Probabilistic Programming Homework 3

Submit your solutions to the TA in the homework submission box in the third floor of the E3-1 building by 2:00pm on 6 December 2017 (Wednesday). If you type up your solutions, you can email them to him (bskim90@kaist.ac.kr).

Question 1

One main selling point of the theory of quasi-Borel spaces is that it supports higher-order functions well and is more suitable for interpreting higher-order probabilistic programming languages. In this question, we will check one property related to this claim.

Let (X, M) and (Y, N) be quasi-Borel spaces. Define (Z, O) to be the quasi-Borel space of QBS morphisms, usually denoted by $[(X, M) \to_q (Y, N)]$. That is,

$$Z := \{f : X \to Y \mid f \text{ is a QBS morphism from } (X, M) \text{ to } (Y, N)\}$$
$$O := \{g : \mathbb{R} \to Z \mid \text{function } r \mapsto g(\gamma(r))(\alpha(r)) \text{ is in } N \text{ for all } \alpha \in M \text{ and measurable } \gamma : \mathbb{R} \to \mathbb{R}\}.$$

Show that the evaluation operator

$$ev : Z \times X \to Y, \qquad ev(f, x) := f(x)$$

is a QBS morphism from the QBS product $(Z, O) \times (X, M)$ to (Y, N). As we discussed, this is in contract with the situation in the measure theory that a similar evaluation operator on functions on reals cannot be made measurable. We remind the students of the definition of the QBS product below:

$$(W, L) := (Z, O) \times (X, M)$$

where

$$W := Z \times X, \qquad\qquad L := \{r \mapsto (\alpha(r), \beta(r)) \mid \alpha \in O \text{ and } \beta \in N\}.$$

Question 2

This question is about amortised inference. Consider finite sets X and Y that are ranged over by x and y, respectively. Assume that we are given a probability distribution $p(x, y)$ on $X \times Y$ in the form of $p(x)$ and $p(y|x)$. The x part of (x, y) is an observed latent state, and the y part is an observation. We consider a situation where we want to perform posterior inference on $p(x, y)$ multiple times for different observations $y_1, y_2, \ldots, y_m$. The idea of amortised inference is to do certain preprocessing, which takes some time now but will save time in the future when we carry out these repeated inference tasks with observations $y_1, \ldots, y_m$.

More concretely, in the amortised inference, we consider a parameterised conditional proposal distribution $q_\theta(x|y)$ for $\theta \in \mathbb{R}^m$ such that for fixed x and y, the function $\theta \mapsto q_\theta(x|y)$ is a differentiable function from $\mathbb{R}^m$ to the interval $[0, 1]$. We often construct such a proposal using a neural net, in which case θ is the weights of the neural net. Given such a parameterised conditional proposal, the amortised inference works as follows. First, it solves the following optimisation problem:

$$\text{argmin}_\theta \left(\text{KL}\Big[p(x, y) \,\Big|\Big|\, p(y) \cdot q_\theta(x|y)\Big] \right) \tag{1}$$

where

$$\mathrm{KL}\Big[p(x,y)\,\Big|\Big|\,p(y)\cdot q_\theta(x|y)\Big] = \sum_{x,y}\left(p(x,y)\cdot\log\frac{p(x,y)}{p(y)\cdot q_\theta(x|y)}\right) = \mathbb{E}_{p(x,y)}\left[\log\frac{p(x,y)}{p(y)\cdot q_\theta(x|y)}\right].$$

Solving this optimisation problem corresponds to the preprocessing mentioned before. Let θ^* be the solution of this optimisation problem. Second, when we are given an observation y_i and asked to estimate the posterior $p(x|y_i)$, we instantiate $q_{\theta^*}(x|y_i)$ by y_i, and perform the importance sampling using $q_{\theta^*}(x|y_i)$ as a proposal.

Of course, the most challenging part of this amortised inference is to solve the optimisation problem (1). A common approach is to use gradient descent (which by the way interacts very well with the backpropagation algorithm for neural nets). In this question, we look at this gradient-descent algorithm in detail. You have two tasks to complete in this question. First, you need to prove the following key equation that lies behind the gradient-based algorithm:

$$\nabla_\theta\left(\mathrm{KL}\Big[p(x,y)\,\Big|\Big|\,p(y)\cdot q_\theta(x|y)\Big]\right) = \mathbb{E}_{p(x,y)}\left[-\frac{\nabla_\theta\, q_\theta(x|y)}{q_\theta(x|y)}\right].$$

Second, using the right-hand side of this equation, you need to derive a gradient-descent algorithm that approximately solves the problem (1). Your algorithm does not have to be super efficient.