AUTOMATIC BAYESIAN INFERENCE AS PROGRAMMING

FRANK WOOD, BROOKS PAIGE

Introduction

Probabilistic generative models can be written concisely as probabilistic programs. In this short exercise we will outline the basic structure of a probabilistic program, and show an example of automatic posterior inference.

A classical example problem deals with estimating the failure rate of a process, as in the following example:

> Suppose that you are thinking about purchasing a factory that makes pencils. Your accountants have determined that you can make a profit (i.e. you should transact the purchase) if the percentage of defective pencils manufactured by the factory is less than 30%.
>
> In your prior experience, you learned that, on average, pencil factories produce defective pencils at a rate of 50%.
>
> To make your judgement about the efficiency of this factory you test pencils one at a time in sequence as they emerge from the factory to see if they are defective.

We let $y_1, \ldots, y_N$, with $y_n \in \{0, 1\}$, be a set of defective / not defective observations. A very simple approach would be to model each observation y_n as an independent Bernoulli trial, with some unknown success rate p. We place a prior distribution on p, the shape of which represents the strength of our conviction that pencil factories produce 50% defective pencils. A traditional choice of prior might be a uniform distribution on the interval $[0, 1]$, the maximum entropy distribution for p which has an expected value of 0.5. In this case, our full model for the pencil factory data is

$$p \sim \text{Uniform}(0, 1) \tag{1}$$

$$y_n \sim \text{Bernoulli}(p). \tag{2}$$

Suppose the very first pencil that comes off the conveyor belt is defective. We can write this model as a probabilistic program, complete with observing our defective pencil, as

```
[assume p (uniform-continuous 0 1)]
[observe (flip p) false]
[predict p]
```

This program is exceedingly simple, but demonstrates the basic structure of a probabilistic program written in Anglican:

- `assume` directives define a prior on p;
- `observe` directives introduce observed data and a likelihood function;
- `predict` directives output posterior samples.

Questions:

(1) Recall that a uniform-continuous distribution over the interval $[0, 1]$ is identical to a Beta distribution with pseudocounts $a = b = 1$. After observing K successes from N trials, we can compute the posterior expectation and variance of p analytically:

$$\mathbb{E}[p] = \frac{a + K}{a + b + N} \tag{3}$$

$$\mathbb{V}[p] = \frac{(a + K)(b + N - K)}{(a + b + N)^2(a + b + N + 1)}. \tag{4}$$

Compare these with the values estimated by averaging over samples from the probabilistic program above (i.e. for $a = b = 1$, $N = 1$, $K = 0$). Confirm for yourself that the printed output of p draws from the correct posterior distribution.

Change the prior on p to be non-uniform, by modifying the first line of the program to draw from a `beta` density. Add a few more observations, of either good or defective pencils. Confirm for yourself that the program output matches the analytic posterior.

(2) Adding many observations in this manner is cumbersome. We'll introduce new notation later, but for now note that we can define our model for arbitrarily many observations by using the `binomial` density with a single `observe`, instead of using many observations of `flip`.

(3) Crucially, when writing probabilistic programs we are not limited to simple conjugate models like the one we have defined above. Instead of the `beta` prior on p, introduce a new prior for which we can no longer compute the posterior distribution by hand.

For example, suppose we were to place a truncated exponential prior on p, with

$$z \sim \text{Exponential}(0.5) \tag{5}$$

$$p|z = \min(z, 1) \tag{6}$$

Modify the probabilistic program to draw the success rate p from this new prior distribution. Suppose we find 3 defective pencils from the first 10 we test. What is the expected posterior failure rate of this pencil factory?

Try to come up with some more interesting, plausible, or outrageous programs to generate p.