R Programming for Demographers

Distributions and Simulation

Tim Riffe*

September 19, 2017

Contents

1 Introduction

In this session we deal with some statistical concepts such as sampling and distributions. They are often crucial tools in demography, and therefore it is important to have them in our tool-box. `R` luckily has lots of built-in goodies for this kind of thing.

2 Random Sampling

Say we want to pick one or more elements of a given data set, randomly. To do so, use the function `sample()`. In its simplest form we have to "feed" the function only with two arguments: the data `x` from which we want to sample, and how many times we want to pick components from the data (`size`).

```
sample(x = c("Prim.", "Sec.", "Tert."), size = 2)

## [1] "Sec."  "Prim."
```

Automatically, `R` assigns equal probabilities to each element of `x`, and does not replace an element which has been picked. These settings can be changed by using the additional arguments `prob` and `replace`.

*First, a big thanks to Giancarlo Camarda, who has let me recycle his impeccable material for teaching this course. Higher order thanks are also due to the very same people that Giancarlo himself thanked, especially Sabine Zinn and Roland Rau. Sabine was my teacher in this very course in 2009 (cohort 5)! Part of these lectures has been freely inspired by courses that both Roland, Sabine, and Giancarlo taught in the past. Furthermore, in recent years, Adam Lenart, Fernando Colchero, and Silvia Rizzi have aided in teaching this course and improving handouts. On the other hand, it goes without saying that I am uniquely responsible for any deficiencies and mistakes you may find in handouts.

```
sample(x = c("Prim.", "Sec.", "Tert."), size = 10, replace = TRUE,
       prob = c(0.5, 0.3, 0.2))

##  [1] "Sec."  "Prim." "Prim." "Tert." "Prim." "Sec."  "Prim." "Sec."
##  [9] "Prim." "Sec."
```

As you have seen in a previous lecture, the function `table()` produces simple frequency tables. Let us check whether we approach the given probabilities when we increase the sample size, i.e. let us check the *law of large numbers*:

```
t <- table(sample(x = c("Prim.", "Sec.", "Tert."), size = 10,
                  replace = TRUE,
                  prob = c(0.5, 0.3, 0.2)))
t/10

##
## Prim.  Sec. Tert.
##   0.6   0.2   0.2

t <- table(sample(x = c("Prim.", "Sec.", "Tert."), size = 100,
                  replace = TRUE,
                  prob = c(0.5, 0.3, 0.2)))
t/100

##
## Prim.  Sec. Tert.
##  0.43  0.39  0.18

t <- table(sample(x = c("Prim.", "Sec.", "Tert."), size = 10000,
                  replace = TRUE,
                  prob = c(0.5, 0.3, 0.2)))
t/10000

##
##  Prim.   Sec.  Tert.
## 0.4939 0.3010 0.2051
```

3 Distributions and Random Numbers

Distributions are one of the basic concepts in statistics. As we have learned, `R` has many built-in functions which are useful for many purposes. Specifically it offers four basic functions for each of the major families of probability distributions: the density function, the cumulative distribution function, the quantile function, and a function which generates random numbers from a given distribution.

A function for a certain distribution is called by giving the name of this distribution in `R` preceded by either one of the following letter:

`d` for the probability function (discrete case)/ density function (continuous case)

`p` for the cumulative distribution function

`q` for the quantile function

`r` for generating random numbers

In the following sections we consider different examples in detail. Table 1 gives an overview over some built-in distributions in the `R` package `stats`.

The package `MASS` provides additional distribution functions; e.g. for the negative binomial distribution (`dnegbin()`, `pnegbin()`, `qnegbin()`, `rnegbin()`). Don't see your favorite distribution? There's probably a package for it... in the EDSD we hand-code our favorite distributions for demography.

name in R	distribution	arguments
beta	beta	shape1, shape2, non-centrality parameter
binom	binomial	size, probability
chisq	χ^2 (Chi-Square)	degrees of freedom, non-centrality parameter
exp	exponential	rate
f	F	degrees of freedom, non-centrality parameter
lnorm	log-normal	mean, standard deviation on the log scale
logis	logistic	location, scale
norm	normal	mean, standard deviation
pois	poisson	rate
t	student's t	degrees of freedom, non-centrality parameter
unif	uniform	min, max
weibull	weibull	shape, scale

Table 1: Functions for the Generation of Random Numbers

3.1 Binomial Distribution

The binomial distribution is a discrete distribution, i.e. binomially distributed random numbers are always whole-numbers. This distribution represents an important case of the multinomial distribution occurring when the variable has only two distinct values.

For example, suppose that a certain proportions p of individuals in a given population are smokers. The binomial distribution informs us about the probability to observe x smokers if we draw n individuals from this population. For another example, assume that the probability of conceiving after a certain treatment is equal to p, and n individuals undergo this treatment. The binomial distribution tells us the probability that $x = [0, 1, 2, \ldots, n]$ individuals will conceive. , In general the probability function of a binomial distribution can be written as follows:

$$B(n, p) = B_{n,p}\,(X = x) = \binom{n}{x} p^x\,(1-p)^{n-x}\,,$$

where n is the number of trials, p is the success probability in each trial, and x is the number of successes. The expected values and the variance of this distribution are given by

$$\mathrm{E}[X] = n\,p \qquad \text{and} \qquad \mathrm{Var}[X] = n\,p\,(1-p)$$

Taking the example above and assuming that 25% of the population smokes ($p = 0.25$) and we draw $n = 15$ individuals from such population, in R we can compute the probability of obtaining $x = 6$ smokers as follows: `dbinom()`

```
p <- dbinom(x = 6, size = 15, prob = 0.25)
p

## [1] 0.09174777
```

i.e., we have less than 10% chances of getting 6 smokers out of 15 if the probability of finding a smoker in the population is 25%.

Using the other example, we compute the probability to get $x = 7$ conceptions out of $n = 12$ trials, given a probability of success equal to $p = 0.75$:

```
p <- dbinom(x = 7, size = 12, prob = 0.75)
p

## [1] 0.1032414
```

i.e., about 10%. But, given the same probability and number of trials, what about finding either no conceptions or only one or two, etc.? We can supply a sequence of numbers to the argument `x` obtaining the associated probabilities. An additional plot (Fig. 1) can help to understand the distribution given our parameters:

```
x  <- 0:12
pp <- dbinom(x = x, size = 12, prob = 0.75)
```

The probability of obtaining at least 8 conceptions can be computed by either summing up the single elements of pp up to the 8th element, or using the cumulative distribution function: `pbinom()`

```
cp <- pbinom(q = 8, size = 12, prob = 0.75)
cp

## [1] 0.3512214
```

i.e., we have a 35% chance of getting at least 8 conceptions out of 12 women undergoing the treatment. By continuing with the same example, if we aim to simulate 10 outcomes from the same experiment, we should proceed as follows: `rbinom()`

```
rn <- rbinom(n = 10, size = 12, prob = 0.75)
rn

##  [1]  6  7  9 10  7 10  9  7 10  8
```

obtaining the number of conceptions that, for instance, 10 independent clinics would achieve with the same treatment (always $p = 0.75$) over the same number of patients, $n = 12$.

3.2 Central Limit Theorem

In this section we use the distribution functions of `R` in a slightly more involved way: *the central limit theorem*. The central limit theorem (CLT) is of huge interest in statistics (especially in estimation and test theory). Here we use it just as a game for learning `R` .

In few words, let X_i, $i = 1, \cdots, n$ be a sequence of n identically and independent distributed (i.i.d.) random numbers each having finite values of expectation μ and variance $\sigma^2 > 0$. The CLT states that as the sample size n increases, the distribution of the sample *average* of these random number variables approaches the normal distribution with mean μ and variance σ^2/n, irrespective of the shape of the common distribution of the individual terms X_i. This convergence works when the second moments of the X_i are finite[1].

Using the CLT ,it is often possible to approximate distributions of sums of a large number of random variables by a normal distribution. Let's clarify all this with an example.

First we generate 30 random numbers 10000 times. These numbers are continuously uniformly distributed in $[0, 1]$, and also i.i.d. So we meet the first requirement of CLT. The other one is that the second moments of the random variables are finite. The second moment of a random variable, which is continuously uniformly distributed in $[0, 1]$, is $1/3$. The assumptions of the central limit theorem are thus met. We can apply the theorem to our example.

```
# number of random numbers for each sample
n <- 30
# number of samples
s <- 10000
# all random values in a matrix
r <- matrix(runif(n * s), nrow = s, ncol = n)
```

Then we calculate 10000 times the arithmetic mean of the 30 uniformly distributed random numbers in each row of `r`.

```
xquer <- apply(r, 1, mean)
# or just:
# rowMeans(r)
```

Now we plot the frequency distribution of the 10000 arithmetic means using a histogram:

[1] For an analytical formulation of the central limit theorem see e.g. Chapter 5 in Casella and Berger (1990). The last requirement means that the expected values of the X_i^2 are finite: $E\left(X_i^2\right) < \infty, i = 1, \cdots, n$.

```
plot(x, pp, t="h", lwd=2)
```

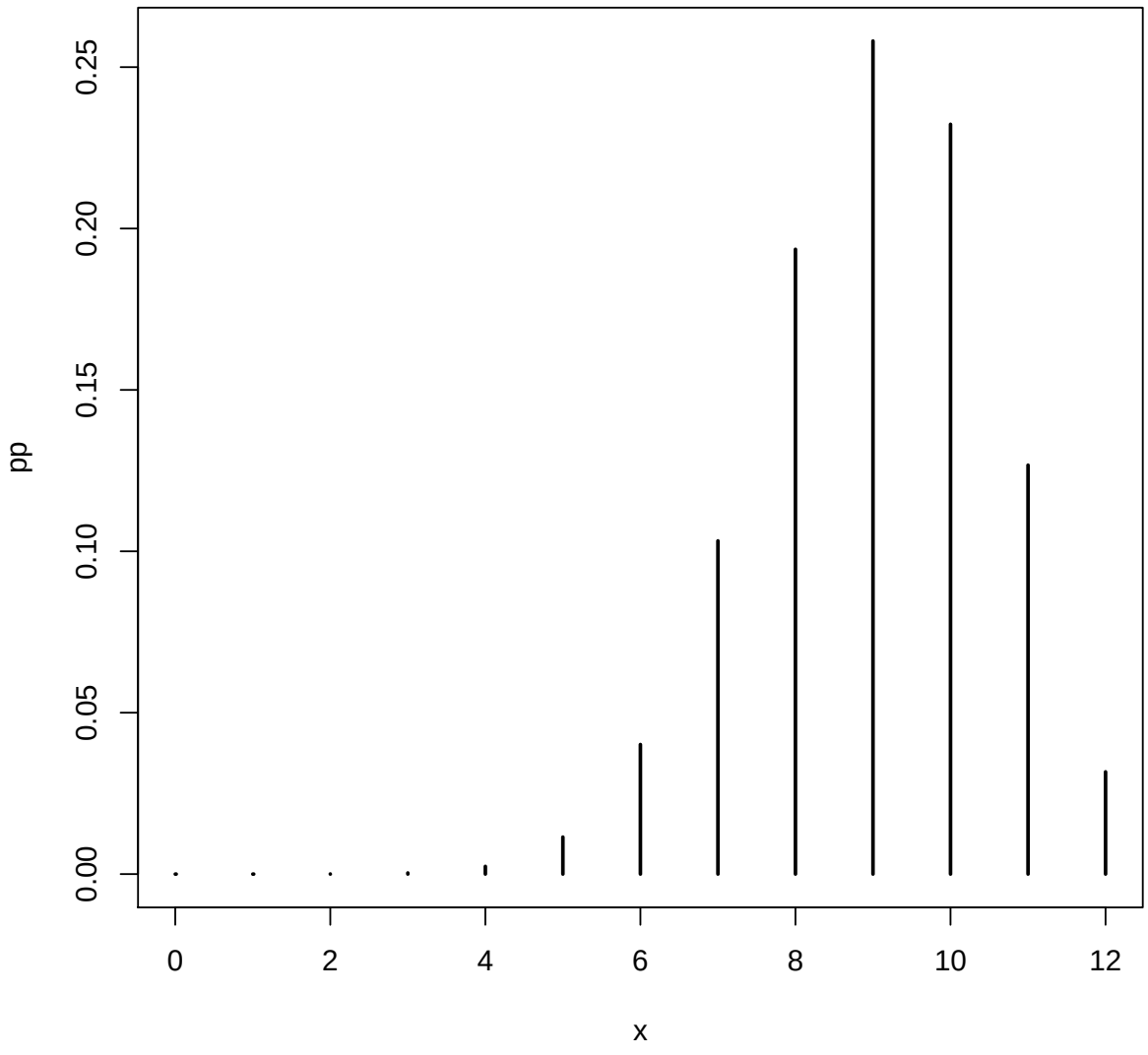
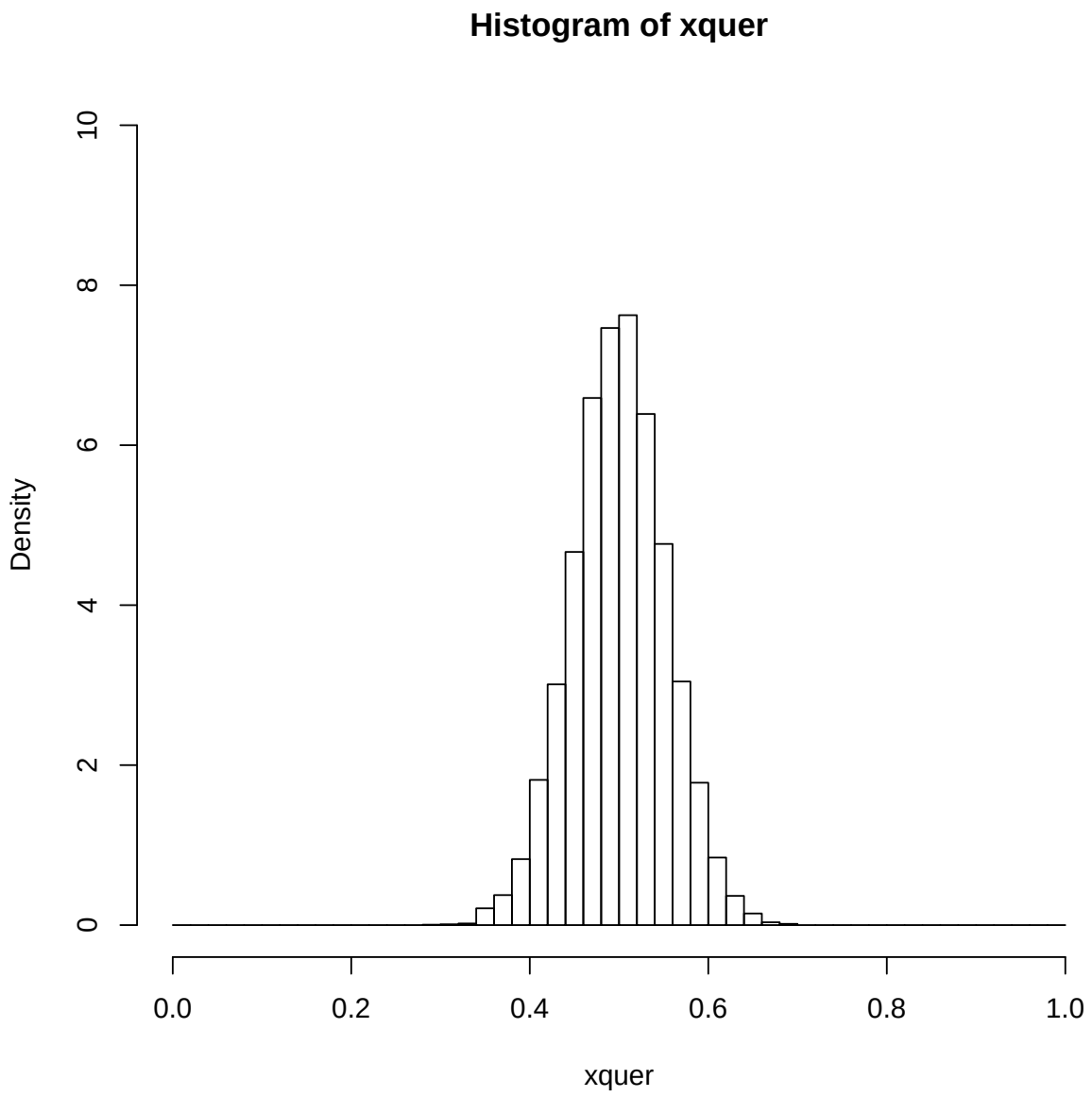


Figure 1: Example of binomial distribution with $p = 0.75$, $n = 12$ and $x = 0, 1, 2, \ldots, 12$

```
hist(xquer, breaks = seq(0, 1, by = 0.02),
     ylim = c(0, 10), freq = FALSE)
```



Look at the next step for understanding the choice for the arguments `breaks`, `ylim` and `freq`.

We know that the expected value and the variance for a uniform distribution between 0 and 1 are equal to $\mu = 0.5$ and $\sigma^2 = 1/12$, respectively. Therefore, the associated density of the limit normal distribution has mean $\mu = 0.5$ and standard deviation equal to $\sigma = \sqrt{\frac{1}{12 \cdot 30}} = \sqrt{1/360}$. For checking the convergence we can add such normal distribution to the previous histogram:

```
x        <- seq(-4, 4, by = 0.02)
normMean <- 0.5
normSD   <- sqrt(1/ 360)
normDens <- dnorm(x, normMean, normSD)
lines(x, normDens, lwd = 3, col = 2)
```

Figure 2 presents the final outcome.

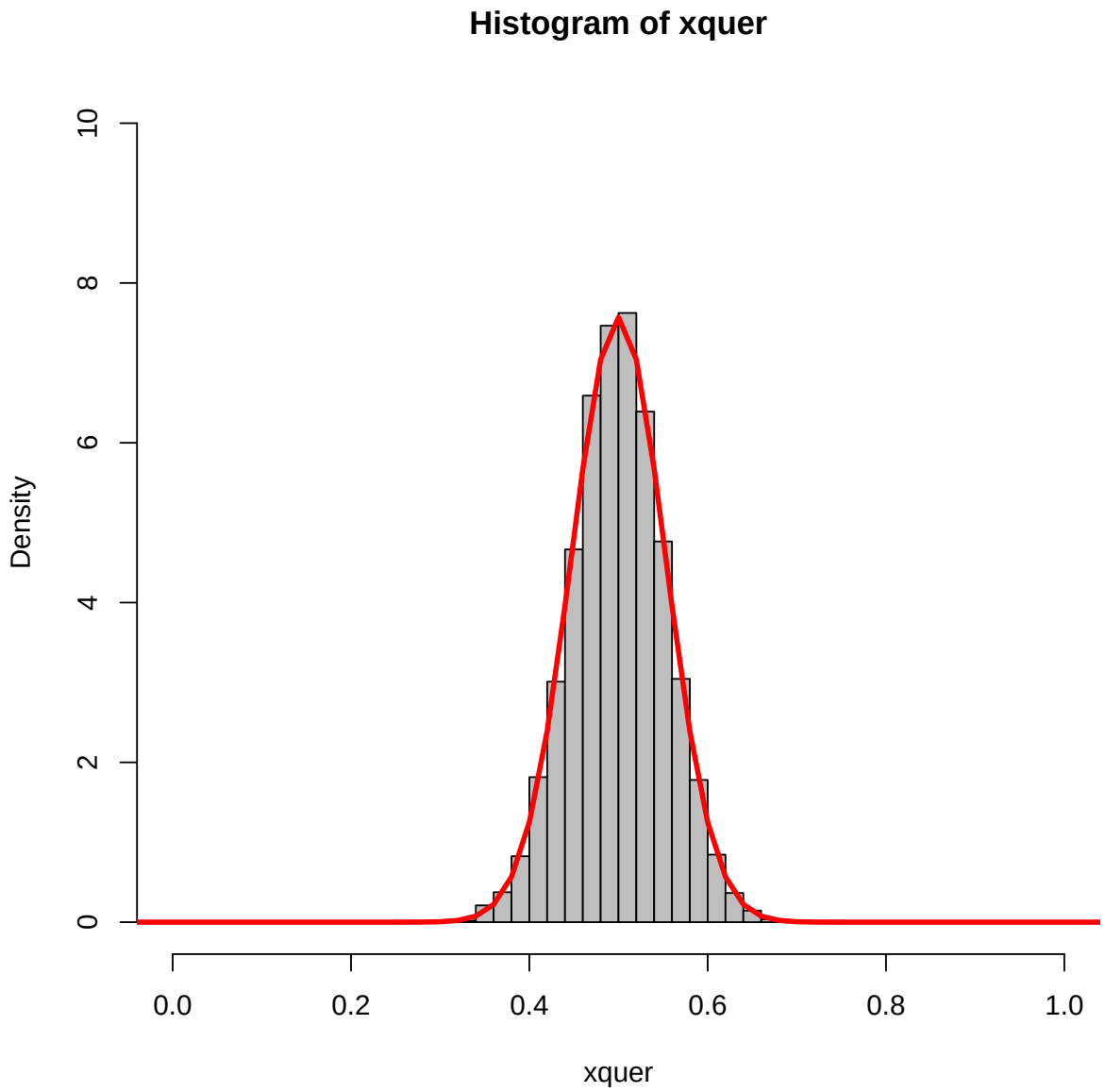


Figure 2: Example of Central Limit Theorem

3.3 Poisson Distribution

Another important distribution in demography is the Poisson, which can be conceived as the limit of a sequence of binomial distributions, where the expected value converges to a constant λ. Specifically both mean and variance have the same value in a Poisson setting, and often λ is called the rate.

We encounter the Poisson when working with count data. In these instances, linear models are not a proper choice, because counts can only take positive values, and it is unlikely that a variable that only takes counts follows a Normal distribution.

Examples of such data are the number of persons killed by horse kicks in the Prussian army per year[2]; the number of people in line in front of you at the grocery store, and the number of awards earned by students in a high school. In demography it is easy to see that we commonly count events (e.g. deaths, births, marriages), therefore this distribution plays an important role for us.

The probability of a Poisson random variable X could take nonnegative integers $(0, 1, 2, \ldots)$ is given by:

$$P(X = x) = \frac{e^{-x}\lambda^x}{x!} \quad x = 0, 1, 2, \ldots$$

In `R` we can directly use the function `dpois()` to evaluate these probabilities. For instance, the probability of having 10 deaths in the next year, given that the average deaths per year in this population is (and ceteris paribus) 15 is equal to:

```
dpois(x = 10, lambda = 15)

## [1] 0.04861075
```

If we aim to see the probability of getting different numbesr of deaths given the same λ, we could type the following lines:

```
nd  <- 0:50
pro <- dpois(x = nd, lambda = 15)
plot(nd, pro)
```

Simulating fertility age-patterns

Let's take the data for United States in 2000 from the file `USAfert2000.txt` and extract the age age column and the associated female population:

```
dati <- read.table("USAfert2000.txt")
x    <- dati$a
W    <- dati$W
m    <- length(x)
```

Assume that age specific fertility rates over ages x in USA in 2000 follow a Hadwiger (1940) function given by:

$$f(a, b, c; x) = \frac{ab}{c}\left(\frac{c}{x}\right)^{\frac{3}{2}} \exp\left[-b^2\left(\frac{c}{x} + \frac{x}{c} - 2\right)\right]$$

where the parameters a, b and c are associated with total fertility, height of the curve, and the (mortality-free) mean age of motherhood.

In this section we use a Poisson distribution to simulate birth counts from Hadwiger function with $a = 1.5$, $b = 3.8$ and $c = 28$. Rather than computing the function only for this specific instance, we write an `R` -function for computing age-specific fertility rates that follows the hypothesized model:

```
Hadwiger <- function(a, b, c, x){
  ## function for computing asfr from
  ## Hadwiger (1940) model
  p1   <- a * b / c * (c / x)^(3 / 2)
```

[2] This is the first thing that came to your mind, right?

```
  p2   <- -b^2 * (c / x + x / c -2)
  asfr <- p1 * exp(p2)
  return(asfr)
}
```

Then we set up the true underlying fertility age-pattern over the object `x`:

```
true.asfr <- Hadwiger(1.5, 3.8, 28, x)
```

To simulate Poisson counts, we need to compute the mean number of births given the true pattern for the USA female population. This is computed by multiplying the total population by the true age-specific fertility rates:

```
true.B  <- true.asfr * W
```

Now we can simulate the number of births for each age from a Poisson distribution and then compute the simulated age-specific fertility rates

```
sim.B    <- rpois(m, true.B)
sim.asfr <- sim.B / W
```

To assess the goodness of our simulation, we plot the simulated rates alongside the true ones. In this case the correspondence is nearly perfect because we dealt with a huge population. But what if we were to divide the exposure population by 1000? A simulated outcome is presented in Figure 3

3.4 Gompertz Distribution

We turn now to another demographic example: We simulate random numbers from a Gompertz distribution. Unfortunately this distribution is not included in `R` among the basic packages[3] so we have to do it ourselves via the *inverse transform sampling method*.

The inverse transform sampling method is a method of sampling a number at random from a probability distribution, given its cumulative distribution function (cdf). Let X be a random number described by cdf, $F(X)$. The method works as follows:

1. generate a random number from a standard uniform distribution; call this u
2. compute the value for x which has the associated cdf value u; call this x_c
3. take x_c to be the random number drawn from the distribution described by $F(X)$.

The diagram in Figure 4 may help you to visualize how the method works for normal and exponential distributions as well as for a generic one.

The cumulative distribution function (cdf) in the Gompertz case is defined as:

$$F(x) = 1 - e^{\left(\frac{\alpha}{\beta}\left(1-e^{\beta x}\right)\right)}.$$

with α and β are the intercept and the slope parameters of the Gompertz distribution.

We know that $F(x) \in [0,1]$. If X is a random number, $F(X)$ is a random number which is continuously and uniformly distributed in $[0,1]$. We obtain a Gompertz distributed random number by rearranging

[3]The Gompertz distribution is included in the library `eha`, but its parametrization differs from the common demographic representation, which will be used in the following.

```
plot(x, sim.asfr)
lines(x, true.asfr, col = 2, lwd = 2)
```

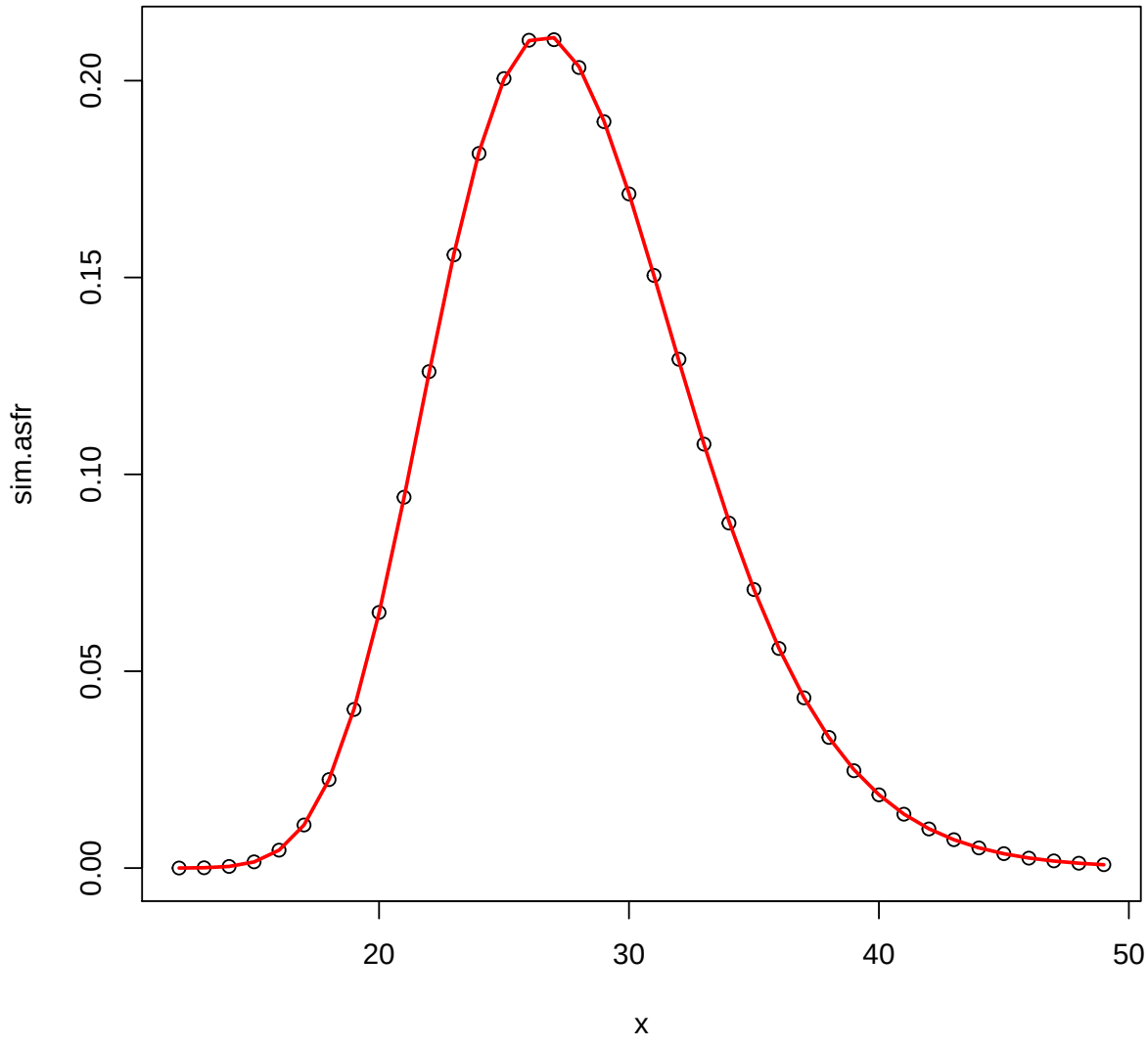


Figure 3: Simulated and true age specific fertility rates from the Hadwiger (1940) model with 1/1000 of the US female population in 2000.

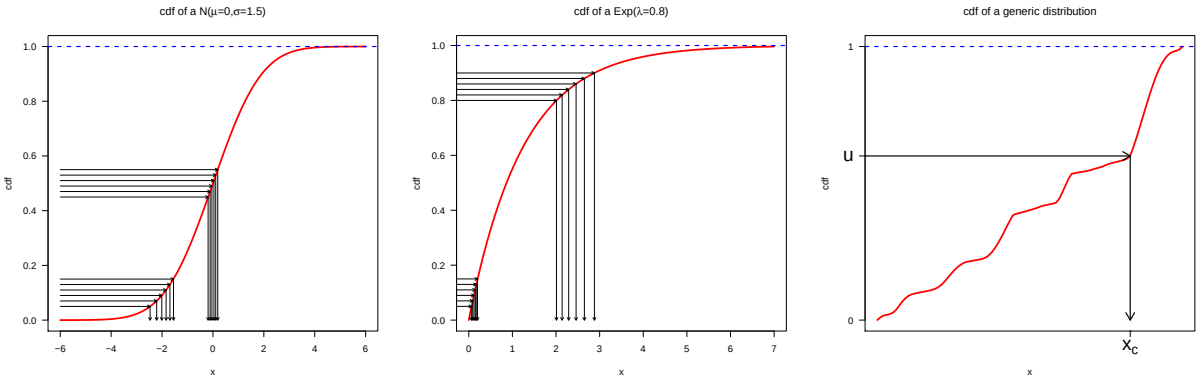


Figure 4: Examples of inverse transform sampling method. Normal distribution (left), exponential distribution (center) and a generic distribution (right).

the cumulative distribution function:

$$\begin{aligned} u &= 1 - e^{\left(\frac{\alpha}{\beta}\left(1-e^{\beta x}\right)\right)} \\ &\Downarrow \\ \ln(1-u) &= \frac{\alpha}{\beta}\left(1-e^{\beta x}\right) \\ &\Downarrow \\ \frac{\beta}{\alpha}\ln(1-u) &= \left(1-e^{\beta x}\right) \\ &\Downarrow \\ \ln\left(1-\frac{\beta}{\alpha}\ln(1-u)\right) &= \beta x \\ &\Downarrow \\ x &= \frac{1}{\beta}\ln\left(1-\frac{\beta}{\alpha}\ln\left(1-u\right)\right) \end{aligned}$$

where $u \sim U(0,1)$.

Now we can write a function in R that generates n random numbers that are Gompertz distributed using this last equation:

```
rgomp <- function(n, alpha, beta) {
  r    <- 1 - runif(n) # <- complement actually not necessary here
  help <- 1 - (beta / alpha) * log(r)
  res  <- log(help) / beta
  return(res)
}
```

We check our code by assuming $\alpha = 0.00003$ and $\beta = 0.1$.[4]

```
a <- 0.00003
b <- 0.1
rgomp(1, a, b)

## [1] 41.36715
```

How does a distribution of 2000 "lifetimes" generated with these parameters α and β look? We plot a histogram of these data with 100 breaks to visualize our results: What is the mean of these Gompertzian life-times? Just type:

[4]The slope parameter β of 0.1 is typical in human populations for adult ages from 30-85. An appropriate value of α has been chosen deliberately to obtain values that sort of resemble human lifetimes.

```
n  <- 2000
lt <- rgomp(n, a, b)
hist(lt, breaks = 100)
```

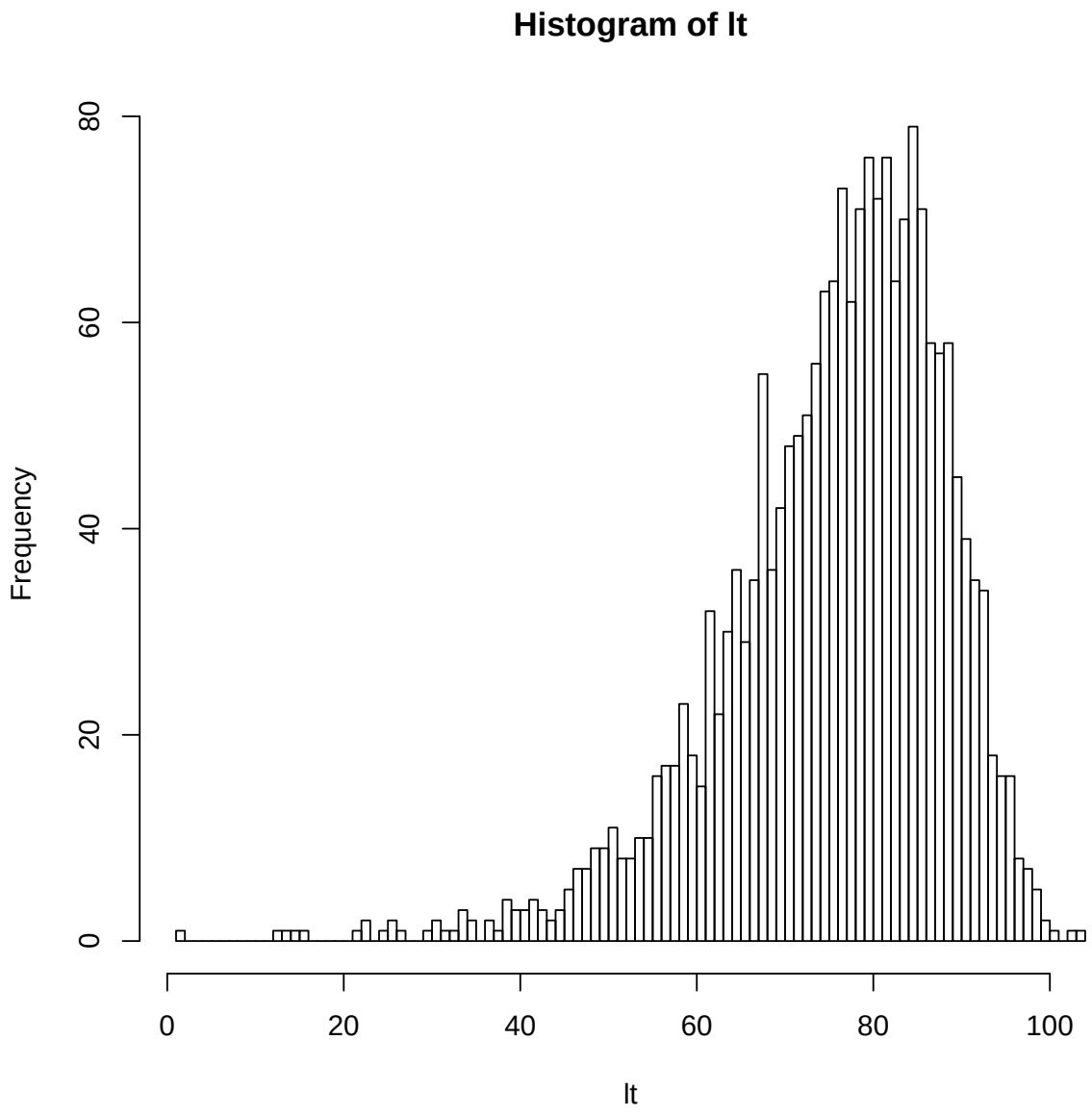


Figure 5: Distribution of 2000 random Gompertz lifetimes ($\alpha = 0.00003, \beta = 0.1$)

```
mean(lt)

## [1] 75.59632
```

To check whether the simulated life-times follow a Gompertz distribution (or whether we have some bug!), overlap the histogram of the simulated data with the true density. Specifically, the density of a Gompertz distribution is given by:

$$f(X = x) = \alpha\, e^{\beta x} \exp\left[\frac{\alpha}{\beta}\left(1 - e^{\beta x}\right)\right]$$

and we translate this in R as follows:

```
dgomp <- function(x, alpha, beta){
  p1 <- alpha * exp(beta * x)
  p2 <- exp(alpha/beta * (1 - exp(beta * x)))
  d  <- p1 * p2
  return(d)
}
```

Compute the true density from age 0 to 120:

```
ages      <- 0:120
true.Gomp <- dgomp(ages, a, b)
```

Finally we can plot the two densities. Remember to set `freq` equal to `FALSE` to plot the density instead of the actual frequency distribution. Outcomes in Figure 6.

```
hist(lt, breaks = 100, freq = FALSE)
lines(ages, true.Gomp, col = 2, lwd = 2)
```

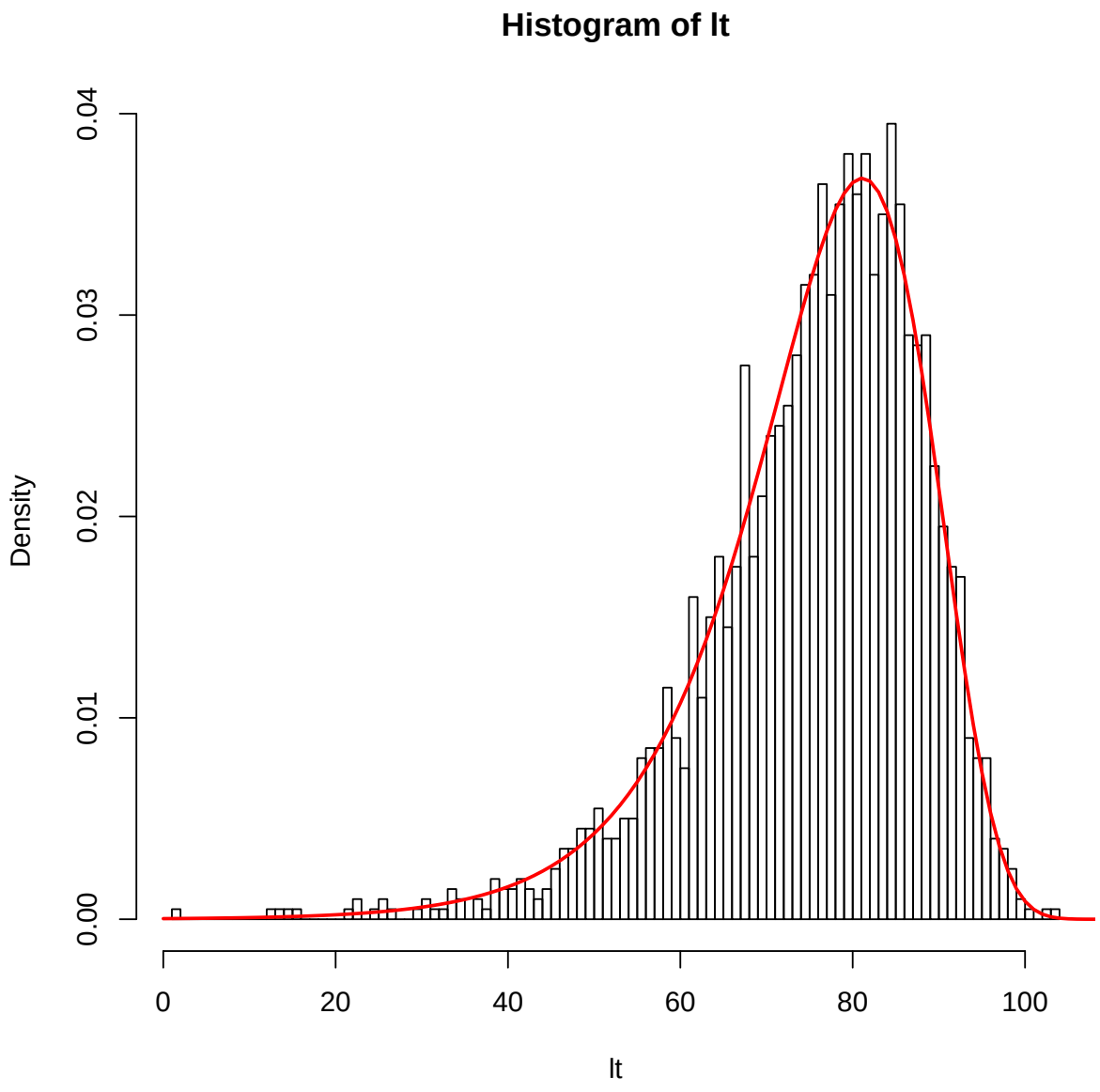


Figure 6: Density of 2000 simulated life-times from a Gompertz distribution along with the true generated density.

4 Exercises

4.1 Exercise 1

Simulate $n = 10000$ random number from a binomial distribution with parameter 20 and 0.3. Then estimate the following values:

- $E[X]$
- $Var(X)$
- the 90th percentile of X
- $P(X \leq 7)$
- $P(X = 7)$

Finally, compare the estimated values with the true ones.

4.2 Exercise 2

We have seen how one may describe mortality over adult ages using a Gompertz model. For this exercise we will use the same model, but the task is to simulate data at the aggregate level, i.e. we will simulate death counts for each age assuming that mortality follow a Gompertz model given by:

$$\mu(a, b; x) = a\, e^{b\, x}$$

where a and b are parameters and x is the set of ages over we aim to study/simulate mortality.

For this exercise we will use the same true parameters that in Section 3.4 and we additionally provide a set of exposures in the file `expdata.txt`. In this file you can find both the age-range over which you will simulate death counts and the associated exposures.

After simulating the data, your task is to plot (in log-scale) both true μ and simulated death rate, i.e. the ratio between deaths and exposures for each age.

References

Casella, G. and R. L. Berger (1990). *Statistical inference. The Wadsworth & Brooks/Cole Statistics/Probability Series. Wadsworth & Brooks*. Cole Advanced Books & Software, Pacific Grove, CA.

Hadwiger, H. (1940). Eine analytische reproduktionsfunktion für biologische gesamtheiten. *Scandinavian Actuarial Journal 1940*(3-4), 101–113.