L7: Bayesian Computation: Independent Monte Carlo Methods

1 Initial Look at Monte Carlo integration methods

While we will look at a variety of integration methods, both deterministic and stochastic, the most commonly used procedures for complex Bayesian models are Monte Carlo methods. As a preview of the Monte Carlo methods, consider the following problem. Suppose one wants to calculate the expected value of a random variable, θ:

$$E[\theta] = \int \theta p(\theta) d\theta \tag{1}$$

Suppose that calculation of the integral is difficult but one can relatively easily generate an arbitrarily large, and for now, independent, sample of size N from the probability distribution $p(\theta)$:

$$\theta^1, \theta^2, \ldots, \theta^N$$

Then a Monte Carlo estimate of the integral (1):

$$\hat{E}[\theta] = \frac{1}{N} \sum_{i=1}^{N} \theta^i \tag{2}$$

This is simply the sample average from the random (Monte Carlo) sample. If $E[\theta]$ exists (is finite), then by the strong Law of Large Numbers, $\hat{E}[\theta]$ converges with probability 1, which means:

$$\Pr(\lim_{N \to \infty} \hat{E}[\theta] = E[\theta]) = 1 \tag{3}$$

Many Bayesian integrals can be viewed as expectations. Two examples.

Probabilities: $\Pr(\theta > \theta^*)$. Calculating a probability over some interval(s) can be viewed as the expected value of the indicator random variable, $I(\theta > \theta^*)$:

$$E[I(\theta > \theta^*)] = \int p(\theta|\boldsymbol{y}) I(\theta > \theta^*) \mathrm{d}\theta = \int_{\theta^*}^{\infty} p(\theta|\boldsymbol{y}) \mathrm{d}\theta$$

Thus if one can generate a sample from $p(\theta|\boldsymbol{y})$, then

$$\hat{E}[I(\theta > \theta^*)] = \frac{1}{N} \sum_{i=1}^{N} I(\theta^i > \theta^*)$$

Normalising constants. The normalising constant for the posterior distribution in the denominator, namely $m(\mathbf{y})$, can be viewed as an expected value:

$$m(\mathbf{y}) = E[f(\boldsymbol{y}|\theta)] = \int f(\boldsymbol{y}|\theta) \pi(\theta) d\theta$$

If one draws a large independent sample of θ from the prior distribution $\pi(\theta)$, the normalising constant can be estimated as follows.

$$\hat{E}[f(\boldsymbol{y}|\theta)] = \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{y}|\theta^i)$$

Demonstration with Weibull example. Here we demonstrate Monte Carlo integration for the posterior distribution of α in the Weibull model for times between hurricanes (see Lecture Notes 6), in particular to estimate the normalising constant, $m(\mathbf{y})$. The integral to estimate is then

$$\int \pi(\alpha) f(\boldsymbol{y}|\alpha) d\alpha = \int \left[\frac{\lambda^\kappa}{\Gamma(\kappa)} \alpha^{\kappa-1} \exp(-\lambda\alpha) \right] \times \alpha^n \exp\left(-\sum_{i=1}^n y_i^\alpha \right) \prod_{i=1}^n y_i^{\alpha-1} \mathrm{d}\alpha$$

Given a random sample of α^i, i=1,...,N,from a Gamma(κ, λ) distribution, the integral can be estimated as:

$$\int \pi(\alpha) f(\boldsymbol{y}|\alpha) \mathrm{d}\alpha \approx \frac{1}{N} \sum_{j=1}^N (\alpha^j)^n \exp\left(-\sum_{i=1}^n y_i^{\alpha^j} \right) \prod_{i=1}^n y_i^{\alpha^j - 1} \tag{4}$$

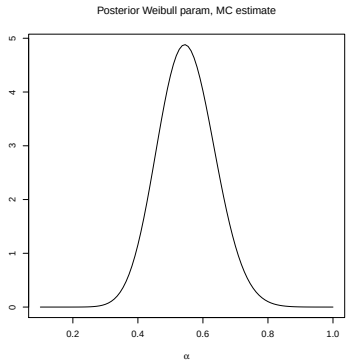
Let the hyperparameters for prior Gamma distribution be κ=λ=0.1. The R code for estimating the normalising constant:

```
  hurricane.gaps <- c(0.30, 4.61, 5.75, 0.24, 0.09, 0.18, 7.38, 1.20, 2.40, 0.18,
                      0.02, 10.07, 0.23, 0.44, 3.34, 0.06, 0.01, 0.71, 0.06, 0.42)
  n <- length(hurricane.gaps)

 #-- Monte Carlo estimate of the normalizing constant with the Weibull --
 # simulating from the prior, Gamma(kappa,lambda)
 set.seed(7301)
 N          <- 1000000
 kappa      <- lambda <- 0.01
 alpha.star <- rgamma(n=N,shape=kappa,rate=lambda)

#--- Calculating the sum by looping over the N samples
 integ.est <- 0
 for(i in 1:N) {
  #integ.est <- integ.est + (1/N)*(alpha.star[i]^n *
  #                                exp(-sum(hurricane.gaps^(alpha.star[i]))) *
  #                                exp(sum((alpha.star[i]-1)*log(hurricane.gaps))))
  integ.est <- integ.est + (1/N)*(exp(n*log(alpha.star[i]) -
                                      sum(hurricane.gaps^(alpha.star[i])) +
                                      sum((alpha.star[i]-1)*log(hurricane.gaps))))
 }
 cat("Estimate of normalising constant=",integ.est,"\n")
 # Estimate of normalising constant= 1.992099e-14
 #   Compare to LN 6 numerical result with pracma: norm constant= 2.008407e-14
```

Additional R code is in Appendix A.1 that also generated an estimate of the posterior distribution and the posterior mean for α, which was 0.5522097. A plot of the estimated posterior distribution is shown in Figure 1.

Important implementation issue: Note how small the normalising constant is. This is not unusual for even moderate size samples such as this, n=20. Computational overflows and underflows are a common problem, e.g., $\prod_{i=1}^n y_i^{\alpha^i - 1}$ may be a number that is too large or too small for the computer. The commands that have been commented out, those preceded by #, carried out the calculation in a manner paralleling equation 4, but led to computational errors:

Figure 1: Posterior distribution for the α parameter in the Weibull example of times between hurricanes based on Monte Carlo integration with samples taken from the prior distribution for α




```
Posterior distribution for the Weibull example of times between hurricanes based on
Estimate of normalising constant= NaN
```

A general solution is to work with logarithms for intermediate calculations and then to exponentiate as late as possible in the calculations. A quote from Gelman et al (BDA):

> To avoid computational overflows and underflows, one should compute with the logarithms of posterior densities whenever possible. Exponentiation should be performed only when necessary and as late as possible; for example, in the Metropolis algorithm, the required ratio of two densities should be computed as the exponential of the difference of the log densities.

2 Direct Sampling

At the heart of the Monte Carlo procedures is the generation of random variables that will be used to either

- Estimate an integral or
- Yield a sample from a desired distribution.

We will call the desired distribution the *target distribution*. For Bayesian inference, the target distribution is often the posterior distribution, $p(\theta|\boldsymbol{y})$, and the integrals of interest can be viewed as expected values:

$$E[h(\theta|\boldsymbol{y})] = \int h(\theta)p(\theta|\boldsymbol{y})dy$$

Common choices for $h(\theta)$, some of which were discussed previously, include:

$$h(\theta) = \theta, \text{ the posterior mean}$$
$$h(\theta) = (\theta - E[\theta])^2, \text{ the posterior variance}$$
$$h(\theta) = I(a \leq \theta \leq b), \text{ the probability that } \theta \text{ is between } a \text{ and } b$$

As seen earlier, if we can generate an independent sample of size N from $p(\theta|\boldsymbol{y})$, θ^1, θ^2, ..., θ^N, we can estimate the expected values:

$$E[h(\theta|\boldsymbol{y})] \approx \hat{E}[h(\theta)|\boldsymbol{y}] = \frac{1}{N}\sum_{i=1}^{N} h(\theta^i)$$

And such estimates are consistent, i.e., they converge to $E[h(\theta|\boldsymbol{y})]$ by the strong law of large numbers.

We will call $\hat{E}[h(\theta)|\boldsymbol{y}] = \frac{1}{N}\sum_{i=1}^{N} h(\theta^i)$ a Monte Carlo estimate of $E[h(\theta|\boldsymbol{y})]$.

Such estimates will not exactly equal $E[h(\theta|\boldsymbol{y})]$, i.e., there will be "Monte Carlo" error:

$$error = \frac{1}{N}\sum_{i=1}^{N} h(\theta^i) - E[h(\theta|\boldsymbol{y})]$$

Comments.

- Given iid data, the Central Limit Theorem can be applied to approximate the sampling distribution of $\hat{E}[h(\theta)|\boldsymbol{y}]$:

  $$\hat{E}[h(\theta)|\boldsymbol{y}] \dot{\sim} \text{Normal}\left(E[h(\theta)|\boldsymbol{y}], \frac{\sigma^2}{N}\right)$$

  where σ^2/N is the Monte Carlo variance, and $\sigma/\sqrt{N}$ is called the Monte Carlo error.

- Assuming that the Monte Carlo estimate is unbiased, σ^2/N can be estimated by the sample variance[1]:

  $$\begin{aligned}\hat{V}\left(\hat{E}[h(\theta)|\boldsymbol{y}]\right) &= \frac{1}{N}\sum_{i=1}^{N}\left(h(\theta^i) - \hat{E}[h(\theta)|\boldsymbol{y}]\right)^2 \\ &= \frac{1}{N}\sum_{i=1}^{N} h(\theta^i)^2 - \left(\frac{1}{N}\sum_{i=1}^{N} h(\theta^i)\right)^2\end{aligned}$$

  The second line is simply an approach to estimating the variance in a single pass through the data.

- The variance can (in theory) be reduced to some arbitrary value by increasing N. (In practice, the size of N to achieve a desired precision might be impractical, e.g., take too long to achieve.)

Example A. $\theta \sim$ from a Gamma(3,0.2), thus $E[\theta]$= 3/0.2=15 (and $V[\theta] = 3/0.04 = 75$). A sample of size N is simulated from this distribution and the sample average is used to estimate $E[\theta]$:

$$\hat{E}[\theta] = \frac{1}{N}\sum_{i=1}^{N} \theta^i$$

By the Central Limit Theorem

$$\hat{E}[\theta] \dot{\sim} \text{Normal}\left(E[\theta] = 15, \frac{\sigma^2}{N} = 75/N\right)$$

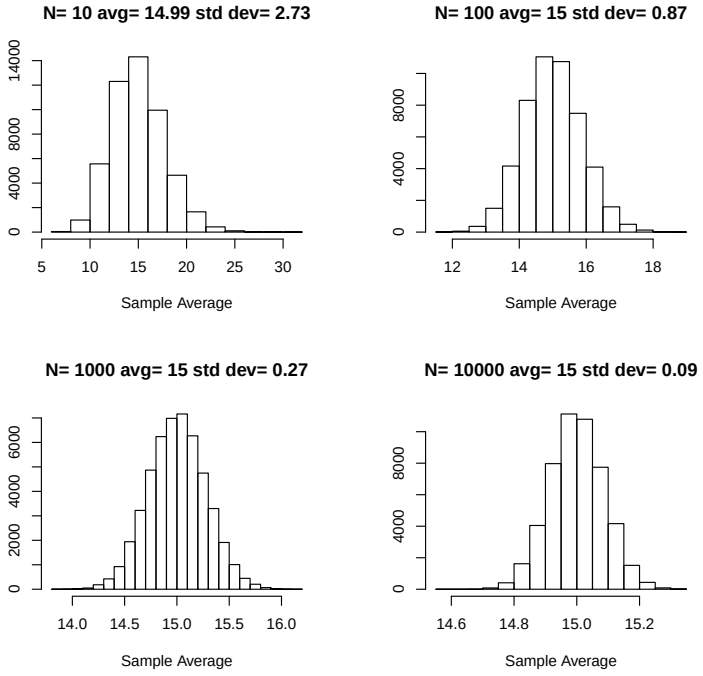
[1]Note: estimates of a population variance, σ^2, with s^2 typically involve division by $N-1$: $s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{y})^2$. With Monte Carlo methods, N is often large enough that division by N instead of $N-1$ is deemed adequate.

The table below shows the average of the $\hat{E}[\theta]$, the empirical standard deviation of $\hat{E}[\theta]$, and the theoretical standard deviation for sample sizes N=10, 100, 1000, and 10000 based upon 50,000 simulations of each sample size.

N	Average $\hat{E}[\theta]$	Std Dev of $\hat{E}[\theta]$	Empirical Std Dev
10	14.992	2.732	2.739
100	14.998	0.866	0.866
1000	15.002	0.274	0.274
10000	15.000	0.086	0.087

Figure 2 shows the distribution of $\hat{E}[\theta]$.

Figure 2: Sampling distribution of $\hat{E}[\theta]$ from a Gamma(3,0.2) for sample sizes N=10, 100, 1000, and 10000 based upon 50,000 simulations of each sample size. True value, $E[\theta]$=15.



Example B. Now we estimate $\Pr(\theta < 5) = E[I(\theta < 5)]$:

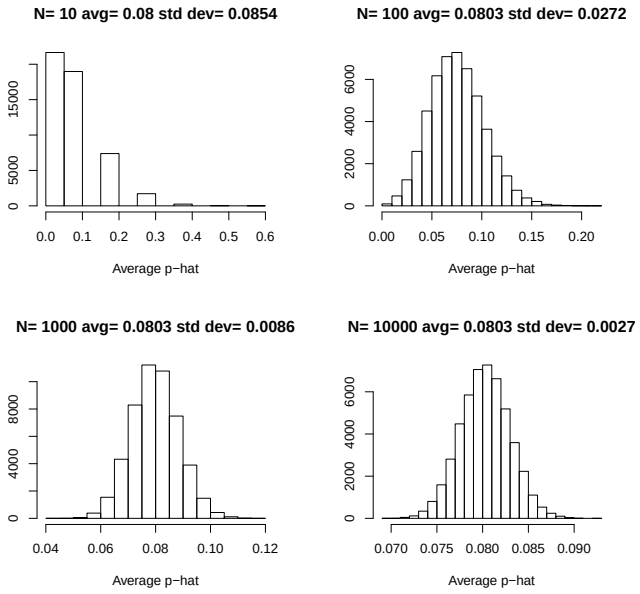
$$\hat{E}[I(\theta < 5)] = \frac{1}{N} \sum_{i=1}^{N} I(\theta^i < 5)$$

where the true value is $pgamma(5, shape = 3, rate = 0.2)$=0.08030, and the theoretical standard error of the estimate is $\sqrt{p * (1-p)/N}$), where p=0.08030. The table below shows the average of the $\hat{E}[\theta]$, the empirical standard deviation of $\hat{E}[\theta]$, and the theoretical standard deviation for sample sizes N=10, 100, 1000, and 10000 based upon 50,000 simulations of each sample size. R code for both examples is in Appendices A.2 and A.3.

N	Average $\hat{E}[I(\theta < 5)]$	Std Dev of $\hat{E}[I(\theta < 5)]$	Empirical Std Dev
10	0.080	0.085	0.086
100	0.080	0.027	0.027
1000	0.080	0.009	0.009
10000	0.080	0.003	0.003

Figure 3 shows the distribution of $\hat{E}[I(\theta < 5)]$.

Figure 3: Sampling distribution of $\hat{\Pr}(\theta < 5) = \hat{E}[I(\theta < 5)]$ from a Gamma(3,0.2) for sample sizes N=10, 100, 1000, and 10000 based upon 50,000 simulations of each sample size. True value, $\Pr(\theta < 5)$=0.08030.



Dependent samples. When the samples generated from $p(\theta|\boldsymbol{y})$ are dependent samples, then:

- For the methods that we will be considering, the Monte Carlo estimates will still be consistent, i.e., they will converge in probability to the expected value—but it's not the standard law of large numbers argument used for independent samples.
- Calculation of the Monte Carlo error is more complicated; e.g., time series methods are sometimes used.

3 Inverse Probability Integral Transform method

For the moment we ignore the particular problem of calculating an integral and consider a general method that is occasionally feasible for simulating random variables from specific probability distributions, the inverse probability integral transform (inverse PIT)[2].

3.1 Continuous random variable

We will focus on univariate random variables and begin with continuous random variables. First recall that given a random variable Y and a function f, we can define new random variable X equal to $f(Y)$. For example, we've demonstrated that with $\theta \sim$ Uniform(0,20) and defining $\phi = g(\theta) = \sqrt{\theta}$.

- Let Y be a continuous random variable with cumulative distribution function, $F_Y(y)$:

$$F_Y(y) = \int_{-\infty}^{y} f(y)dy$$

  where $f(y)$ is the probability density function for Y. Note that for whatever value of y passed to $F_Y(y)$ the value is on [0,1].

- Define a new random variable $U = F_Y(Y)$.

- Claim: $U \sim$ Uniform(0,1) distribution.
  Proof: We prove this by deriving the cumulative distribution function for U:

$$\begin{aligned} F_U(u) = \Pr(U \leq u) &= \Pr(F_Y(Y) \leq u) \\ &= \Pr(Y \leq F_Y^{-1}(u)) \\ &= F_Y(F_Y^{-1}(u)) = u \end{aligned}$$

  which is the CDF for a Uniform(0,1) random variable.

The Inverse PIT method. This result has a corollary that can be used to "go the other way": transform a Uniform random variable using the inverse of the CDF of Y, $g(U) = F_Y^{-1}(U)$, and then the resulting transformed random variable is from the distribution for Y.

Proof: Let $Z = g(U)$, the CDF for Z evaluated at y:

$$\Pr(Z \leq y) = \Pr(F_Y^{-1}(U) \leq y) = \Pr(U \leq F_Y(y)) = F_Y(y)$$

namely the CDF for the random variable Y.

Thus if one can evaluate the inverse cdf for a random variable Y, F^{-1}, one can generate a sample from that distribution by simulating a Uniform(0,1) random variable, u, and setting y=$F^{-1}(u)$.

A simple example is generating a random variable from the Exponential(λ) distribution. The cdf for the exponential:

$$F_Y(y) = \int_0^y \lambda \exp(-\lambda x)dx = 1 - \exp(-\lambda y)$$

[2] R, of course, has built-in functions for many distributions, *runif*, *rnorm*, *rbeta*, *rgamma*, *rweibull*, and the aim here is to introduce an approach that can be useful at times.

Letting $u = F_Y(y)$, the inverse, $F^{-1}(u)$, is found as follows:

$$\begin{aligned} u = F_Y(y) = 1 - \exp(-\lambda y) \quad &\Rightarrow \\ \exp(-\lambda y) = 1 - u \quad &\Rightarrow \\ -\lambda y = \ln(1-u) \quad &\Rightarrow \\ y = -\frac{\ln(1-u)}{\lambda} = F^{-1}(u) & \end{aligned}$$

Thus one can generate a random sample from Exponential(λ) by generating a random sample from Uniform(0,1), u_1^*, u_2^*, ..., u_n^*, and setting $y_i^* = -\frac{\ln(1-u_i^*)}{\lambda}$. Because $1-U$ and U have the same distribution, $y_i^* = -\frac{\ln(u_i^*)}{\lambda}$ can be used.

R code:

```
lambda <- 2
n      <- 100
u      <- runif(n=n,min=0,max=1)
y      <- -log(u)/lambda
```

3.2 Discrete random variable

The inverse PIT method works for discrete random variables as well. The key distinction is how the inverse CDF is defined:

$$F^{-1}(u) = \min_y \{y : F_Y(y) \geq u\}$$

Binomial(2,θ) example. Let $Y \sim$ Binomial(n=2,θ) random variable; thus Y has three possible values, 0, 1, and 2. The CDF, $F_Y(y)$, is a discontinuous function with three jumps:

$$\begin{aligned} F_Y(y) &= 0, y < 0 \\ F_Y(y) &= (1-\theta)^2, 0 \leq y < 1 \\ F_Y(y) &= 1 - \theta^2, 1 \leq y < 2 \\ F_Y(y) &= 1, 2 \leq y \end{aligned}$$

To make things concrete, let θ=0.7. Then

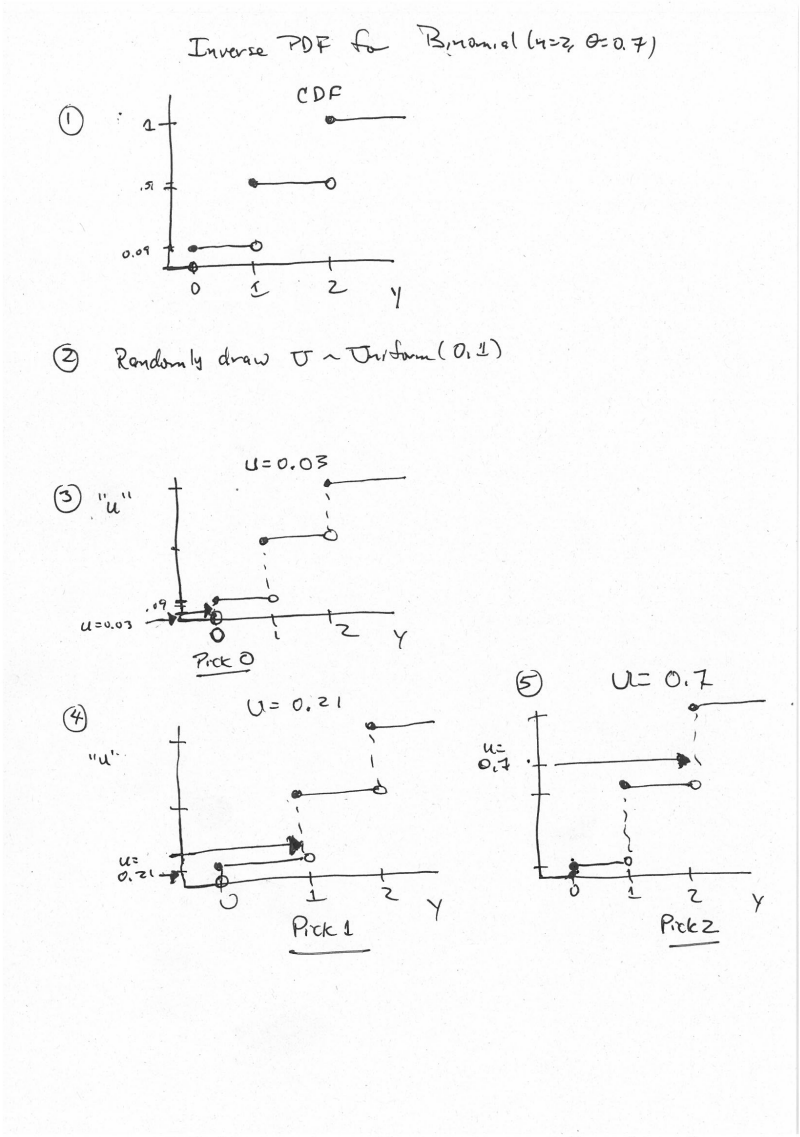
y	$F_Y(y)$
$y < 0$	0
$0 \leq y < 1$	0.09
$1 \leq y < 2$	0.51
$2 \leq y$	1.00

Consider 3 uniform random variable values: 0.03, 0.21, 0.70:

$$\begin{aligned} F^{-1}(u = 0.03) &= \min_y \{y : F_Y(y) \geq 0.03\} = 0 \\ F^{-1}(u = 0.21) &= \min_y \{y : F_Y(y) \geq 0.21\} = 1 \\ F^{-1}(u = 0.70) &= \min_y \{y : F_Y(y) \geq 0.70\} = 2 \end{aligned}$$

Note, for example, that the probability of generating Y=1 is the probability of $0.09 < U \leq 0.51 = 0.42$. Figure 4 demonstrates the results for these 3 uniform draws.

Figure 4: Demonstration of Inverse PIT method for a discrete random variable, Binomial(n=2,θ=0.7).



4 Rejection sampling

Rejection sampling is another Monte Carlo method that generates *independent* samples from the target distribution, e.g., $p(\theta|\boldsymbol{y})$.

- It does so by generating independent samples from another distribution, which we will call the envelope distribution, and denote $g(\theta)$.
- Only some of the generated values are kept or used, while others are discarded: hence the name "rejection" sampling. Thus to produce a sample of size n, one might generate $n+K$ samples from $g(\theta)$ before ending up with n samples.
- It is more often used for generating a univariate random variable than a multivariate random vector.

To reduce notation, the target distribution will be denoted $p(\theta)$ even when it is a posterior distribution.

4.1 Simple demonstration where $p(\theta)$ has bounded support

As a simple demonstration of the main idea, consider a target distribution, $p(\theta)$, where $0 \leq \theta \leq a$, where a is a constant, and the envelope is $Uniform(0, a)$. See Figure 5 for a graphical explanation. Note that two random variables are being generated in a given iteration: one from the envelope and another from a Uniform distribution (different from the envelope in this case).

The proof that the resulting kept value is a sample from the target distribution is shown in Figure 6.

Figure 5: Demonstration of rejection sampling when $p(\theta)$ has support on $(0, a)$ and envelope is Uniform(0,a).

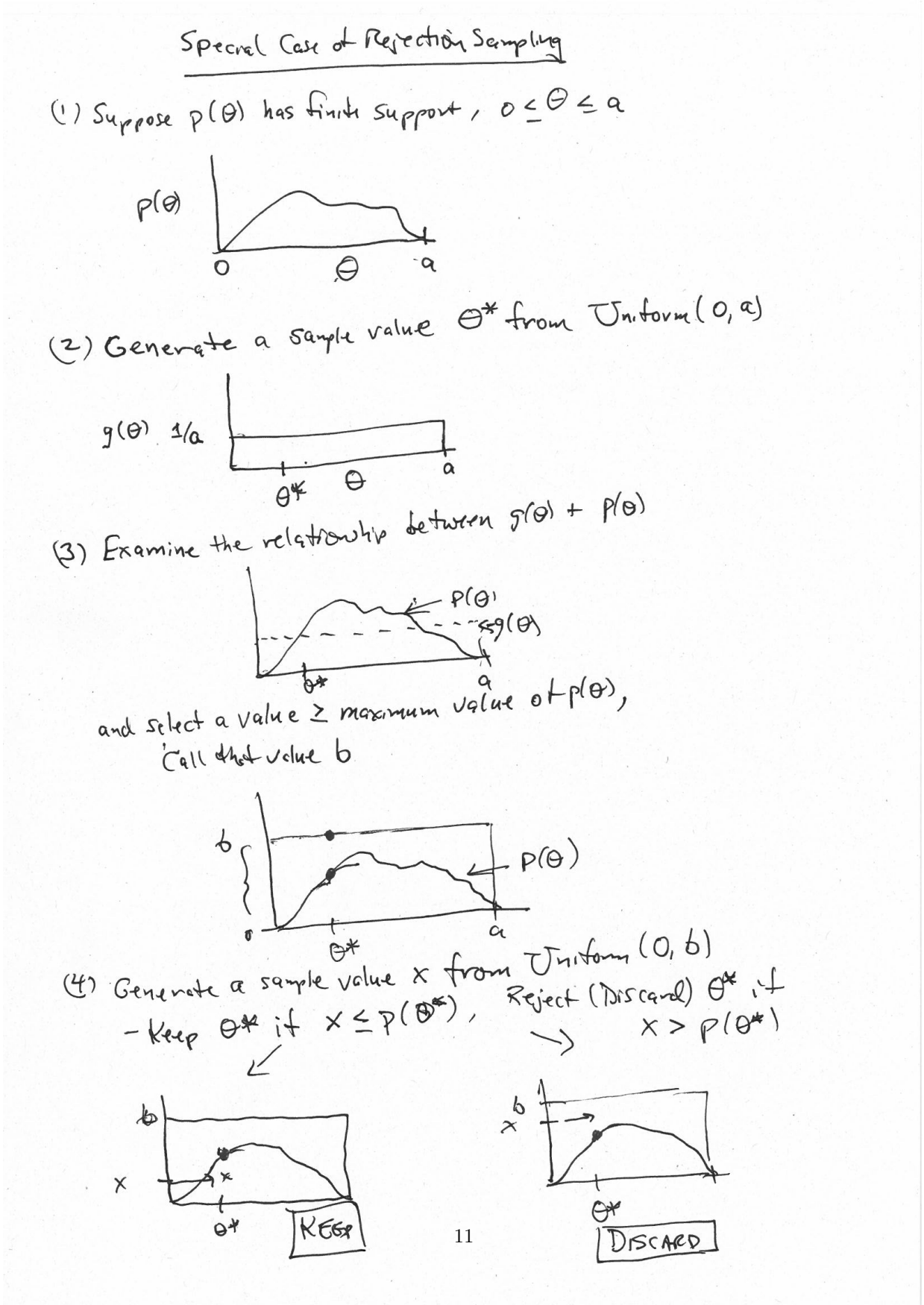


Figure 6: Proof for Figure 5 that the resulting kept value is from $p(\theta)$ (with support on $(0, a)$) given the envelope is Uniform(0,a).

Why this yields a sample from $p(\theta)$:

By Bayes theorem (Conditional probability) - the pdf for Kept Values

$$f(\theta \mid x \le p(\theta)) = \frac{f(\theta, x \le p(\theta))}{f(x \le p(\theta))}$$

$$= \frac{f(x \le p(\theta) \mid \theta)\, g(\theta)}{\int_0^b f(x \le p(\theta) \mid \theta)\, g(\theta)\, d\theta}$$

$$= \frac{\frac{p(\theta)}{b} \cdot \frac{1}{a}}{\int_0^a \frac{p(\theta)}{b} \cdot \frac{1}{a}\, d\theta}$$

$$= \frac{p(\theta)}{\int_0^a p(\theta)\, d\theta}$$

$$= \frac{p(\theta)}{1} = p(\theta)$$

4.2 General rejection sampling algorithm

In general, given any target distribution, $p(\theta)$, which may or may not have finite support, the only constraint on $g(\theta)$ is that its support includes the support of $p(\theta)$, i.e. if $p(\theta) > 0$, then $g(\theta) > 0$. One then finds a bound M on the ratio of $p(\theta)/g(\theta)$:

$$M \geq \frac{p(\theta)}{g(\theta)}, \text{for all } \theta$$

Equivalently:

$$Mg(\theta) \geq p(\theta)$$

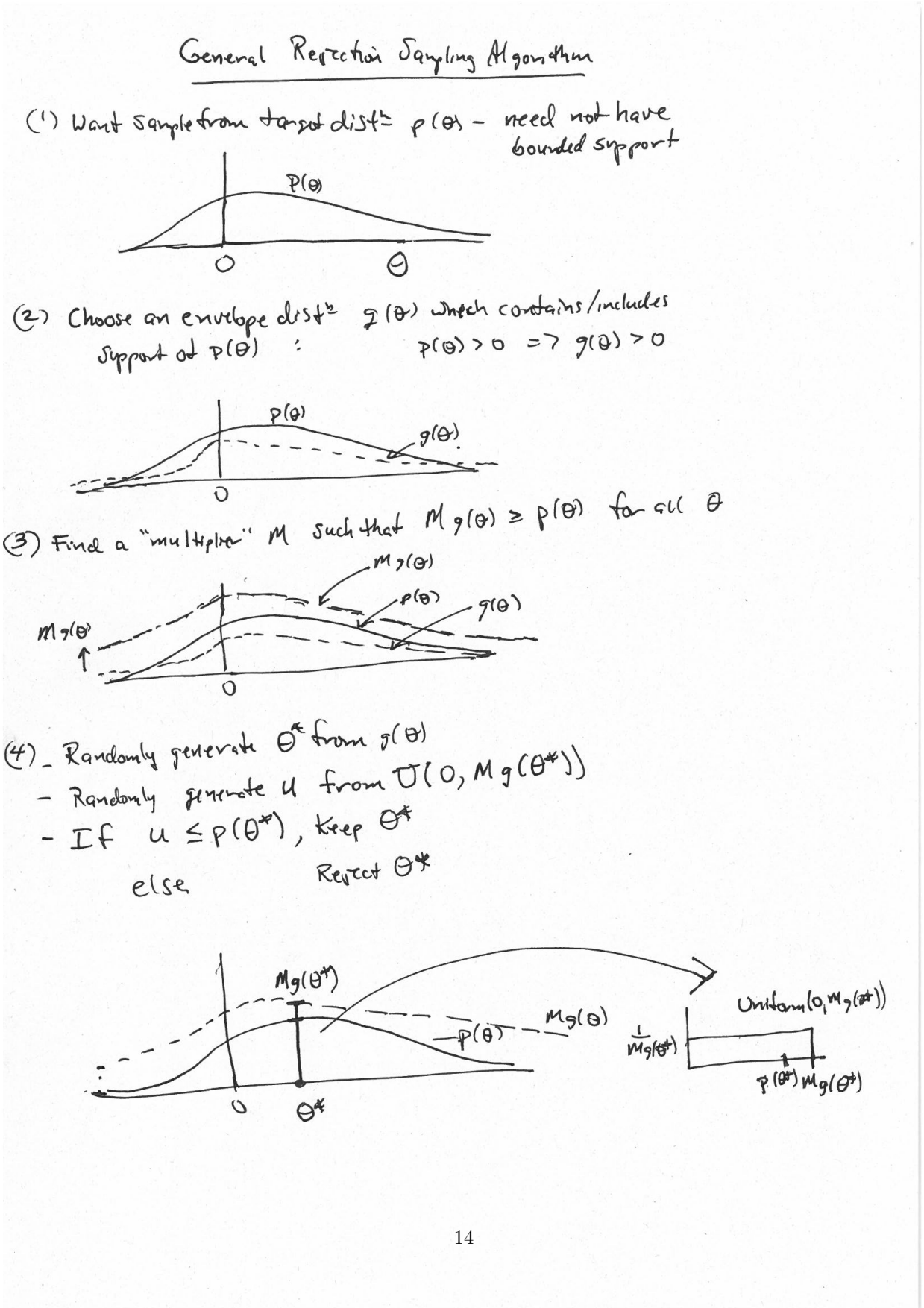
The algorithm to generate a single value from $p(\theta)$.

1. Generate θ^* from $g(\theta)$.
2. Generate u from Uniform$(0, Mg(\theta^*))$.
3. If $u \leq p(\theta^*)$, keep θ^*, else go back to 1.

Variation. Instead of sampling from Uniform$(0, Mg(\theta^*))$ and keeping if $u \leq p(\theta^*)$, one can sample u from Uniform(0,1) and keep if $u \leq p(\theta^*)/Mg(\theta^*)$.

Figure 7 is a graphical representation of the procedure in this general setting.

Figure 7: Demonstration of rejection sampling in the case of arbitrary $p(\theta)$.



4.2.1 Proof that this yields a sample from the target distribution

Conditional on being kept, the pdf for θ is the following.

$$f(\theta|u \le p(\theta)) = \frac{f(\theta, u \le p(\theta))}{f(u \le p(\theta))} = \frac{\Pr(u \le p(\theta)|\theta)g(\theta)}{\int \Pr(u \le p(\theta)|\theta)g(\theta)d\theta}$$
$$= \frac{\frac{p(\theta)}{Mg(\theta)}g(\theta)}{\int \frac{p(\theta)}{Mg(\theta)}g(\theta)d\theta} = \frac{\frac{p(\theta)}{M}}{\int \frac{1}{M}p(\theta)d\theta} = \frac{p(\theta)}{\int p(\theta)d\theta} = p(\theta)$$

4.2.2 The acceptance rate is $1/M$

The probability that θ is kept is $1/M$:

$$\Pr(\theta \text{ is kept}) = \Pr(u \le p(\theta)) = \int \Pr(u \le p(\theta)|\theta)g(\theta)d\theta$$
$$= \int \frac{p(\theta)}{Mg(\theta)}g(\theta)d\theta = \frac{1}{M}\int p(\theta)d\theta = \frac{1}{M}$$

Thus for a high acceptance rate, want $1/M$ close to 1, thus M to be relatively small, e.g., a little larger than 1. (Can't have $M < 1$.)

Thus ideally, select:

$$M_{opt} = \sup\left\{\frac{p(\theta)}{g(\theta)}\right\}$$

Namely, find the Least Upper Bound of the ratio of the target to the envelope.

In some cases for some target distribution and some envelope distribution, can find M_{opt}, but not always.

4.2.3 Example C

The target distribution, $p(\theta)$, Beta$(3, 2)$, and the envelope distribution is $Uniform(0, 1)$. To find a "suitable" M (actually an optimal M), examine the ratio $p(\theta)/g(\theta)$:

$$\frac{p(\theta)}{g(\theta)} = \frac{\Gamma(5)}{\Gamma(3)\Gamma(2)}\theta^{3-1}(1-\theta)^{2-1} = 12\theta^2(1-\theta)$$

Find a critical point by differentiating $\ln(12\theta^2(1-\theta)) \propto 2\ln(\theta) + \ln(1-\theta)$, setting equal to 0 and solving for θ. The result is θ=2/3 is a critical point (and 2nd derivative test shows that it yields a maximum). And

$$\sup_\theta \frac{p(\theta)}{g(\theta)} = 12 * (2/3)^2 * (1/3) = 16/9$$

Then set M=16/9. Note the acceptance rate will be $1/M = 9/16 = 0.5625$.

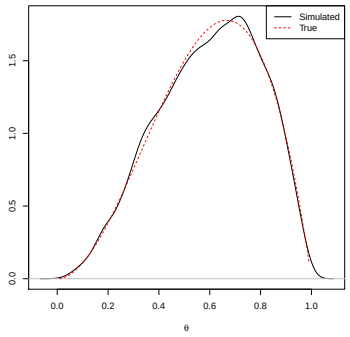
Demonstration with R:

```
#- Rejection sampling with target Beta(3,2) and envelope U(0,1)
a          <- 3; b          <- 2
n          <- 10000
```

```
M.opt      <- 16/9
theta.sim  <- numeric(n)
total.sims <- 0
tally      <- 0
while(tally < n) {
  total.sims <- total.sims+1
  theta.star <- runif(n=1,min=0,max=1)
  u.star     <- runif(n=1,min=0,max=M.opt*1)
  if(u.star <= dbeta(theta.star,a,b)) {
     tally <- tally+1
     theta.sim[tally] <- theta.star=
  }
}
accept.rate <- n/total.sims
cat("acceptance rate=",accept.rate,"Theory=",1/M.opt,"\n")
# acceptance rate= 0.5617978 Theory= 0.5625
```

Note that the empirical acceptance rate, 0.5618, was very close to the expected rate of 9/16=0.5625. Figure 8 compares the true Beta(3,2) density to the empirical density based on the rejection sample.

Figure 8: Demonstration of rejection sampling target Beta(3,2) and envelope U(0,1).



4.3 Target density need only be known up to a proportionality constant

The target density $p(\theta)$ need only be evaluated up to a proportionality constant, i.e., just need to be able to evaluate the terms in $p(\theta)$ that include θ. For example, if θ is Beta(α, β), need only evaluate $\theta^{\alpha-1}(1-\theta)^{\beta-1}$, not $\frac{\Gamma(\alpha+\beta}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$.

The algorithm is essentially the same as before. Write $p(\theta)$ as $p_1(\theta)c$, where c is the proportionality constant. Find M_1 such that

$$M_1 \geq \frac{p_1(\theta)}{g(\theta)}$$

Then

- Generate θ^* from $g(\theta)$.

- Generate u from Uniform(0,$M_1 g(\theta^*)$).
- Keep θ^* if $u < p_1(\theta^*)$

The proof is just like for the case with $p(\theta)$. Conditional on being kept, the pdf for θ is the following.

$$\begin{aligned} f(\theta|u \le p_1(\theta)) &= \frac{f(\theta, u \le p_1(\theta))}{f(u \le p(\theta))} = \frac{\Pr(u \le p_1(\theta)|\theta)g(\theta)}{\int \Pr(u \le p_1(\theta)|\theta)g(\theta)d\theta} \\ &= \frac{\frac{p_1(\theta)}{M_1 g(\theta)}g(\theta)}{\int \frac{p_1(\theta)}{M_1 g(\theta)}g(\theta)d\theta} = \frac{\frac{p_1(\theta)c}{M_1 g(\theta)}g(\theta)}{\int \frac{p_1(\theta)c}{M_1 g(\theta)}g(\theta)d\theta} \\ &= \frac{\frac{p(\theta)}{M_1}}{\int \frac{1}{M_1}p(\theta)d\theta} = \frac{p(\theta)}{\int p(\theta)d\theta} = p(\theta) \end{aligned}$$

What has happened here is that $M_1 \equiv M/c$. Note also that the acceptance probability is now $1/Mc$.

4.4 Example D

Not needing to evaluate the normalising constant, $m(\boldsymbol{y})$, is what makes rejection sampling very attractive for Bayesian inference.

For example (from Givens and Hoeting, 2013), a sample of size n=10 is generated from a Poisson(θ) distribution. The observed sample is:

$$8, 3, 4, 3, 1, 7, 2, 6, 2, 7$$

and $\bar{y} = 4.3$. A lognormal prior distribution is assumed for θ: $\theta \sim$ Lognormal(log(5), 0.5^2).

The objective is to generate a sample from the posterior distribution which is proportional to the Poisson likelihood and the lognormal prior:

$$\pi(\theta)f(\boldsymbol{y}|\theta) = \left[\frac{1}{\sqrt{2\pi 0.5^2}}\frac{1}{\theta}e^{-\frac{1}{2*0.5^2}(\ln(\theta)-\log(5))^2}\right] \times \left[\frac{\exp(-n\theta)\theta^{n\bar{y}}}{\prod_{i=1}^n y_i!}\right]$$

which will have a complicated normalising constant.

For a given envelope distribution, need to find M_1 such that:

$$M_1 \ge \sup_\theta \frac{\pi(\theta)f(\boldsymbol{y}|\theta)}{g(\theta)}$$

A convenient envelope distribution, $g(\theta)$, is the prior (not necessarily the most efficient however) as that makes calculation of M_1 easier:

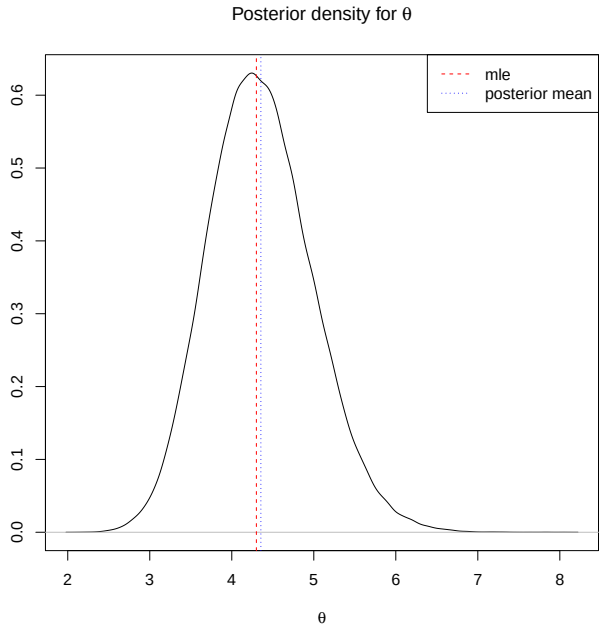
$$M_1 \ge \sup_\theta \frac{\pi(\theta)f(\boldsymbol{y}|\theta)}{\pi(\theta)} = \max_\theta f(\boldsymbol{y}|\theta)$$

The largest value of the likelihood function, $f(\boldsymbol{y}|\theta)$, is at the mle ($\hat{\theta}$), in this case the sample mean, 4.3. Thus

$$M_1 = \frac{\exp(-n\hat{\theta})\hat{\theta}^{n\bar{y}}}{\prod_{i=1}^n y_i!} = 1.439438e-10$$

The R code is shown below and Figure 9 shows the posterior density for θ. The estimate of the posterior mean is 4.355 and the estimate of the posterior variance is 0.398. The acceptance was about 28%.

Figure 9: Rejection Sampling: Posterior distribution for Poisson parameter θ given a Lognormal prior.




```
set.seed(214)
y <- c(8, 3, 4, 3, 1, 7, 2, 6, 2, 7)
n <- length(y)
mle <- mean(y)
M1 <- prod(dpois(y,lambda=mle))

log.mu <- log(5)
log.sd <- 0.5
N         <- 100000
M.opt      <- M1
theta.sim  <- numeric(n)
total.sims <- 0
tally      <- 0
while(tally < N) {
  total.sims <- total.sims+1
  theta.star <- rlnorm(n=1,meanlog = log.mu, sdlog=log.sd)
  #due to cancellations, g.theta need not be calculated,
  # but this is just to make the ratio calculations explicit
  g.theta    <- dlnorm(theta.star, meanlog=log.mu, sdlog=log.sd)
  u.star     <- runif(n=1,min=0,max=M.opt*g.theta)
  if(u.star <= prod(dpois(y,lambda=theta.star))*g.theta) {
    tally <- tally+1
    theta.sim[tally] <- theta.star
  }
}
cat("Estimated posterior mean=",mean(theta.sim),"posterior variance=",var(theta.sim),"\n")
#Estimated posterior mean= 4.356888 posterior variance= 0.395814
```

4.5 Advantages and Disadvantages of Rejection Sampling

Advantages:

1. It's a very general method that can be used for any distribution (so long as the target distribution, $p(\theta)$, can be calculated).
2. Relatively easy to implement.
3. The distribution only need to be evaluated to a constant of proportionality. This is most advantageous for Bayesian inference for $p(\theta|\boldsymbol{y})$—don't need $m(\boldsymbol{y})$.
4. If can find an envelope distribution, $g(\theta)$, where M is small, can be very efficient.

Disadvantages (or difficulties):

1. If for the chosen envelope, $g(\theta)$, M is large, the rejection rate can be quite high, thus inefficient.
2. Can be difficult to find a good envelope, $g(\theta)$, even for just three or four dimensions, e.g., $p(\theta_1, \theta_2, \theta_3|\boldsymbol{y})$.

Main point: if $g(\theta)$ is roughly proportional to the target distribution $p(\theta)$, has similar shape, then rejection sampling can be quite efficient.

4.6 Two refinements on rejection sampling

Reference: Givens and Hoeting, Computational Statistics, 2nd Ed., 2013.

One refinement on rejection sampling is *Squeezing*: the essence of the idea is "enclose" the target distribution, $p(\theta)$, above by the envelope function $Mg(\theta)$ and below by one or more simple functions, e.g., a constant function. See top graph in Figure 10.

Another refinement, similar to squeezing is *Adaptive Rejection Sampling*. The essence of ARC is to define envelope and squeezing functions that are tangents and chords, respectively, to the target distribution. The tangents and chords get updated when generated values fall outside of the envelope and the squeezing functions. See bottom graph in Figure 10.

Figure 10: (From Givens and Hoeting, Computational Statistics, 2013.) Demonstration of "squeezing" (top figure) and Adaptive Rejection Sampling (ARS) (bottom figure).

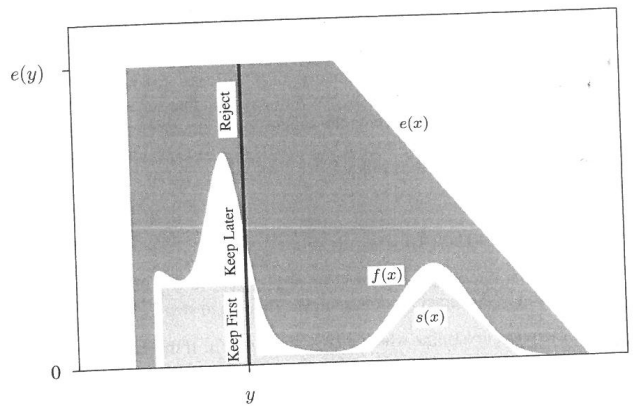


Fig. 6.3 Illustration of squeezed rejection sampling for a target distribution, f, using envelope e and squeezing function s. 'Keep first' and 'Keep later' correspond to steps 3 and 4 of the algorithm, respectively.

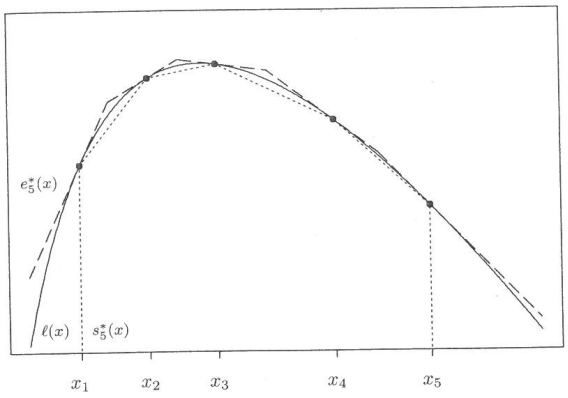


Fig. 6.4 Piecewise linear outer and inner hulls for $\ell(x) = \log f(x)$ used in adaptive rejection sampling when $k = 5$.

5 Importance Sampling

5.1 Basic idea

Similar to Rejection Sampling, Importance Sampling is a procedure

- where an *independent* sample of θ is generated from the *wrong* distribution, an importance sampling or envelope distribution, $g(\theta)$, but then the values are "adjusted" to correspond to the "right" distribution, the target distribution, $p(\theta)$ or $p(\theta|\boldsymbol{y})$ for Bayesian applications.

Importance Sampling differs from Rejection Sampling, however, in a crucial way: All the generated θ are kept—there is no rejection.

While importance sampling can be used to generate a sample directly from a target distribution, its utility for estimating integrals is perhaps more apparent. Suppose the objective is to estimate the following integral

$$E_p(h(\theta|\boldsymbol{y})) = \int h(\theta)p(\theta|\boldsymbol{y})\mathrm{d}\theta$$

where $p(\theta|\boldsymbol{y})$ is a posterior distribution and the subscript p has been added to E_p to emphasise the probability distribution $p(\theta|\boldsymbol{y})$.

Let $g(\theta)$ be another distribution that has the same support as $p(\theta|\boldsymbol{y})$[3]. Then the integral can be rewritten:

$$\begin{aligned} E_p(h(\theta|\boldsymbol{y})) &= \int h(\theta)p(\theta|\boldsymbol{y})\frac{g(\theta)}{g(\theta)}d\theta = \int h(\theta)\frac{p(\theta|\boldsymbol{y})}{g(\theta)}g(\theta)\mathrm{d}\theta \\ &= \int h(\theta)w(\theta)g(\theta)\mathrm{d}\theta = E_g\left(h(\theta)w(\theta)\right) \end{aligned}$$

where $w(\theta) = p(\theta|\boldsymbol{y})/g(\theta)$ is the importance ratio. The subscript g in E_g is emphasising the probability distribution $g(\theta)$.

If one can generate a (iid) sample of size N from $g(\theta)$, then a Monte Carlo estimate of $E_g(w(\theta))$ can be calculated:

$$\hat{E}_g(h(\theta)w(\theta)) \equiv \hat{E}_p(h(\theta|\boldsymbol{y})) = \frac{1}{N}\sum_{i=1}^{N} h(\theta^i)w(\theta^i)$$

Again by the Strong Law of Large Numbers, the Monte Carlo estimate will converge with probability 1 to $E_g(h(\theta)w(\theta))$, which as showed above, is also $E_p(h(\theta|\boldsymbol{y}))$.

Comments.

1. The Monte Carlo error in the estimated integral is a function of the distribution of weights (or importance ratios). The expected value of importance ratios is 1:

$$E[p(\theta|\boldsymbol{y})/g(\theta)] = \int \frac{p(\theta|\boldsymbol{y})}{g(\theta)}g(\theta)\mathrm{d}\theta = \int p(\theta|\boldsymbol{y})\mathrm{d}\theta = 1$$

[3]Technically $g(\theta)$ just needs to have the same support as the product, $h(\theta)p(\theta|\boldsymbol{y})$, not the posterior distribution alone. For example, if $h(\theta) = I(\theta < 5)$, then the support of $g(\theta)$ just needs to include $\theta < 5$.

But the variation in these ratios is what causes Monte Carlo error.

$$V[p(\theta|\boldsymbol{y})/g(\theta)] = \int \left(\frac{p(\theta|\boldsymbol{y})}{g(\theta)} - 1\right)^2 g(\theta)d\theta$$

Very large weights, e.g., $p(\theta^i|\boldsymbol{y})/g(\theta^i) >> 1$, can result when g is "light" in the tails of the distribution of $p(\theta|\boldsymbol{y})$. A common preference for $g(\theta)$ is a distribution with relatively "heavy" tails and t and multivariate t distributions are popular choices.

2. Similar to Rejection Sampling, the Monte Carlo error decreases as the distributional shape of importance sampler becomes more similar to that of the integrand, $h(\theta)p(\theta|\boldsymbol{y})$.

3. Also similar to Rejection Sampling, the target distribution, or the integrand, need only be known up to a constant of proportionality. To begin the expected value is written as a ratio of two integrals:

$$E_p(h(\theta|\boldsymbol{y})) = \int h(\theta)p(\theta|\boldsymbol{y})\mathrm{d}\theta = \int h(\theta)\frac{\pi(\theta)f(\boldsymbol{y}|\theta)}{m(\boldsymbol{y})}\mathrm{d}\theta = \frac{\int h(\theta)\pi(\theta)f(\boldsymbol{y}|\theta)\mathrm{d}\theta}{\int \pi(\theta)f(\boldsymbol{y}|\theta)\mathrm{d}\theta}$$

Use importance sampling to estimate the integrals in the numerator and denominator:

$$\frac{1}{N}\sum_{i=1}^{N} h(\theta^i)\frac{\pi(\theta^i)f(\boldsymbol{y}|\theta^i)}{g(\theta^i)} \approx \int h(\theta)\pi(\theta)f(\boldsymbol{y}|\theta)\mathrm{d}\theta \tag{5}$$

$$\frac{1}{N}\sum_{i=1}^{N} \frac{\pi(\theta^i)f(\boldsymbol{y}|\theta^i)}{g(\theta^i)} \approx \int \pi(\theta)f(\boldsymbol{y}|\theta)\mathrm{d}\theta = m(\boldsymbol{y}) \tag{6}$$

Then

$$\hat{E}_p(h(\theta|\boldsymbol{y})) = \frac{\sum_{i=1}^{N} h(\theta^i)\frac{\pi(\theta^i)f(\boldsymbol{y}|\theta^i)}{g(\theta^i)}}{\sum_{i=1}^{N}\frac{\pi(\theta^i)f(\boldsymbol{y}|\theta^i)}{g(\theta^i)}} = \sum_{i=1}^{N} h(\theta^i)w^{*i}$$

where

$$w^{*i} = \frac{\frac{\pi(\theta^i)f(\boldsymbol{y}|\theta^i)}{g(\theta^i)}}{\sum_{j=1}^{N}\frac{\pi(\theta^j)f(\boldsymbol{y}|\theta^j)}{g(\theta^j)}}$$

The estimator has some bias, but is consistent (asymptotically unbiased).

5.2 Example E: Poisson-Lognormal case again

Importance sampling was applied to the previous Poisson data with lognormal distribution. The prior was used as the envelope distribution (also called importance sampler), thus the importance ratio:

$$w^i = \frac{\pi(\theta^i)f(\boldsymbol{y}|\theta^i)}{\pi(\theta^i)} = f(\boldsymbol{y}|\theta^i) \propto e^{-n*\theta^i}(\theta^i)^{n\bar{y}}$$

The weights were then scaled to cancel the normalising constant:

$$w^{*i} = \frac{w^i}{\sum_{j=1}^{N} w^j}$$

And the posterior mean and variance were estimated as:

$$\hat{E}(\theta) = \sum_{i=1}^{N} w^{*i}\theta^i$$

$$\hat{V}(\theta) = \sum_{i=1}^{N} w^{*i}(\theta^i - \hat{E}(\theta))^2$$

The R code is shown below. The estimate of the posterior mean is 4.359 and variance is 0.400, quite similar to the rejection sampling estimates of 4.355 and 0.398, respectively.

```
#--- Importance sampling to estimate the mean and variance of the Poisson-Lognormal posterior
y <- c(8, 3, 4, 3, 1, 7, 2, 6, 2, 7)
n <- length(y)
ybar <- mean(y)
set.seed(381)
N <- 1000000
theta.star <- rlnorm(n=N,meanlog = log.mu, sdlog=log.sd)
importance.ratio <- exp(-n*theta.star)*theta.star^(n*ybar)
weight.star  <- importance.ratio/sum(importance.ratio)
hat.E.theta <- sum(theta.star*weight.star)
hat.V.theta <- sum((theta.star-hat.E.theta)^2*weight.star)
alt.V       <- sum(theta.star^2*weight.star)-hat.E.theta^2

cat("Estimated posterior mean=",hat.E.theta,"\n")
# Estimated posterior mean= 4.358722
cat("Estimated posterior variance=",hat.V.theta,"\n")
# Estimated posterior variance= 0.3996452
cat("alt variance=",alt.V,"\n")
#alt variance= 0.3996452
```

6 Sampling Importance Re-Sampling

Importance sampling can be seen as a means to estimate integrals, but the generated θ and the importance ratios, w^i, can be used to produce *an approximate sample from the target distribution*, e.g., $p(\theta|\boldsymbol{y})$. To generate a sample of size n:

1. Choosing $N > n$, generate N samples of θ^i, i=1,...,N, from the envelope distribution (importance sampler), $g(\theta)$.
2. Calculate the importance ratios, w^i, and scale them by the sum of the weights:

$$w^{*i} = \frac{w^i}{\sum_{j=1}^{N} w^j}$$

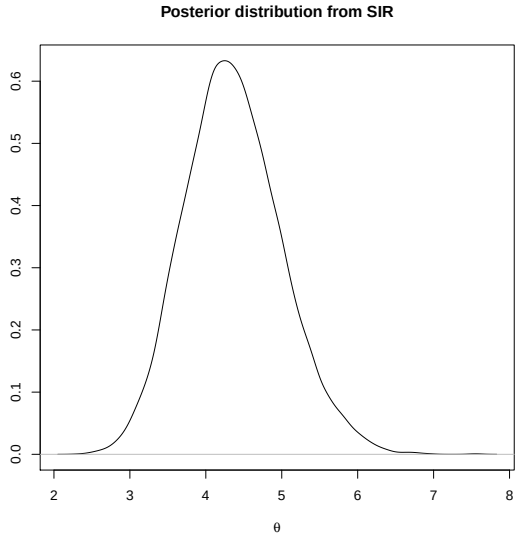
   Thus $0 < w^{*i} < 1$ and $\sum_{j=1}^{N} w^{*j} = 1$.
3. Randomly sample n θ^i, with replacement, with probabilities w^{*1}, w^{*2}, ..., w^{*N}.

Note that for Bayesian inference when the target distribution is the posterior, due to the scaling of the importance ratios, the normalising constant is cancelled out and does not need to be known.

Example F: SIR with the Poisson-Lognormal example. The scaled importance weights from the Importance Sampling example above were used for the SIR sampling, and the estimated posterior density for θ is plotted in Figure 11, which looks much like the result from the Rejection Sampling (Figure 9).

```
#--- SIR sample of the posterior ----
set.seed(9371)
n <- 10000
theta.post.sample <- sample(theta.star,size=n,prob=weight.star,replace=TRUE)
plot(density(theta.post.sample),type="l",xlab=expression(theta),ylab="",
     main="Posterior distribution from SIR")
```

Figure 11: Estimated posterior density for θ from the Poisson-Lognormal example based a SIR sampling.



A R Code

A.1 Monte Carlo inference for the Weibull example.

```
 hurricane.gaps <- c(0.30, 4.61, 5.75, 0.24, 0.09, 0.18, 7.38, 1.20, 2.40, 0.18,
                     0.02, 10.07, 0.23, 0.44, 3.34, 0.06, 0.01, 0.71, 0.06, 0.42)
 n <- length(hurricane.gaps)

#-- Monte Carlo estimate of the normalizing constant with the Weibull --
# simulating from the prior, Gamma(kappa,lambda)
set.seed(730)
N          <- 100000
kappa      <- lambda <- 0.01
alpha.star <- rgamma(n=N,shape=kappa,rate=lambda)

#--- Calculating the sum by looping over the N samples
integ.est <- 0
natural.approach <- FALSE
for(i in 1:N) {
  if(natural.approach) {
    integ.est <- integ.est + (1/N)*(alpha.star[i]^n *
                                    exp(-sum(hurricane.gaps^(alpha.star[i]))) *
                                    exp(sum((alpha.star[i]-1)*log(hurricane.gaps))))
  } else {
    integ.est <- integ.est + (1/N)*(exp(n*log(alpha.star[i]) -
                                      sum(hurricane.gaps^(alpha.star[i])) +
                                      sum((alpha.star[i]-1)*log(hurricane.gaps))))
  }
}
cat("Estimate of normalising constant=",integ.est,"\n")
# Estimate of normalising constant= 1.864341e-14

#--- sample from posterior distribution....
alpha.seq <- seq(0.1,1,length=100)
mc.density <- numeric(length(alpha.seq))
for(i in 1:length(alpha.seq)) {
  mc.density[i] <- dgamma(alpha.seq[i],shape=kappa,rate=lambda)*
    prod(dweibull(x=hurricane.gaps,shape= alpha.seq[i],scale=1))/integ.est
}
plot(alpha.seq,mc.density,type="l",xlab=expression(alpha),ylab="",
    main=expression("Posterior Weibull param, MC estimate"))

#-- estimating expected value for posterior, can use the previously generate alpha.star
alpha.star.weights <- numeric(length(alpha.star))
for(i in 1:length(alpha.star)) {
  alpha.star.weights[i] <- prod(dweibull(x=hurricane.gaps,shape=alpha.star[i],scale=1))/integ.est
}

#--   problem with NAs when alpha.star=0 or is too large, >300,..."cheating"
ok <- !is.na(alpha.star.weights)
ok.N <- sum(ok)
# post.alpha.mean <- (1/N)* sum(alpha.star*alpha.star.weights,na.rm=TRUE)

post.alpha.mean <- (1/ok.N)* sum(alpha.star[ok]*alpha.star.weights[ok])
cat("Est of posterior mean=",post.alpha.mean,"\n")
# Est of posterior mean= 0.5522097
```

A.2 Demonstration of Monte Carlo error in estimation of $E[\theta]$ where $\theta \sim$ Gamma(3,0.2).

```
#--- Monte Carlo error demonstration---- simulating from a Gamma and estimating expected value
set.seed(530)
number.simulations <- 50000
N.vec              <- c(10,100,1000,10000)
avg.averages       <- numeric(4)
sd.averages        <- numeric(4)
a      <- 3;  b    <- 0.2
true.ExpVal <- a/b
cat("Expected value=",true.ExpVal,"\n")
# Expected value= 15
true.stderr <- sqrt(a/b^2)/sqrt(N.vec)
cat("Theoretical Std Errors",round(true.stderr,3),"\n")
```

```
par(mfrow=c(2,2),oma=c(0,0,3,0))
for(i in 1:4) {
  out <- matrix(rgamma(N.vec[i]*number.simulations,shape=a,rate=b),
                nrow=number.simulations,ncol=N.vec[i])
  per.sample.average <- apply(out,1,mean)
  avg.averages[i] <- mean(per.sample.average)
  sd.averages[i]  <- sd(per.sample.average)
  hist(per.sample.average,xlab="Sample Average",ylab="",
       main=paste("N=",N.vec[i],"avg=",round(avg.averages[i],2),
             "std dev=",round(sd.averages[i],2)))
}
par(mfrow=c(1,1))
```

A.3 Demonstration of Monte Carlo error in estimation of $Pr(\theta < 5)$ where $\theta \sim$ Gamma(3,0.2).

```
# simulating from a Gamma and estimating Pr(theta < 5)
set.seed(530)
number.simulations <- 50000
N.vec              <- c(10,100,1000,10000)
avg.averages       <- numeric(4)
sd.averages        <- numeric(4)
a      <- 3;  b    <- 0.2
true.prob <- pgamma(5,shape=a,rate=b)
cat("Pr(theta<5)=",true.prob,"\n")
# Pr(theta<5)= 0.0803014
true.stderr <- sqrt(true.prob*(1-true.prob))/sqrt(N.vec)
cat("Theoretical Std Errors",round(true.stderr,3),"\n")
#Theoretical Std Errors 0.086 0.027 0.009 0.003

indicator.fun <- function(x,y) x<y

par(mfrow=c(2,2),oma=c(0,0,3,0))
for(i in 1:4) {
  out <- matrix(rgamma(N.vec[i]*number.simulations,shape=a,rate=b),
                nrow=number.simulations,ncol=N.vec[i])
  out <- indicator.fun(out,5)
  per.sample.est <- apply(out,1,mean)
  avg.averages[i] <- mean(per.sample.est)
  sd.averages[i]  <- sd(per.sample.est)
  hist(per.sample.est,xlab="Average p-hat",ylab="",
       main=paste("N=",N.vec[i],"avg=",round(avg.averages[i],4),
                  "std dev=",round(sd.averages[i],4)))
}
par(mfrow=c(1,1))
```

November 10, 2020