Inference in Probabilistic Programming I

Xin Zhang
Peking University

Part of the content is from "An Introduction to Probabilistic Programming" by Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood

Recap of Last Lecture

- Regarding inference, before talking about **How**, we need to define **What**

- Program semantics: formally define what a program computes
  - Modular
  - Can be used to answer various questions in a mechanic way

Recap of Last Lecture

- Operational semantics

- Denotational semantics

General Approximate Inference Techniques

- Variational inference
- Transformation method
- Rejection sampling
- Importance sampling
- Markov chain Monte Carlo

This Class and Next Class

- We are going to talk about instantiating general inference techniques in probabilistic programming

Question

- Which construct of a language might cause trouble for inference?

Outline of the Lecture

- Graph-based inference (this lecture)
  - More on MCMC
- Evaluation-based inference (next lecture)

Graph-Based Inference: Introduction

- Key idea: convert a program into a graphical model
  - We know to do inference on graphical models

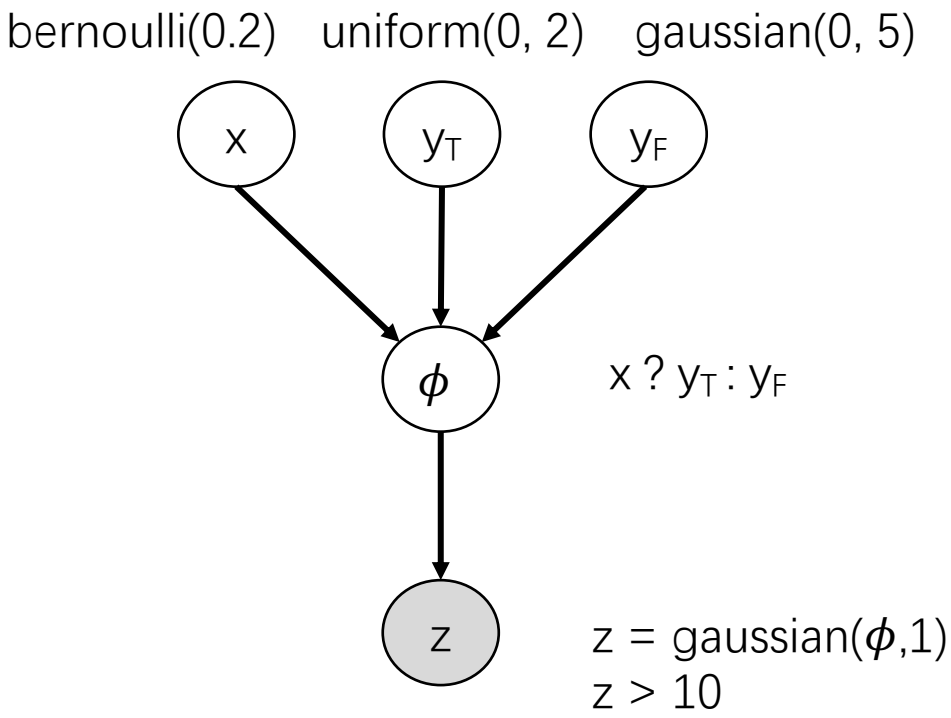
- Limitation: a static method
  - Have trouble to deal with cases where conditional dependences are dynamic
  - Moreover, cannot deal with loops that can iterate for arbitrarily many times

Graph-Based Inference: Example 1

```
x = bernoulli(0.2)
if(x){
        y = uniform(0, 2)
}
else
        y = gaussian(0, 5)

z = gaussian(y, 1)

condition(z>10)
```



Graph-Based Inference: Example 2

```
x = guassian(0, 1)
y = uniform(0, x)
if (x>10){
        condition(y >1.5)
}
else{
        condition(y<0.5)
}
```

Formal Definition of Transformation

- What information should a graph contain?

Formal Definition: Graph (Bayesian Network)

$$G = (V, A, P, Y)$$

- V is a set of vertices
- A is a set of arcs
- P is a map that defines the density functions or mass functions of each variable
- Y is a set that tracks the conditioned variables

Formal Definition: Our Language (SSA)

- $t ::= a \ \ (a \in R, constant) \mid v \ (v \ is \ a \ variable) \mid \ t \ op \ t \ (op \in \{+, -, \times, \div\} \mid \text{phi}(b, v_1, v_2) \mid$
  $uniform(t, t) \mid gaussian \ (t, t) \mid bernoulli(t)$

- $b ::= \ true \mid false \mid t > t \mid t < t \mid t == t \mid b \&\& b \mid b \mid\mid b$

- $e ::= skip \mid \text{e; e} \mid \text{if b then e else e} \mid \text{condition(b)} \mid \text{v} = t$

Translation

$$\rho, \phi, G, e \Downarrow \rho', \phi', G'$$

- ρ: environment, which maps a variable to a constant or a node variable
- ϕ: path condition
- e: program

Translation Rules: Terms

$$\rho, \phi, G, t \Downarrow G', E$$

E is a deterministic expression

$$\frac{}{\rho, \phi, G, a \Downarrow G, a} \qquad \frac{}{\rho, \phi, G, v \Downarrow G, \rho[v]}$$

$$\frac{\rho, \phi, G, b \Downarrow G', E'}{\rho, \phi, G, \phi(b, v_1, v_2) \Downarrow G', if\ E'\ then\ \rho(v_1) else\ \rho(v_{2)}}$$

$$\frac{\rho, \phi, G, t_1 \Downarrow G_1, E_1 \quad \rho, \phi, G, t_2 \Downarrow G_2, E_2}{\rho, \phi, G, t_1\ op\ t_2 \Downarrow G_1 + G_2, E_1\ op\ E_2}$$

Translation Rules: Terms

$$\rho, \phi, G, t \Downarrow G', E$$

$$\frac{\begin{array}{c}\rho, \phi, G, t_1 \Downarrow G_1, E_1 \quad \rho, \phi, G, t_2 \Downarrow G_2, E_2 \\ \hat{v}\ is\ a\ fresh\ variable, V\ are\ free\ varaibles\ in\ E_1\ and\ E_2 \\ F = score\big(uniform(E_1, E_2)\big)\end{array}}{\begin{array}{l}\rho, \phi, G, uniform(t_1, t_2) \Downarrow \\ G_1 + G_2 + (\{\hat{v}\}, \{(v, \hat{v}) | v \in V\}, \{\hat{v} \mapsto F\}, \{\}),\quad \hat{v}\end{array}}$$

Translation Rules: Tests

$$\rho, \phi, G, t \Downarrow G', E$$

$$\frac{}{\rho, \phi, G, true \Downarrow G, true} \qquad \frac{}{\rho, \phi, G, false \Downarrow G, false}$$

$$\frac{\rho, \phi, G, t_1 \Downarrow G_1, E_1 \quad \rho, \phi, G, t_2 \Downarrow G_2, E_2}{\rho, \phi, G, t_1\ op\ t_2 \Downarrow G_1 + G_2, E_1\ op\ E_2}$$

Translation Rules: Program

$$\frac{}{\rho, \phi, G, skip \Downarrow \rho, \phi, G}$$

$$\frac{\rho, \phi, G, t \Downarrow G', E}{\rho, \phi, G, x := t \Downarrow \rho[x \mapsto E], \phi, G'}$$

$$\frac{\rho, \phi, G, e_1 \Downarrow \rho_1, \phi_1, G_1 \qquad \rho_1, \phi_1, G_1, e_2 \Downarrow \rho_2, \phi_2, G_2}{\rho, \phi, G, e_1; e_2 \Downarrow \rho_2, \phi_2, G_2}$$

Translation Rules: Program

$$\frac{\begin{array}{c}\rho, \phi, G, b \Downarrow G', E \\ \rho, \phi \wedge E, G', e_1 \Downarrow \rho_1, \phi_1, G_1 \qquad \rho, \phi \wedge \neg E, G', e_2 \Downarrow \rho_2, \phi_2, G_2\end{array}}{\rho, \phi, G, if\ b\ then\ e_1 else\ e_2 \Downarrow \rho_1 + \rho_2, \phi, G_1 + G_2}$$

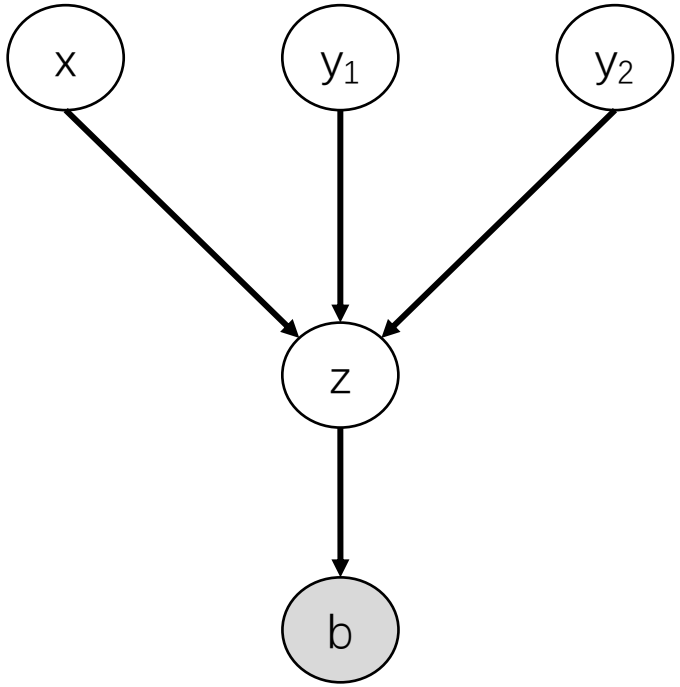
$$\frac{\begin{array}{c}\rho, \phi, G, b \Downarrow (V, A, P, Y), E \\ \hat{v}\ is\ a\ fresh\ variable \qquad F = if\ \phi\ score(E, \hat{v})\ else\ 1 \\ V\ are\ free\ varaibles\ in\ F/\{\hat{v}\}\end{array}}{\rho, \phi, G, condition(b) \Downarrow \rho, \phi, (V \cup \{\hat{v}\}, A \cup \{(v, \hat{v}) | v \in V\}, P \cup \{\hat{v} \mapsto F\}, Y \cup \{\hat{v}\})}$$

Translation: Example

x = bernoulli(0.2)
if(x){
    y_1 = uniform(0, 2)
}
else
    y_2 = gaussian(0, 5)

y_3 = phi(x, y_1, y_2)
z = gaussian(y_3, 1)

condition(z>10)



Translation: Questions

- Are the translated graphs always trees?
- How do we deal with function calls?
- How to evaluate the density/mass function?

Translation: Questions

- What about factor?

$$\frac{\begin{array}{c}\rho, \phi, G, b \Downarrow (V, A, P, Y), E \\ \hat{v}\ is\ a\ fresh\ variable \qquad F = if\ \phi \exp(score(E, \hat{v}))\ else\ 1 \\ V\ are\ free\ varaibles\ in\ E\end{array}}{\rho, \phi, G, factor(b) \Downarrow \rho, (V \cup \{\hat{v}\}, A \cup \{(v, \hat{v}) | v \in V\}, P \cup \{\hat{v} \mapsto F\}, Y)}$$

Translation: Questions

- Can we simplify the graph before inference?

Partial evaluation: function calls and if statements

Inference on the Translated Graph

- If we want to compute marginal probabilities or most likely assignment, we can use (loopy) belief propagation

- But we often want to draw samples, so methods like sampling methods are used more often

Sampling Method

- Rejection sampling
  - Straightforward, but often inefficient

- MCMC
  - Used most widely
  - Need to evaluate Z*p, where Z can be any positive number

MCMC Variant: Gibbs Sampling

- Often we want to change one assignment at a time
- Proposal distribution
  - Change one assignment at a time
  - p(x | Y, X\{x}), where Y are observed variables
- Proof of correctness
  - Stationary: change x doesn't affect P(V\{x}), therefore P(V') = P(V\{x}) * P(x | V\{x}) = P(V)
  - Ergodic: depends on the distribution. A sufficient condition: none of the conditional distribution is anywhere zero

More on Gibbs Sampling

- Very useful when we can analytically compute p(x | V/{x})

- When not possible, we turn to Metropolis-within-Gibbs algorithms
  - Use a proposal distribution q(x | V/{x})
  - Still use the acceptance probability in Metropolis-Hasting

Why Metropolis-within-Gibbs?

- When computing

$$A_k(\mathbf{z}^\star, \mathbf{z}^{(\tau)}) = \min\left(1, \frac{\widetilde{p}(\mathbf{z}^\star) q_k(\mathbf{z}^{(\tau)} | \mathbf{z}^\star)}{\widetilde{p}(\mathbf{z}^{(\tau)}) q_k(\mathbf{z}^\star | \mathbf{z}^{(\tau)})}\right).$$

- Many terms in the two $p(z)$ will cancel out
  - $\frac{p(V)}{p(V')} = \frac{p(x \mid \bar{V}) * p(\bar{V})}{p(x' \mid \bar{V}) * p(\bar{V})} = \frac{p(x \mid \bar{V})}{p(x' \mid \bar{V})}$
  - We can further decompose $p(x \mid \bar{V})$ into products of conditional probability, we only need to evaluate the part that involves x

Full Description of Metropolis-within-Gibbs

- Page 78 of "An Introduction to Probabilistic Programming"

Optimization: Block Sampling

- Sample highly correlated variables together
- Example:

```
x = gaussian(0, 1)
y = gaussian(0, 1)
z = gaussian(x+y, 0.01)
condition(z == 2)
```

- Need to analyze the model to do appropriately

Hamiltonian Monte Carlo (HMC)

- In many optimization techniques, gradients are good guidance

- Hamiltonian Monte Carlo is an MCMC technique that utilizes gradients
  - Works for continuous variables
  - Scales better for high dimensional distributions
  - Make large changes while keeping the rejection probability small
  - Analogy to dynamical systems in physics
  - Compared to MH: the proposal distribution uses information from the target distribution

Dynamical Systems

- Key idea of HMC is to cast probabilistic simulation in the form of a Hamiltonian system
  - Exploiting the framework of Hamiltonian dynamics

- Classical dynamics: Newton's second law of motion
  - $a = v' = S'' = F/M$

Dynamical Systems

- The dynamics we consider: the evolution of $\mathbf{z} = \{z_i\}$ under continuous time τ
- Intermediate momentum variable $r_i = \frac{dz_i}{d\tau}$, $\mathbf{r} = \{r_i\}$
- We view $\mathbf{z}$ as position variables
- The joint space of position and momentum is called phase space

Intuition of HMC

- We can imagine a ball in a bowl without friction
  - Define p = f(E), where E is the potential energy of the ball
  - The higher the probability is, the lower the potential energy is
  - We use ball's position to sample p
  - Sometimes we need to give the ball a random kick

- Visualization: https://arogozhnikov.github.io/2016/12/19/markov_chain_monte_carlo.html

Probability Distributions to Dynamical Systems

- We can write $p(\mathbf{z})$ as

$$p(\mathbf{z}) = \frac{1}{Z_p} \exp\left(-E(\mathbf{z})\right)$$

- $E(\mathbf{z})$ is the potential energy of the system in state $\mathbf{z}$

- The system acceleration is

$$\frac{\mathrm{d}r_i}{\mathrm{d}\tau} = -\frac{\partial E(\mathbf{z})}{\partial z_i}.$$

Probability Distributions to Dynamical Systems

- Kinetic energy:

$$K(\mathbf{r}) = \frac{1}{2}\|\mathbf{r}\|^2 = \frac{1}{2}\sum_i r_i^2$$

- The total energy of the dynamical system is given by the Hamiltonian function:

$$H(\mathbf{z}, \mathbf{r}) = E(\mathbf{z}) + K(\mathbf{r})$$

Probability Distributions to Dynamical Systems

- The dynamics of the systems can be expressed using Hamiltonian equations:

$$\begin{aligned} \frac{\mathrm{d}z_i}{\mathrm{d}\tau} &= \frac{\partial H}{\partial r_i} \\ \frac{\mathrm{d}r_i}{\mathrm{d}\tau} &= -\frac{\partial H}{\partial z_i}. \end{aligned}$$

Properties of Hamiltonian Dynamical Systems

- During the evolution of the system, the Hamiltonian is constant

$$\frac{\mathrm{d}H}{\mathrm{d}\tau} = \sum_i \left\{ \frac{\partial H}{\partial z_i}\frac{\mathrm{d}z_i}{\mathrm{d}\tau} + \frac{\partial H}{\partial r_i}\frac{\mathrm{d}r_i}{\mathrm{d}\tau} \right\}$$

$$= \sum_i \left\{ \frac{\partial H}{\partial z_i}\frac{\partial H}{\partial r_i} - \frac{\partial H}{\partial r_i}\frac{\partial H}{\partial z_i} \right\} = 0$$

Properties of Hamiltonian Dynamical Systems

- Liouville's theorem: they preserve volume in phase space ($\mathbf{z}$, $\mathbf{r}$)

- To see why it holds, we define the flow field

$$\mathbf{V} = \left(\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}\tau}, \frac{\mathrm{d}\mathbf{r}}{\mathrm{d}\tau} \right)$$

- The divergence of this field vanishes

Probability Distributions to Dynamical Systems

- We now define joint distribution

$$p(\mathbf{z}, \mathbf{r}) = \frac{1}{Z_H} \exp(-H(\mathbf{z}, \mathbf{r})).$$

- Using the two properties of Hamiltonian Dynamical Systems, we can show that the Hamiltonian dynamics will leave p(**z,r**) invariant

$$\begin{aligned} \frac{\mathrm{d}z_i}{\mathrm{d}\tau} &= \frac{\partial H}{\partial r_i} \\ \frac{\mathrm{d}r_i}{\mathrm{d}\tau} &= -\frac{\partial H}{\partial z_i}. \end{aligned}$$

Probability Distributions to Dynamical Systems

- Using the two properties of Hamiltonian Dynamical Systems, we can show that the Hamiltonian dynamics will leave p(**z**,**r**) invariant

- We can integrate over a finite time duration to make large changes to **z** in a systematic way

Probability Distributions to Dynamical Systems

- However, sampling using the Hamiltonian dynamics will not form an ergodic Markov chain because H is constant

- To fix it, we can replace the value $\mathbf{r}$ with $p(\mathbf{r}|\mathbf{z})$, which can be a gaussian because $\mathbf{z}$ and $\mathbf{r}$ are independent

Put Things Together: HMC

- Augment distribution $p(\boldsymbol{z})$ with $p(\boldsymbol{z}, \boldsymbol{r})$

- Proposal distribution:
  - Update $\boldsymbol{z}, \boldsymbol{r}$ using Hamiltonian dynamics (in practice, a discretized approximation called leapfrog integration)
  - Update $\boldsymbol{r}$ stochastically

- Acceptance probability (After applying Hamiltonian dynamics):

$$\min\left(1, \exp\{H(\mathbf{z}, \mathbf{r}) - H(\mathbf{z}^\star, \mathbf{r}^\star)\}\right)$$

Account for approximation

The Leapfrog Approximation

$$
\begin{aligned}
\widehat{r}_i(\tau + \epsilon/2) &= \widehat{r}_i(\tau) - \frac{\epsilon}{2}\frac{\partial E}{\partial z_i}(\widehat{\mathbf{z}}(\tau)) \\
\widehat{z}_i(\tau + \epsilon) &= \widehat{z}_i(\tau) + \epsilon \widehat{r}_i(\tau + \epsilon/2) \\
\widehat{r}_i(\tau + \epsilon) &= \widehat{r}_i(\tau + \epsilon/2) - \frac{\epsilon}{2}\frac{\partial E}{\partial z_i}(\widehat{\mathbf{z}}(\tau + \epsilon)).
\end{aligned}
$$

To remove biases introduced by numerical errors,
the steps are sampled from ϵ and $-\epsilon$

Why HMC is usually better than MH?

- MH has difficulties exploring low-density points
- HMC uses information of the target distribution
  - This will always go to the high-density points
- Momentum helps!

HMC in Probabilistic Programming

- Have trouble with branching statements

- The density function has to be differentiable everywhere
  - What about 0 gradients?

Next Lecture

- Evaluation-based inference