Final assignment
Advanced R for Life Sciences 2018

This is an individual assignment; do not confer with your fellow students. Plagiarism is fraud, don't cut and paste

Make sure your code is properly formatted and documented, and that it works.

If something seems unclear, you can approach Henk van den Toorn (h.w.p.vandentoorn@uu.nl).

Task 1 of 2 (30 pt)

Knocking out a gene (conditionally or constitutively) often leads to changes in the transcriptional programme of a cell. We have studied this quite extensively in yeast, and found that there is a clear relationship with growth rate (more 'damaging' deletions lead to slower growth). What is interesting is that there is a common denominator to the expression changes of slow-growing strains, which we have called the 'slow growth profile'. The more a gene expression profile resembles this slow growth profile, the slower it grows in practice. Your task is to visualize and test this relationship.

The file data.txt contains the relative gene expression changes (profiles) of 6109 yeast genes (rows), in 13 different deletion strains (columns). The values are log2(deletion / wildtype).

The file slowgrowthprofile.txt contains the profile of slow-growing cells. That is, transcripts with a high value are typically upregulated in slow-growing strains, transcripts with a low value are typically downregulated in such cases.

The file reldoublingtime.txt contains measurements of the relative growth rate, expressed as log2(relative doubling time). For strains growing more slowly than wildtype yeast, this value is > 0. Note that there are more relative doubling times available than gene expression data, so you will have to extract and order the correct subset of this time data.

Create a plot that shows the relationship between *'resemblance to the slow growth profile'* and *'relative doubling time of the strain'*. Create a script that reads the data, makes the plot with include a linear regression line, and writes the correlation, the p-value, and what is the equation of the regression line to the screen. Hand in the script.

The *'resemblance to the slow growth profile'* is calculated as the covariance between that profile, and the actual expression profile of a particular strain. Use the function `cov` for this. (it also accepts complete matrices).

Task 2 of 2 (70 pt)

We will be building (a) a package, and (b) a shiny server to load and display a TCGA data example file. The package contains functions to normalize our data. The shiny server will use these functions, and display the data in a user-friendly way.

The data

This is part of the table:

	MUT.AKT	CNA.AKT	RPPA.AKT	RNA.AKT	RNAZ.AKT	MUT.BRAF	CNA.BRAF	RPPA.BRAF	RNA.BRAF	RNAZ.BRAF	MUT.
TCGA.3N.A9WB.06	NA	0.015	NA	6380.856	0.2806	NA	-0.013	NA	47.2536	-1.4275	NA
TCGA.3N.A9WC.06	NA	-0.580	NA	2640.972	-1.2975	NA	0.003	NA	103.1818	-0.6712	NA
TCGA.3N.A9WD.06	NA	0.077	NA	5296.454	-0.1770	NA	0.049	NA	112.0567	-0.5512	NA
TCGA.BF.A1PU.01	NA	NA	1.13259500	7891.234	0.9180	V600E	NA	-0.04333331	60.0649	-1.2543	NA
TCGA.BF.A1PV.01	NA	NA	NA	4926.778	-0.3330	NA	NA	NA	138.9719	-0.1872	NA
TCGA.BF.A1PX.01	NA	NA	0.51587610	5558.916	-0.0662	V600E	NA	-0.39676597	93.5002	-0.8021	NA
TCGA.BF.A1PZ.01	NA	NA	1.14743100	7128.867	0.5963	NA	NA	-0.55964910	80.3606	-0.9798	NA
TCGA.BF.A1Q0.01	NA	NA	NA	1149.713	-1.9268	NA	NA	NA	413.2770	3.5223	NA
TCGA.BF.A3DJ.01	NA	NA	-0.11449920	2341.548	-1.4239	V600E	NA	1.10421595	112.0110	-0.5518	NA
TCGA.BF.A3DL.01	NA	NA	0.96861180	15285.470	4.0381	V600K	NA	1.32100038	103.8961	-0.6615	NA
TCGA.BF.A3DM.01	NA	NA	0.86584390	7338.245	0.6846	V600K	NA	1.73690940	255.9257	1.3944	NA
TCGA.BF.A3DN.01	NA	NA	1.67705900	5838.057	0.0516	V600E	NA	0.74740599	283.8710	1.7723	NA
TCGA.BF.A5EO.01	NA	NA	NA	14692.467	3.7879	NA	NA	NA	60.3715	-1.2501	NA
TCGA.BF.A5EP.01	NA	NA	NA	8082.904	0.9988	NA	NA	NA	483.3863	4.4704	NA
TCGA.BF.A5EQ.01	NA	NA	NA	4388.761	-0.5600	V600K	NA	NA	130.3160	-0.3043	NA
TCGA.BF.A5ER.01	NA	NA	NA	10076.280	1.8400	NA	NA	NA	175.1637	0.3022	NA

Column names: you see the following prefixes, before each gene name

MUT	what is the mutation in that sample, if any
CAN	copy number
RPPA	Protein expression level
RNA	RNA expression level
RNAZ	RNA expression level z-transformed

Columns to the right contain clinical data about the patients. Each row is a sample from a patient.

Package (35 pt)

Create the package **tcgatools**. In this package, create the following functionality:

- Extract the indexes of the columns where a prefix of choice exists (e.g. get all columns that start exactly with 'RPPA.')
- Return a data frame with only selected columns normalized, based on median and mad
- Extract the columns of a subtype

Make sure you only make the relevant functions visible. Take care of the 'one thing per function' rule. Check the package, and fix anything that's wrong. The data should *not* be part of the package! If you have tested the package, build and install the package locally, so you can use it from the shiny server.

To hand in this part of the assignment, create a source package (under the build menu)

Shiny server (40 pt)

Create a Shiny graphical user interface for inspecting the data. The GUI shall contain the following functionality. The tcgatools package should be loaded and used to perform the tasks in the UI.

- An upload possibility, so that the file can be uploaded
- Box plots of an expression level of choice (RPPA, RNA or RNAZ)
- Plots are done with ggplot
- The possibility to normalize the data (eg. By clicking a button under the plot)
- You can create Scatter plots, with the possibility to use the clinical data to make different colors or to split the plot

Scoring:

yeast analysis

runs without errors (7 pts)
generates correct output (8 pts)
well-structured code (8 pts)
comments (7 pts)

tcgatools package

runs without errors (5 pts)
generates correct output (5 pts)
correct package structure and documentation (8pts)
well-structured code (6pts)
comments (6pts)

Shiny server
runs without errors (5 pts)
generates correct output (7 pts)
plot is pretty (e.g. different colors, labels and titles) (7pts)
useful application of clinical data (7pts)
well-structured code (7pts)
comments (7pts)

	Unsatisfactory	Satisfactory	Good	Excellent
Coding Standards	• Disorganized and messy • Poor use of variables (confusing naming and local/global errors)	• Some organization in the code • Good use of variables (local/global variable, meaningful names)	• Well-structured code • Good use of variables (local/global variable, meaningful names)	• Well-structured code with enough indentation/space. • Coding standards are beyond expectations • Good use of variables (local/global variable, meaningful names)
Documentation & Comments	• No comments • No documentation with the code	• Basic documentation including descriptions of the main variables • Purpose is noted for each function	• Well-documented code with rich comments • Specific purpose is noted for each function	• Well-documented code with rich comments • Specific purpose is noted for each function and control structure • All functions can return a Help-message consisting of the correct command line usage for the program, including the list of the available command-line options
Runtime	• Does not execute due to errors • Output is not clear/pretty • Execution is abnormally slow	• Executes without errors • Output is clear but not pretty	• Executes without errors • Output is clear and pretty	• Executes without errors • Output is clear and pretty • Execution is fast
Efficiency	• Overcomplicated and inefficient solution	• A logical solution that is easy to follow but not the most efficient	• Solution is efficient and easy to follow	• Solution is very elegant and easy to understand