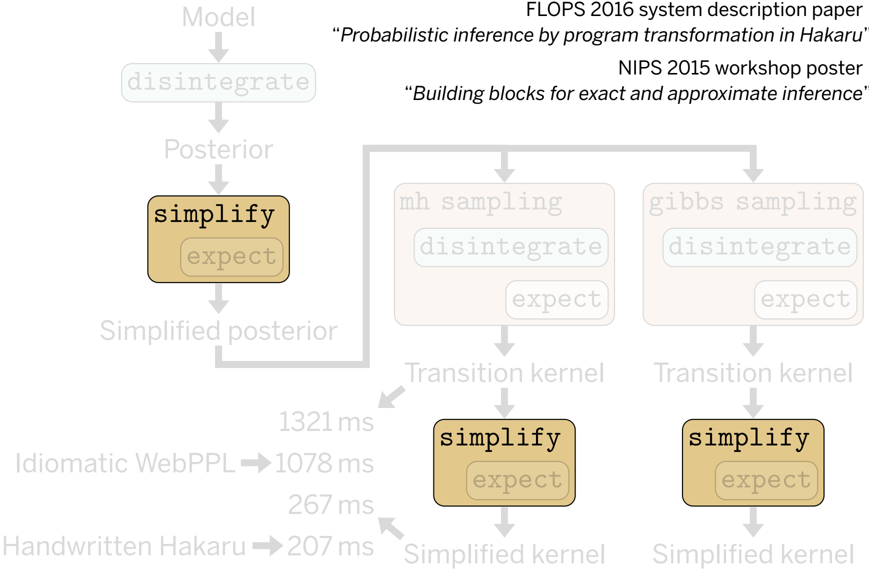
NIPS 2015 workshop poster
"*Building blocks for exact and approximate inference*"



Hakaru: meaningful and reusable, from clear to fast

FLOPS 2016 system description paper
"*Probabilistic inference by program transformation in Hakaru*"

NIPS 2015 workshop poster
"*Building blocks for exact and approximate inference*"



Simplifying probabilistic programs via semantics

PADL 2016 paper
"Simplifying probabilistic programs using computer algebra"

Probabilistic program

Continuation passing

Abstract integral

Computer algebra

Improved integral

Computer algebra

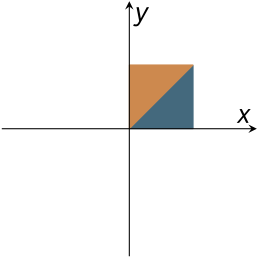
Simplified program

Patently linear

Simplifying a discrete distribution

do $\{x \leftsquigarrow$ **uniform** $0\ 1;$
$\quad y \leftsquigarrow$ **uniform** $0\ 1;$
$\quad$ **if** $x < y$ **then return true**
$\quad\quad$ **else return false**$\}$

PADL 2016 paper
"*Simplifying probabilistic programs using computer algebra*"



$$\int_0^1 \int_0^1 \left(\begin{cases} f(\mathbf{true}) & \text{if } x < y \\ f(\mathbf{false}) & \text{otherwise} \end{cases} \right) dy\, dx$$

Symbolic integration

$$\frac{1}{2} \cdot f(\mathbf{true}) + \frac{1}{2} \cdot f(\mathbf{false})$$

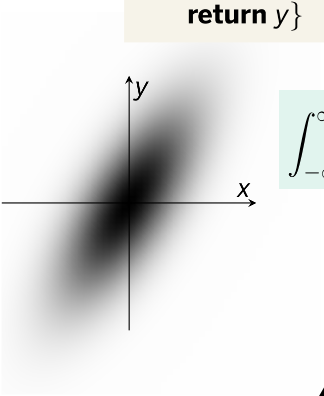
Patently linear in f

$$\wr\, \mathbf{true} \mapsto 1/2,\ \mathbf{false} \mapsto 1/2 \,\wr$$

Simplifying a continuous distribution

PADL 2016 paper
"*Simplifying probabilistic programs using computer algebra*"

$$\textbf{do}\ \{x \leftsquigarrow \textbf{normal}\ 0\ 1;\ y \leftsquigarrow \textbf{normal}\ x\ 1;\ \textbf{return}\ y\}$$



$$\int_{-\infty}^{\infty} \frac{\exp\left(-\frac{x^2}{2}\right)}{\sqrt{2\cdot\pi}} \cdot \int_{-\infty}^{\infty} \frac{\exp\left(-\frac{(y-x)^2}{2}\right)}{\sqrt{2\cdot\pi}} \cdot f(y)\, dy\, dx$$

Symbolic integration

$$\int_{-\infty}^{\infty} \frac{\exp\left(-\frac{y^2}{4}\right)}{2\cdot\sqrt{\pi}} \cdot f(y)\, dy$$

Holonomic representation

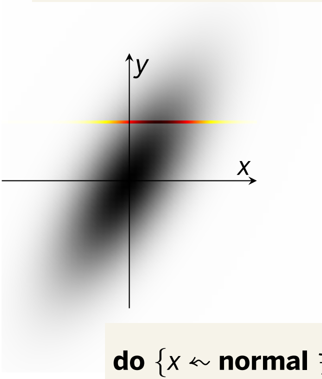
$$\textbf{normal}\ 0\ \sqrt{2}$$

Patently linear in f

Simplifying a conditional distribution

PADL 2016 paper
"*Simplifying probabilistic programs using computer algebra*"

$$\textbf{do}\ \{x \leftarrow \textbf{normal}\ 0\ 1;\ \lbag x \mapsto \textbf{density}\ (\textbf{normal}\ x\ 1)\ y \rbag\}$$



$$\int_{-\infty}^{\infty} \frac{\exp\left(-\frac{x^2}{2}\right)}{\sqrt{2\cdot\pi}} \cdot \frac{\exp\left(-\frac{(y-x)^2}{2}\right)}{\sqrt{2\cdot\pi}} \cdot f(x)\,dx$$

"

Holonomic representation

Patently linear in f

$$\textbf{do}\ \left\{x \leftarrow \textbf{normal}\ \frac{y}{2}\ \frac{1}{\sqrt{2}};\ \lbag x \mapsto \frac{\exp\left(-\frac{y^2}{4}\right)}{2\cdot\sqrt{\pi}} \rbag\right\}$$

Use conjugacies (**normal**, **gamma**, …)
without pairwise hard-coding

Summary

General strategy:

- represent concepts
- formal, hence **executable**
- **meaningful**, hence modular

Meaning-preserving **transformations** on probabilistic programs!

- disintegrate
- simplify
- etc.

Example **modularity** payoffs:

- combine exact and approximate techniques without re-coding
- use conjugacies without pairwise hard-coding
- same **performance**