Generating Random Variates

Max Turgeon

STAT 3150–Statistical Computing

Lecture Objectives

- Recognize when to use the inverse-transform method.
- Be able to generate random variates through transformations.
- Derive bounding densities for accept-reject sampling.

Motivation

- A staple of modern statistical research is the **simulation study**.
  - Finite sample properties can then be compared to theoretical expectations.
- More generally, by simulating data we can study the properties of a method or a model.
- **Bayesian statistics** strongly relies on generating data to estimate the posterior density of the parameters (cf. STAT 4150).

Inverse-Transform Method i

- *Recall*: Let X be a random variable with CDF $F(x)$. The **quantile** function is defined as

$$F^{-1}(p) = \inf\{x \in \mathbb{R} \mid F(x) \geq p\}.$$

- If X is continuous, this is simply the inverse function.

Inverse-Transform Method ii

Theorem

If U is uniform on $[0, 1]$, then $F^{-1}(U)$ has the same distribution as X.

- In R, we can sample random variates from $U(0, 1)$ by using the function `runif`:

```r
runif(5)
```

```
## [1] 0.2681359 0.3308333 0.4411671 0.8352923
0.9690489
```

Inverse-Transform Method iii

Algorithm

To generate random variates from F:

1. Generate random variates from $U(0, 1)$.
2. Compute the quantile function F^{-1}.
3. Plug-in the uniform variates into F^{-1}.

Example i

- Let X follow an exponential distribution with parameter λ:

$$F(x) = 1 - \exp(-\lambda x).$$

- Since X is continuous, the quantile function is the inverse of F:

$$\begin{aligned} p = 1 - \exp(-\lambda x) &\Rightarrow \exp(-\lambda x) = 1 - p \\ &\Rightarrow -\lambda x = \log(1 - p) \\ &\Rightarrow x = \frac{-\log(1-p)}{\lambda}. \end{aligned}$$

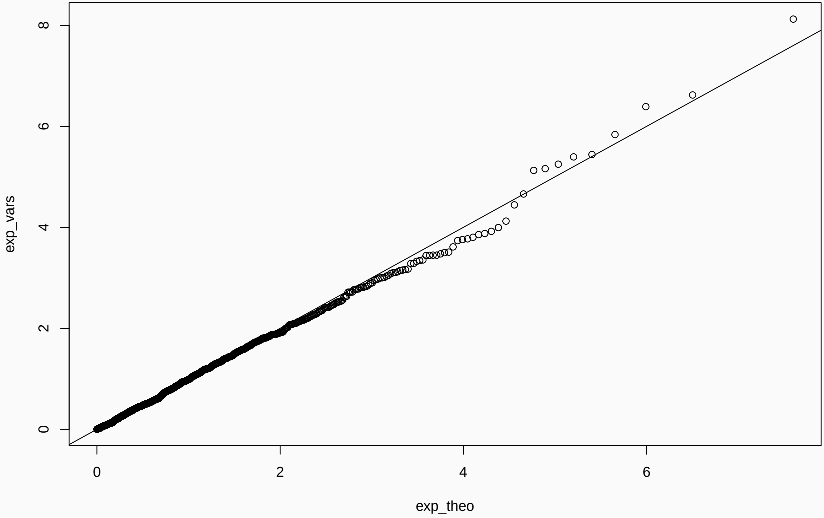
Example ii

```r
lambda <- 1
# We want 1000 samples
n <- 1000
unif_vars <- runif(n)
exp_vars <- -log(1 - unif_vars)/lambda
```

Example iii

```
# Compute theoretical quantiles
# using qexp
exp_theo <- qexp(ppoints(n))
qqplot(exp_theo, exp_vars)
# Add diagonal line
abline(a = 0, b = 1)
```

Example iv



Example v

Note: If U is uniform on $[0, 1]$, so is $1 - U$.

- Therefore $\frac{-\log(U)}{\lambda}$ also follows an $Exp(\lambda)$ distribution.

Exercise

Compute the quantile function for the Cauchy distribution $\text{Cauchy}(\theta, \gamma)$ with CDF

$$F(x) = \frac{1}{\pi}\arctan\left(\frac{x-\theta}{\gamma}\right) + \frac{1}{2}.$$

Use the inverse transform to generate 5 random variates from $\text{Cauchy}(0, 1)$.

Solution i

$$
\begin{aligned}
p = \frac{1}{\pi}\text{arctan}\left(\frac{x-\theta}{\gamma}\right) + \frac{1}{2} &\Rightarrow \pi(p-0.5) = \text{arctan}\left(\frac{x-\theta}{\gamma}\right) \\
&\Rightarrow \tan\left(\pi(p-0.5)\right) = \frac{x-\theta}{\gamma} \\
&\Rightarrow \gamma\tan\left(\pi(p-0.5)\right) = x-\theta \\
&\Rightarrow x = \gamma\tan\left(\pi(p-0.5)\right) + \theta.
\end{aligned}
$$

Note: We always have $\pi(p-0.5) \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ for $p \in (0,1)$.

Solution ii

```
invcdf_cauchy <- function(p, theta = 0,
                          gamma = 1) {
  gamma*tan(pi*(p - 0.5)) + theta
}
unif_vars <- runif(5)
invcdf_cauchy(unif_vars)
```

```
## [1] -0.8263884 0.9488969 2.6537989 1.3050843
1.2012162
```

Inverse Transform—Discrete Edition

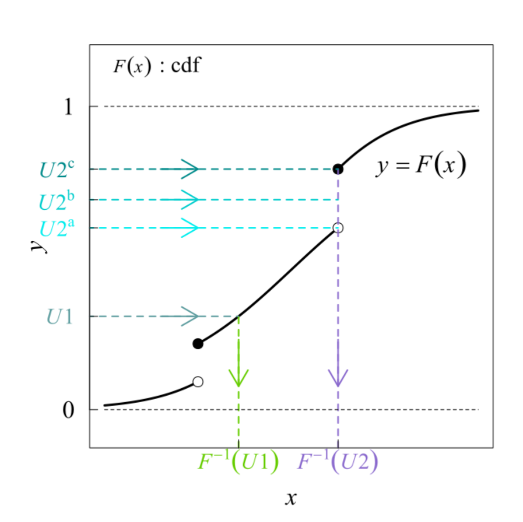


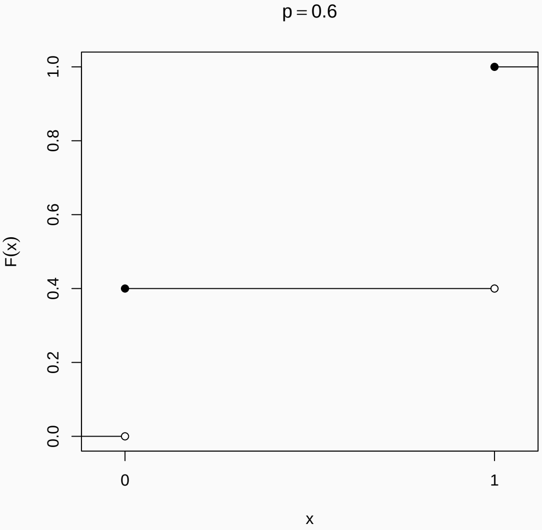
Figure 1: From Wikipedia

Example i

- Let X follow a Bernoulli distribution with parameter p:

$$F(x) = \begin{cases} 0 & x < 0, \\ 1 - p & x \in [0, 1), \\ 1 & x \geq 1. \end{cases}$$

Example ii



Example iii

- As we can see, we have

$$F^{-1}(u) = \begin{cases} 0 & u \leq 1 - p, \\ 1 & u > 1 - p. \end{cases}$$

- In other words, we sample U. If it is less than $1 - p$, we set $X = 0$; else, we set $X = 1$.

Example iv

```r
p <- 0.6
n <- 1000
unif_vars <- runif(n)
# as.numeric turns FALSE into 0
# and TRUE into 1
bern_vars <- as.numeric(unif_vars > 1 - p)
```

```r
c(mean(bern_vars), var(bern_vars))
```

```
## [1] 0.6410000 0.2303493
```

Example v

```
# Compare with theory
c(p, p*(1 - p))
```

```
## [1] 0.60 0.24
```

More General Transformations

- Inverse transform is just one type of transformation!
- We can use relationships between distributions to generate random variates. For example:
    - If $Z \sim N(0,1)$, then $Z^2 \sim \chi^2(1)$.
    - If $V_1, \ldots, V_p \sim \chi^2(1)$, then $\sum_{i=1}^{p} V_i \sim \chi^2(p)$.
    - If $U \sim \chi^2(p)$ and $V \sim \chi^2(q)$, then

$$\frac{U/p}{V/q} \sim F(p,q).$$

Example i

```
# Choose degrees of freedom
p <- 2
q <- 4

# rnorm samples from a normal distribution
U <- sum(rnorm(p)^2)
V <- sum(rnorm(q)^2)

# Take ratio
(U/p)/(V/q)
```

Example ii

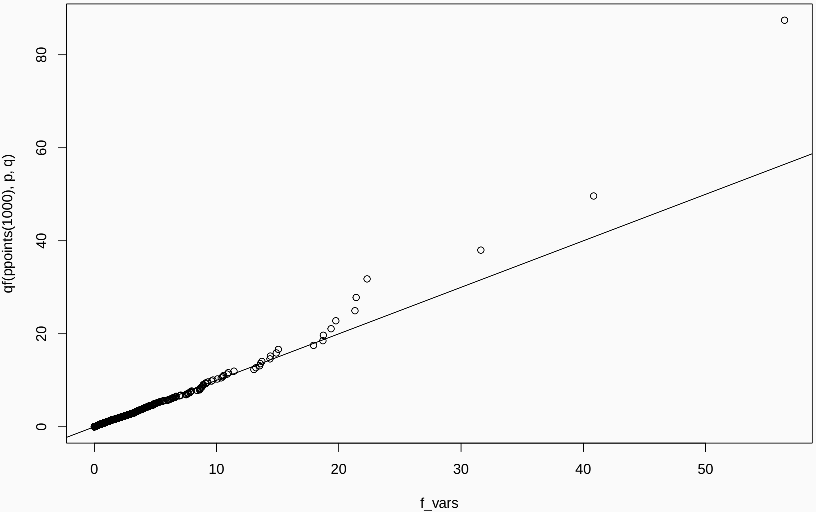
```
## [1] 1.575616
```

```r
# What if we want 1000 replicates?
# Use the function replicate!
# First argument: number of replicates
# Second argument: expression to be run multiple times
f_vars <- replicate(1000, {
  U <- sum(rnorm(p)^2)
  V <- sum(rnorm(q)^2)
  (U/p)/(V/q)
})
```

Example iii

```
qqplot(f_vars, qf(ppoints(1000), p, q))
# Add diagonal line
abline(a = 0, b = 1)
```

Example iv



Acceptance-Reject Method i

- Suppose you want to sample from a distribution X with density f, but you can only sample from a different distribution Y with density g.
- Further suppose that there exists a constant $c > 1$ such that

$$\frac{f(t)}{g(t)} \leq c$$

  for all t such that $f(t) > 0$.
- The *Acceptance-Reject method* is a way to transform random variates of Y into random variates of X.

Algorithm

1. Sample y from Y.
2. Sample a uniform variate u from $U(0, 1)$.
3. Compute the ratio $r := \frac{f(y)}{cg(y)}$. If $u < r$, set $x = y$. Otherwise, reject y and repeat from Step 1.

Note: The number of iterations before we accept a draw from Y follows a geometric distribution with mean c. So we want the constant c to be as small as possible.

(If you want a proof of why this works, see UM Learn.)

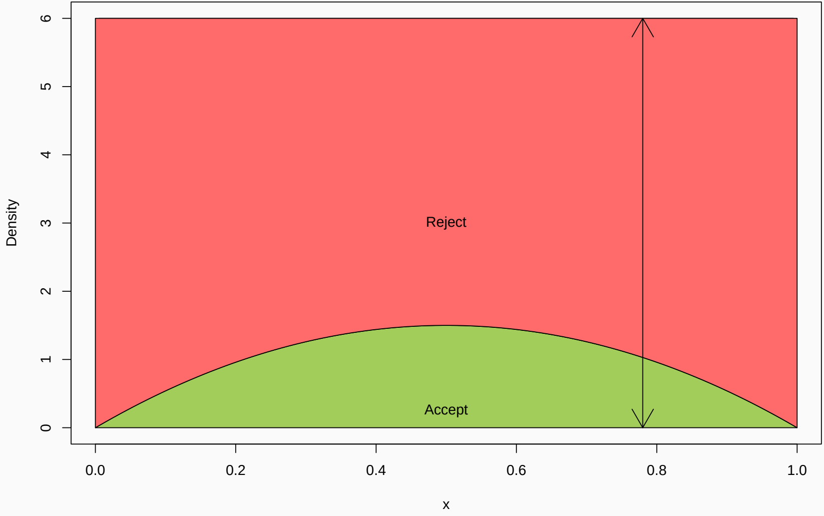
Example i

- We want to sample from $X \sim Beta(2, 2)$ whose density is $f(x) = 6x(1 - x)$.
  - The *proposal* distribution will be $Y \sim Beta(1, 1)$ (i.e. a uniform distribution).
- Let $t \in (0, 1)$. We have

$$\frac{f(t)}{g(t)} = \frac{6t(1-t)}{1} \leq 6,$$

since the maximum t and $1 - t$ can take is 1. So we can set $c = 6$.

Example ii



Example iii

```r
# Set parameters----
C <- 6 # Constant
n <- 1000 # Number of variates
k <- 0 # counter for accepted
j <- 0 # iterations
y <- numeric(n) # Allocate memory
```

Example iv

```r
# A while loop runs until condition no longer holds
while (k < n) {
  u <- runif(1)
  j <- j + 1
  x <- runif(1) # random variate from g
  if (u < 6*x*(1-x)/C) {
    k <- k + 1
    y[k] <- x
    }
}
```

Example v

```r
# How many iterations did we need?
j
```

```
## [1] 6271
```

```r
# Compare theoretical and empirical quantiles
p <- seq(0.1, 0.9, by = 0.1)
Qhat <- quantile(y, p) # empirical
Q <- qbeta(p, 2, 2) # theoretical
```

Example vi

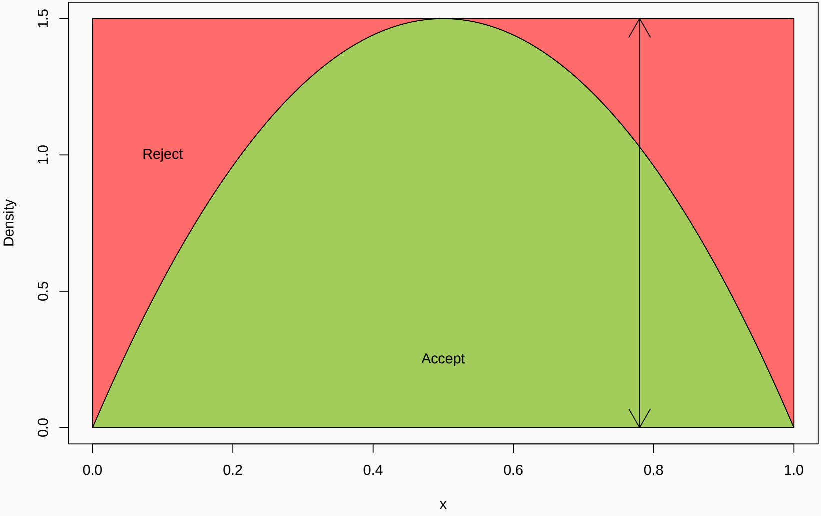
```
round(cbind(Qhat, Q, diff = abs(Qhat - Q)), 3)
```

```
##      Qhat     Q  diff
## 10% 0.201 0.196 0.005
## 20% 0.283 0.287 0.004
## 30% 0.356 0.363 0.007
## 40% 0.428 0.433 0.005
## 50% 0.491 0.500 0.009
## 60% 0.564 0.567 0.004
## 70% 0.640 0.637 0.004
## 80% 0.718 0.713 0.005
## 90% 0.805 0.804 0.001
```

Example vii

- As the graph showed, the "Rejection" region is very large.
  - In fact, it is unnecessarily large.
- With a little bit of calculus, we can show that the maximum value of $6x(1-x)$ is 1.5.
  - In other words, we can set the constant $c = 1.5$.
  - This means that we can sample from X while rejecting 4 times *less* often.

Example viii



Example ix

```
C <- 1.5; k <- j <- 0 # Reset counters
while (k < n) {
  u <- runif(1)
  j <- j + 1
  x <- runif(1)
  if (u < 6*x*(1-x)/C) {
    k <- k + 1
    y[k] <- x
    }
}
```

Example x

```r
# How many iterations did we need this time?
j
```

```
## [1] 1491
```

Summary

- When we can compute the quantile function, the inverse transform is simple to implement.
    - But it can be hard to compute!
- We can leverage relationships between distributions to transform one random variate into another.
- Accept-reject can be used when we have a bounding density.