Computational Statistics

An Introduction to 

Günther Sawitzki

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Computational Statistics

An Introduction to 

Exercises

Günther Sawitzki
StatLab Heidelberg
April 28, 2014

in preparation

http://sintro.r-forge.r-project.org/

Contents

CHAPTER 1

Basic Data Analysis

1.1 R Programming Conventions

1.2 Generation of Random Numbers and Patterns

1.2.1 Random Numbers

Exercise 1.1	
	Try experimenting with these plots and `runif()`. Do the plots show images of random numbers? To be more precise: do you accept these plots as images of 100 independent realisations of random numbers, distributed uniformly on $(0, 1)$? Repeat your experiments and try to note as precisely as possible the arguments you have for or against (uniform) randomness. What is your conclusion? Walk through your arguments and try to draft a test strategy to analyse a sequence of numbers for (uniform) randomness. Try to formulate your strategy as clearly as possible. *Hint:* For comparison, you can keep several plots in a window. The code `par(mfrow = c(2, 3))` parametrises the graphics system to show six plots simultaneously, arranged row wise as a 2×3 matrix (2 rows, 3 columns). The function `par` is the central function to control graphics parameters. For more information, see `help(par)`.

1.2.2 Patterns

Exercise 1.2	
	Use `plot(sin(1:100))` to generate a plot of a discretised sine function. Use your strategy from Exercise 1.1. Does your strategy detect that the sine function is not a random sequence? *Hint:* If you do not recognise the sine function at first sight, use `plot(sin (1:100), type = "l")` to connect the points.

1.3 Case Study: Distribution Diagnostics

1.3.1 First Pass for Example ??*: Distribution Functions*

1.3.2 First Pass for Example ??*: Histograms*

Exercise 1.3	
	Use `runif(100)` to draw random numbers and generate histograms with 5, 10, 20, 50 cells of equal size. Use repeated samples. Do the histogram plots correspond to what you expect from independent uniform random variates? Try to note your observations in detail. Repeat the experiment with two cells $(0, 0.5], (0.5, 1)$. `hist(runif(100), breaks = c(0, 0.5, 1))` Repeat the experiment with random numbers generated by `rnorm(100)` and compare the results from `runif(100)` and `rnorm(100)`.

Exercise 1.4	
	Modify Example ?? (page ??) to include the kernel name and the bandwidth used in the kernel density estimation. You have to store the result from `density()` and access its components in analogy to Example ?? (page ??).

Barcharts

1.3.3 Statistics of Distribution Functions; Kolmogorov-Smirnov Tests

Exercise 1.5	
	Using `help(rbeta)` you get information about the functions available to work with beta distributions. Generate plots for the densities of the beta distribution for $n = 16, 32, 64, 128$ and $i = n/4, n/2, 3n/4$. Use the function `curve()` to generate the plots. For more information, see `help(curve)`.

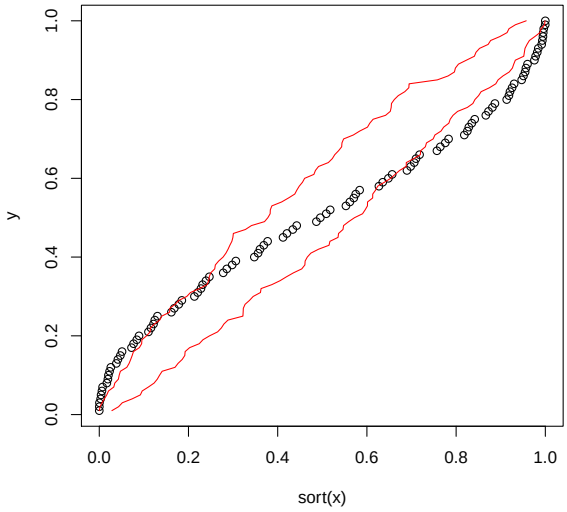
Exercise 1.6	
	Draw the distribution function with the corrected reference line.
*	We use the graphical display for a single sample, not for a run of samples. Is the expected value of $X_{(i)}$ an adequate reference? Are there alternatives that can serve as references? If you see alternatives, give an implementation.

Monte Carlo Confidence Bands

***Example* 1.1: Monte Carlo Confidence Bands**

Input

```
x <- (sin(1:100)+1)/2                    # demo example only
y <- (1:length(x))/length(x)
plot(sort(x), y)
nrsamples <- 19                          # no of simulations
samples <- matrix(data = runif(length(x)* nrsamples),
    nrow = length(x), ncol = nrsamples)
samples <- apply(samples, 2, sort)
envelope <- t(apply(samples, 1, range))
lines(envelope[, 1], y, col = "red")
lines(envelope[, 2], y, col = "red")
```



Exercise 1.7	
	Make use of the `help()`-function and comment on Example 1.1 step by step. Take special note of the new functions that are introduced here.

R ***Iterators***	

(cont.)→

R *Iterators* (cont.)	
`apply()`	applies a function to the rows or columns of a matrix. *Example:* `samples <- apply(samples, 2, sort)` sorts by column.
`outer()`	generates a matrix of all pair-wise combinations of two vectors, and applies a function to all pairs.

Exercise 1.8	
*	Why 19? *Hint:* Try to take an abstract simplified view of the problem first: let T be a measurable function and $X_0, X_1, \ldots, X_{nrsamples}$ independent samples with a common distribution function. What is $P(T(X_0) > T(X_i))$ for all $i > 0$? In a second step, give an abstract formulation for the example above. Then consider the special case `nrsamples = 19`.

Exercise 1.9	**Monte Carlo Coverage**
*	Estimate the coverage probability of the Monte Carlo band by first generating a band as above. (How can you draw the band without first making a plot for a special sample?) Next, generate *sim* simulation samples of uniform random numbers of sample size 100. Count how many simulations give a sample within the band. You have to make your choice of the number *sim* of simulations (100? 1000? 999?) for this step. Use this information to estimate the coverage probability. *Hint:* `any()` can be used to evaluate a comparison for a full vector.

Theorem 1.1 *For all integer n and any positive λ, we have*

$$P(\sqrt{n}\sup|F_n - F| > \lambda) \leq 2e^{-2\lambda^2}.$$

Proof. [3], Corollary 1 □

This inequality is valid even if F is not continuous.

Exercise 1.10	**Finite Sample Bounds**
	Use the inequality given in Theorem 1.1 to calculate bounds for $\sqrt{n}\sup\|F_n - F\|$.
	Add finite sample bands to the empirical distribution function.

Exercise 1.11	
	Using *help(ks.test)* you get information on how to invoke the function *ks.test*. Which results do you expect if you test the following vectors for a uniform distribution? *1:100* *runif(100)* *sin(1:100)* *rnorm(100)* Perform these tests and discuss the results. For the test, scale the values so that they fall into the interval $[0, 1]$, or use a uniform distribution on an interval that is adapted to the data.

1.3.4 Statistics of Histograms and Related Plots; χ^2-Tests

Exercise 1.12	
	Use *help(chisq.test)* to see the calling structure for χ^2 tests. Apply it to test the hypothesis $(p_j = 1/J),\ J = 5$ on the following vectors of bin counts: (3 3 3 3 3) (1 2 5 3 3) (0 0 9 0 6).

Exercise 1.13	
	Which results do you expect if you use a χ^2 test to check the following vectors for a uniform distribution? *1:100* *runif(100)* (cont.)→

Exercise 1.13	(cont.)

	`sin(1:100)`
	`rnorm(100)`
	Perform these tests and discuss the results.
	Hint: The function `chisq.test()` expects a frequency table as input. The function `table()` can be used to generate a frequency table directly (see `help(chisq.test)`). But you can also use the function `hist()`, which gives `counts` as one component of its result.

Exercise 1.14	
*	Sketch comparable test environments for fixed and adaptive choice of histogram cells. For fixed and for adaptive choice of histogram cells draw `s = 1000` samples of size 50 from `runif()`. Calculate in both settings the formal χ^2 statistics and plot its distribution functions. Compare the distribution functions.

Exercise 1.15	
	For $n = 10, 50, 100$, draw 300 samples using `runif(n)`. For each sample, calculate the χ^2 and the Kolmogorov-Smirnov statistic. You have to choose a χ^2 test. What is your choice? Plot the distribution functions of these statistics and compare them to the theoretical (asymptotic) distributions. Are there any indications against the assumption of independent uniform random numbers? *Hint:* The functions for the χ^2 and Kolmogorov-Smirnov test keep their internal information as a list. To get the names of the list elements, you can create a sample object. For example, use `names(chisq.test(runif(100)))`.

Exercise 1.16	
**	Analyse the power of the Kolmogorov-Smirnov test and the χ^2 tests. Select values for n, m and α, and choose 9 pairs for (a, b). What are your arguments for your choices? Use your chosen parameters to draw samples from `rbeta()`. Apply the Kolmogorov-Smirnov test and a χ^2 test with 10 cells of equal size on $(0, 1)$.
	Choose alternative parameters (a, b) so that you can compare the decision rules along the following lines: i) a = b ii) b = 1 iii) a = 1 and run these simulations.
	Choose alternative parameters (a, b) so that you can compare the decision rules over the range $0 < a, b < 5$. Your conclusions? *Hint:* `outer(x, y, fun)` applies a function `fun()` to all pairs of values from x, y and returns the result as a matrix. Using `contour()` you can generate a contour plot. See `demo("graphic")`.

Exercise 1.17	
⋆⋆	Design a test strategy to unmask "pseudo-random numbers". Test this strategy using simple examples i) $x \quad x = 1..100 \mod m$ for convenient m ii) $\sin(x)$ $x = 1..100$ iii) ... Do you tag these sequences as "not random"? Now try to unmask the random number generators provided by R. Can you identify the generated sequences as "not random"?

1.4 Moments and Quantiles

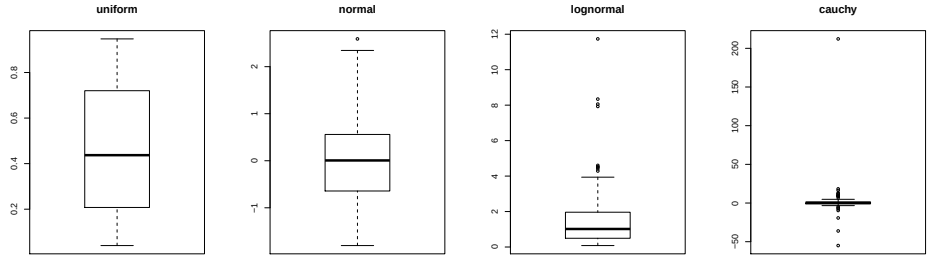
Exercise 1.18	
	Generate a sample of random variables with sample size 100 from the distributions with the following densities: $$p(x) = \begin{cases} 0 & x < 0 \\ 1 & 0 \leq x \leq 1 \\ 0 & x > 1 \end{cases}$$ and $$p(x) = \begin{cases} 0 & x \leq 0 \\ 2 & 0 < x \leq 1/4 \\ 0 & 1/4 < x \leq 3/4 \\ 2 & 3/4 < x \leq 1 \\ 0 & x > 1 \end{cases}$$ Estimate the mean, variance and standard deviation in each of these. Repeat the estimation for 1000 samples. Analyse the distribution of estimated mean, variance and standard deviation for repeated samples.

Exercise 1.19	
	Generate a sample of 100 random variables from the distributions of Exercise 1.18. Estimate the median, and the lower and upper quartiles. Repeat the estimation for 1000 samples. Analyse the distribution of the estimated median, lower and upper quartiles from repeated samples.

***Example* 1.2: Box-and-Whisker Plot**

Input

```
oldpar <- par(mfrow = c(1, 4))
boxplot(runif(100), main = "uniform")
boxplot(rnorm(100), main = "normal")
boxplot(exp(rnorm(100)), main = "lognormal")
boxplot(rcauchy(100), main = "cauchy")
par(oldpar)
```



Exercise 1.20	
	Modify Example 1.2 so that the plots are comparable: adjust the location so that the medians are at the same height. Adjust the scales so that the inter-quartile ranges have same length.

Exercise 1.21	
	For continuous distributions and the median X_{med} we have $$P(X_i \geq X_{med}) = 0.5.$$ Hence we can find a k such that $$k = min\{k : P(X_{(k)} \leq X_{med}) < \alpha\}$$ and $X_{(k)}$ as an upper bound for the median with confidence level $1 - \alpha$. Use this idea to construct a confidence interval for the median with confidence level $1 - \alpha = 0.9$.
	Modify the box-and-whisker plot to show this interval. (cont.)→

Exercise 1.21	(cont.)
	Hint: You need the distribution function F_X, evaluated at the position marked by the order statistic $X_{(k)}$. The distributions of $F_X(X_{(k)})$ are discussed in Theorem **??**.
	The box-and-whisker plot offers an option `notch = TRUE` to mark confidence intervals. Try to use the documentation to find out how a `notch` is calculated. Compare your confidence intervals with those marked using `notch`.
$*$	Use an analogous strategy to get a distribution-independent confidence interval for the inter-quartile range.
$***$	Augment the box-and-whisker plot so that it gives information about the scale in a way that is statistically reliable. *Hint:* Why is it not sufficient to mark confidence intervals for the quartiles?

1.5 R Complements

1.5.1 Random Numbers

1.5.2 Graphical Comparisons

Exercise 1.22	
	Generate a PP plot of the $t(\nu)$ distribution against the standard normal distribution in the range $0.01 \leq p \leq 0.99$ for $\nu = 1, 2, 3, \ldots$.
	Generate a QQ plot of the $t(\nu)$ distribution against the standard normal distribution in the range $-3 \leq x \leq 3$ for $\nu = 1, 2, 3, \ldots$.
	How large must ν be so that the t distribution is barely different from the normal distribution in these plots?
	How large must ν be so that the t distribution is barely different from the normal distribution if you compare the graphs of the distribution functions?

Exercise 1.23	
	Use PP plots instead of distribution functions to illustrate the χ^2- and Kolmogorov-Smirnov approximations.

Exercise 1.24	
	Use QQ plots instead of distribution functions. Can you add confidence regions to these plots with the help of the χ^2- resp. Kolmogorov-Smirnov statistics?

Exercise 1.25	
	Generate a matrix of dimensions $(nrow * ncol - 1), length(x)$ with random numbers and use `apply()` to avoid the loop. *Hint:* See Example 1.1 (page 3).

Exercise 1.26	
	Use `rnorm()` to generate with pseudo-random numbers for the normal distribution for sample size $n = 10, 20, 50, 100$. For each sample, generate a PP plot and a QQ plot, using the theoretical normal distribution as a reference.
	Add Monte Carlo bands from the envelope of 19 simulations. Instead of the uniform distribution, you have to use the normal distribution to generate the Monte Carlo bands. Then you have to represent the results in the coordinate system of the QQ plots, that is, the x axis represents the quantiles of the normal distribution. *Hint:* Inspect the source of `qqnorm()`.
*	The bands are initially bands for the standard normal distribution. Find bands adjusted in scale and location of the data at hand.

1.5.3 Complements: Functions

Exercise 1.27	
	Rework your programming exercises and write reusable parts as functions.

Exercise 1.28	
	Write as functions: • A function `ehist` showing an augmented histogram. • A function `eecdf` showing the empirical distribution. • A function `eqqnorm` showing a QQ plot with the standard normal distribution as comparison. • A function `eboxplot` showing a box-and-whisker plot. and • A wrapper function `eplot` showing a plot matrix with these four plots. Your functions should call the standard functions (or modify them, if necessary) and guarantee that the plots have an adequate complete annotation.

Vectorisation

I

Exercise 1.29	**Vectorisation**
	Write `sqrt0()` as a vectorised function using `ifelse()`.

Compilation

1.5.4 Complements: Enhancing Graphical Displays

Exercise 1.30	
	Use `help(plot)` to inspect the possibilities of customising the plot function. Information on details of the parameters is only available if you use `help(plot.default)`. Modify your latest plot so it has a correct main title.

1.5.5 Complements: R *Internals*

Executing Files

1.5.6 Search Paths, Frames and Environments

1.6 Additional Exercises

Exercise 1.31	**Feature Detection**
	This series of example tries to judge feature detection sensitivity of various displays for univariate data. for a preparation, find (and fix) a display arrangement that is convenient for your display, (for example `par( mfrow=c(5,4))`. Select a false detection rate you are willing to tolerate (for example, $2/20 = 10\%$).
*	Write a function `plotdens <- function(n)` that draws n normal random numbers for each of the display frames. Find a number n_{sym} so that with $n \geq n_{sym}$ observations most results appear symmetric (i.e. the non-symmetric samples are below tolerance rate. Find a number $n_{unimodal}$ so that with $n \geq n_{unimodal}$ observations most results appear unimodal (i.e. the multimodal samples are below tolerance rate.
	Modify your function by adding an additional parameter `plotdens <- function(n, generator=rnorm)` that allows to select a random number generator. For the following distributions, find a sample size that allow detection of the given features reliably within tolerance. Generator | Features uniform | flat density lognormal | unimodal, skewed, long tail Cauchy | symmetric, frequent outliers, long tail.

Exercise 1.32	**Distribution**
	Prepare a plot with one display frame showing a test sample, the others showing uniform random samples. What is the required sample size to identify a normal distribution with an error rate below tolerance level? With this sample size, what is the false detection rate if you start with a uniform sample.
	Exchange the roles. Use a uniform sample as a test sample, and normal samples for comparisons. What is the required sample size to identify a uniform distribution with an error rate below tolerance level? With this sample size, what is the false detection rate if you start with a gaussian sample.

Exercise 1.33	
	Repeat the example above using a QQ-Plot.
	Repeat the example above using a histogram.

1.6.1 Complements: Packages

Using Packages

Building Packages

Exercise 1.34	
	Install the functions from Exercise 1.28 as a package. You can prepare the package with `package.skeleton()`, if you have already defined the functions. Load the package. Verify that you can still load the package with `library()` if you have restarted the R system. *Hint:* For an object x, the statement `prompt(x)` generates a skeleton upon which you can build a documentation for `x`.

Compilation

1.7 Statistical Summary

1.8 Literature and Additional References

[4] R Development Core Team (2000–2008): Writing R Extensions.
See: <http://www.r-project.org/manuals.html>.

[5] Shorack, G. R.; Wellner, J. A.: *Empirical Processes with Applications to Statistics.* Wiley, New York, 1986.

[1] Gänßler, P.; Stute, W.: *Wahrscheinlichkeitstheorie.* Springer, Heidelberg, 1977.

[2] Gentleman, R.; Ihaka, R.: Lexical Scope and Statistical Computing. *Journal of Computational and Graphical Statistics* 9 (2000) 491–508.

References

[1] Peter Gänßler and Winfried Stute. *Wahrscheinlichkeitstheorie*. Springer, Heidelberg, 1977.

[2] Robert Gentleman and Ross Ihaka. Lexical scope and statistical computing. *Journal of Computational and Graphical Statistics*, 9:491–508, 2000.

[3] Paul Massart. The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. *The Annals of Probability*, 18(3):1269–1283, July 1990.

[4] R Development Core Team. *Writing R Extensions*, 2008.

[5] Galen R. Shorack and Jon A. Wellner. *Empirical Processes with Applications to Statistics*. Wiley, New York, 1986.

Subject Index