Advanced Scientific Computing with R

5. Simulating Data

Michael Hahsler

Southern Methodist University

September 29, 2011

SMU | Bobby B. Lyle School of Engineering

Introduction

Simulated ("random") data is used in many areas:

- gambling
- statistical sampling
- computer simulation
- cryptography
- simulations (Monte Carlo experiments)

Table of Contents

Sampling

'sample' takes a sample of the specified size from the elements of 'x' using either with or without replacement.

```
R> sample(1:100, size=10)
 [1] 12 62 60 61 83 97  1 22 99 47
R> sample(1:10, size=100, replace=TRUE)
  [1]  7  6  3 10  3  9  3  3  2  3  4  4  2  1  3  9  6 10
 [19]  9  1  5  3  4  6  2  8  3  3 10  9  6  7  4  7  4  6
 [37]  7  5  3  8  1  4  8  6  2  6  5  8  2  9  9  1  4  1
 [55]  3  8  4  6  1  6  2  9  1  8  1  6  4  1  4  7 10  5
 [73]  2  6  2  9  4  4  2  9  2 10  2  2  2  6  4  1  4  8
 [91]  1  6  3  3  2  4  2  2  5  1
```

sample can be used to sample from data.frames and matrices.

```
R> data(iris)
R> dim(iris)
[1] 150   5
R> s <- iris[sample(1:nrow(iris), size=50), ]
R> dim(s)
[1] 50  5
```

Simple Coin Tossing

We can specify the probability for each outcome.

```
R> x <- sample(c(TRUE, FALSE), 100, replace=TRUE,
prob=c(0.2,0.8))
R> x
  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
 [10] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [19] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE
 [28] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE
 [37]  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
 [46] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
 [55] FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
 [64] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
 [73] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
 [82] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
 [91] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[100] FALSE
R> table(x)
x
FALSE  TRUE
   83    17
```

Simple Coin Tossing II

```
R> barplot(table(x))
```

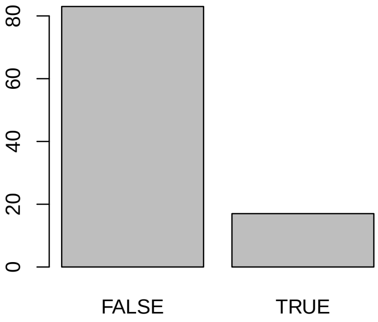


Table of Contents

Distributions

Functions for all distributions in R come in 4 variants. For example for the normal distribution we have:

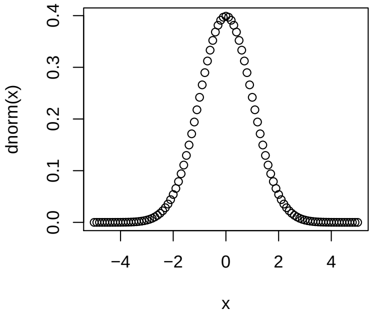
```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

Probability density function (d), distribution function (p), quantile function (q) and random deviates (r).

Probability density function (pdf)

Probability of a random variable taking certain values: $f(x)$

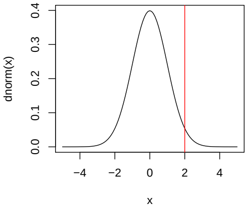
```
R> x <- seq(-5,5, by=.1)
R> plot(x, dnorm(x))
```



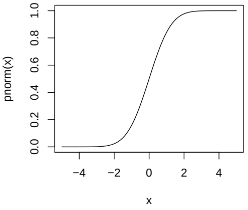
(Cumulative) distribution function (cdf)

Probability that a real-valued random variable X with a given probability distribution will be found at a value less than or equal to x: $F_X(x) = P(X \leq x)$

```
R> plot(x, dnorm(x), "l")
R> abline(v=2, col="red")
```



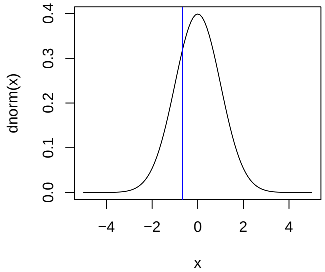

```
R> plot(x, pnorm(x), "l")
```



Quantile function

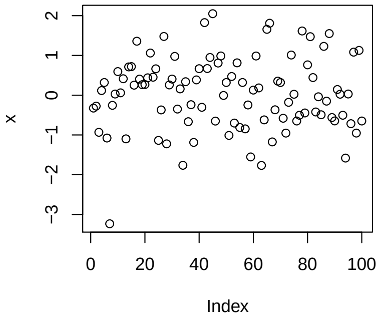
$$Q(p) = \inf\{x \in R : p \leq F(x)\}$$

```
R> qnorm(.25)
[1] -0.674
R> ## 25% quantile
R> plot(x, dnorm(x), type="l")
R> abline(v=qnorm(.25), col="blue")
```



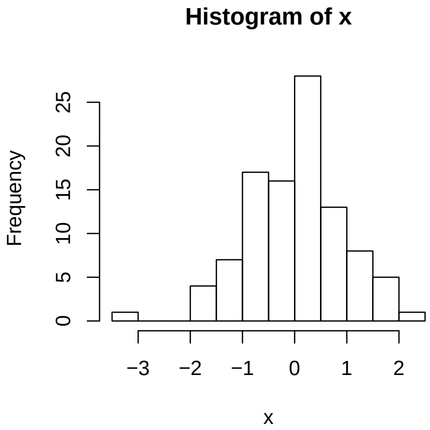
Random deviates

```
R> x <- rnorm(100)
R> head(x)
[1]  1.014  0.253 -1.172  0.669 -1.650 -0.366
R> plot(x)
```



Random deviates II

```
R> hist(x)
```



Some useful distributions

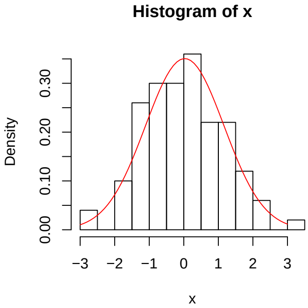
- rnorm
- rlnorm
- runif
- rpois
- rexp
- rbinom
- rnbinom
- rmultinom
- rchisq
- rt
- rbeta
- rweibull

Table of Contents

Histogram

Compare empirical distribution with a fitted theoretical distribution.

```
R> x <- rnorm(100)
R> hist(x, breaks=20, probability=TRUE)
R> mu <- mean(x)
R> sd <- sd(x)
R> r <- seq(-3,3, by =.1)
R> lines(r, dnorm(r, mean=mu, sd=sd), col="red")
```



Quantile-Quantile plot

```
R> qqplot(x, rnorm(100, mean=mu, sd=sd))
R> # use qqnorm for normal distribution
R> abline(0,1, col="red")
```

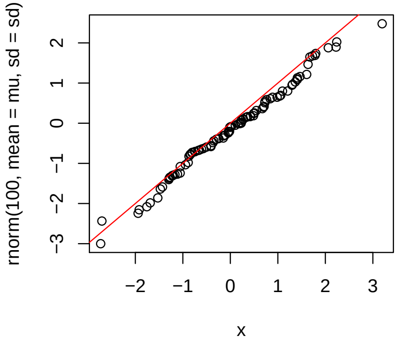
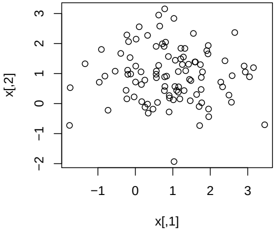


Table of Contents

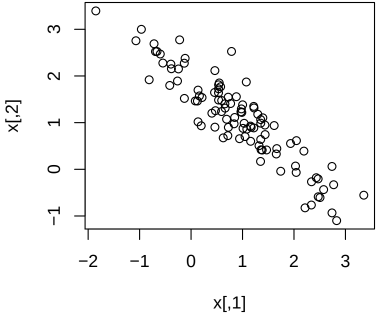
Multivariate Distributions

```
R> library(MASS)
R> Sigma <- rbind(c(1,0), c(0,1)) ## covariance matrix
R> x <- mvrnorm(100, c(1,1), Sigma=Sigma)
R> head(x)
      [,1]  [,2]
[1,] 2.189 0.767
[2,] 2.060 1.156
[3,] 2.337 0.396
[4,] 0.633 1.629
[5,] 0.696 1.714
[6,] 0.650 2.076
R> plot(x)
```



Multivariate Distributions

```
R> Sigma <- rbind(c(1,.9), c(-.9,1)) ## strong correlation
R> x <- mvrnorm(100, c(1,1), Sigma=Sigma)
R> plot(x)
```



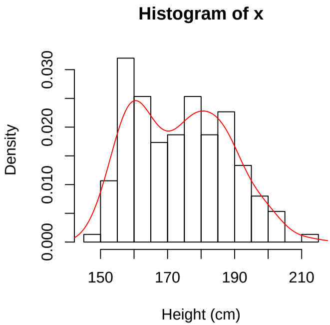
More about multivariate data can be found in the Task View "Multivariate"

Table of Contents

Mixture of two univariate Gaussian

Measurement of height (in centimeters) for subjects from two groups (female/male).

```
R> female <- rnorm(50, 160, 5)
R> male <- rnorm(100, 180, 10)
R> x<-c(female,male)
R> hist(x, prob=TRUE, breaks=20, xlab="Height (cm)")
R> lines(density(x), col="red")
```



Multivariate data

Create a dataset for clustering with two clusters and uniform noise.

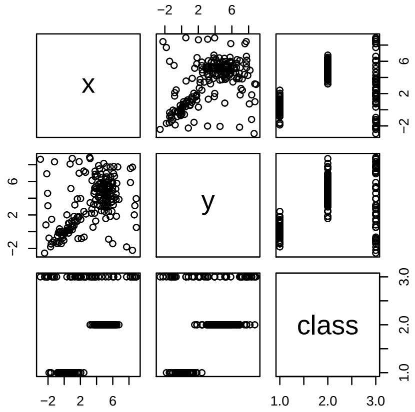
```
R> c1 <- mvrnorm(50, c(0,0), Sigma=rbind(c(1,.9), c(.9,1)))

R> c2 <- mvrnorm(100, c(5,5), Sigma=rbind(c(.5,0),
c(-.3,2)))
R> noise <- cbind(runif(50, -3,9), runif(50, -3,9))
R> x <- rbind(c1,c2,noise)
R> class <- c(rep("c1", nrow(c1)), rep("c2",nrow(c2)),
rep("noise", nrow(noise)))
R> data <- cbind(as.data.frame(x), class)
R> colnames(data) <- c("x", "y", "class")
R> data <- data[sample(1:nrow(data)), ] ## shuffle the data

R> head(data)
       x      y class
51  4.68  4.625    c2
164 5.49 -0.898 noise
86  4.97  4.373    c2
79  5.68  7.728    c2
167 3.91  5.375 noise
71  5.68  1.813    c2
```

Multivariate data II

```
R> plot(data)
```



Multivariate data III

```
R> cl <- kmeans(data[-3],2)
R> plot(data, col= cl$cluster)
```

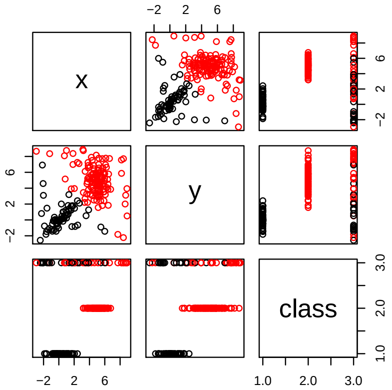


Table of Contents

Exercises

1. You use two dice for a party. The first die is fair while the second one has a 10% higher chance of rolling a 6 and a 5% each lower chance to role a 1 or a 4. Each time a player chooses randomly one die and rolls it. Display the distribution of the numbers rolled after 100 times. Hint: use sample for the dice.
2. Create a variable with 100 random values following a Poisson distribution with parameters of your choice. Use a histogram and a Q-Q plot to compare the distribution to a normal distribution and to a Poisson distribution.