1 Example 1: computing the posterior for a discrete distribution

Suppose we are interested in estimating the number of different fish in a particular lake. We extract 50 fish, and mark them. The a week later we extract another 50 fish, and count how many of them have marks. This approach is called "Capture-Mark-Recapture" Suppose X of them are marked, and we want to compute the posterior distribution on the number of fish.

I am going to use a prior $p(N) \propto 1/N$ for $N = 50, \ldots, 10^6$. Note that this is approximately uniform on the log scale, so represents being uncertain about N over orders of magnitude.

The likelihood is the number of ways of drawing 50 with X marked and $50 - X$ unmarked, divided by the total number of ways of drawing 50 fish.

$$P(X|N) = \frac{\binom{50}{X}\binom{N-50}{50-X}}{\binom{N}{50}} \tag{1}$$

Here's example R code for $X = 1$:

```
N=50:10^6
X=1 # for example!
prior = 1/N
prior = prior/sum(prior)
lik = choose(N-50,50-X)/choose(N,50) #I dropped the term not depending on N
post = prior*lik
post=post/sum(post) #normalize to sum to 1
```

Now we can compute the probability that N lies in any given interval. For example

```
sum(post[N<10^4])
[1] 0.7798534
```

so our posterior probability that there are fewer than 10,000 fish is about 0.78.

2 Example 2: Computation by naive Monte Carlo

This example is modeled on an example from Jim Berger.

Suppose we have an imperfect test for whether someone is affected with a condition (eg a virus). Let f be the overall prevalence of the condition, p_0 be the probability of a positive test result when the individual is unaffected, p_1 be the probability of a positive test result if the individual is affected.

Given a positive test result the probability of being affected is

$$p_A := fp_1/(fp_1 + (1-f)p_0).$$

Now suppose f, p_0 and p_1 are not known with certainty, but estimated based on the following pilot samples.

- For f: 1 out of 100 individuals were affected.
- For p_0: 1 out of 10 unaffected individuals gave a positive result when tested.
- For p_1: 8 out of 10 affected individuals gave a positive result when tested.

Assuming independent Beta(0.5,0.5) priors on f, p_0, p_1 the posteriors given these pilot data are also independent and $f \sim \text{Beta}(1.5, 99.5)$, $p_0 \sim \text{Beta}(1.5, 9.5)$ and $p_1 \sim \text{Beta}(8.5, 2.5)$.

Because it is straightforward to simulated form these posterior distributions, we can easily estimate a posterior median and 95% Credible Interval for p_A by naive Monte Carlo simulation from the posterior:

```
Nsim=10000
```

pAhist.pdf

Figure 1: Histograph of posterior samples of p_A from example 2

```
f = rbeta(Nsim,1.5,99.5)
p0= rbeta(Nsim,1.5,9.5)
p1= rbeta(Nsim,8.5,2.5)
pA = f*p1/(f*p1+(1-f)*p0)
quantile(pA,c(0.025,0.5,0.975))

       2.5%         50%       97.5%
0.005374967 0.074966382 0.542589567
```

Note: the posterior of p_A (Figure **??**) is sufficiently skew that the symmetric interval is not ideal here, but it is simple.

3 Example 3: Naive Monte Carlo vs Importance Sampling

This example is entirely artificial, to illustrate importance sampling. Throughout we assume that $X \sim \text{Beta}(2, 2)$. Suppose we want to estimate the probability that $X < 0.25$. We can do it easily by naive Monte Carlo simulation:

```
mean(rbeta(10000,2,2)<0.25)
[1] 0.1522
> pbeta(.25,2,2)
[1] 0.15625
```

The correct answer is 0.15625, so we're getting reasonable accuracy from the Monte Carlo estimate with only 10,000 simulations.

But if we want to estimate the probability that $X < 0.001$ we have more trouble:

```
> mean(rbeta(10000,2,2)<0.0001)
[1] 0
> pbeta(0.0001,2,2)
[1] 2.9998e-08
```

Well you might argue that 0 is not a bad estimate in absolute terms, but the *relative* error is large, and sometimes we might care about this.

Here importance sampling can help. The intuition is that we need to sample from a distribution that has more mass < 0.0001. How about Beta(0.5,0.5)?

```
y = rbeta(10000,0.5,0.5)
w = dbeta(y,2,2)/dbeta(y,0.5,0.5)
mean(w*(y<0.0001))
```

```
[1] 2.727679e-08
sd(w*(y<0.0001))
[1] 5.434473e-07
```

which is at least the right order of magnitude. But the standard deviation of the estimator is rather large, and we can do better. Note in particular that the Beta(0.5,0.5) spends a lot of it's time sampling parts of the space where the summands ($wI(y < 0.001)$)are 0. Recall, also, that the optimal IS distribution is proportional to Beta$(2, 2)I(X < 0.0001)$. Based on this, let's try uniform on (0,0.0001).

```
y = runif(10000,0,0.0001)
w = dbeta(y,2,2)/dunif(y,0,0.0001)
> mean(w*(y<0.0001))
[1] 2.959391e-08
> sd(w*(y<0.0001))
[1] 1.732338e-08
```

which we can see gives better accuracy and lower standard error.