Examination Advanced R Programming

Linköpings Universitet, IDA, Statistik

Course code and name:	732A94 Advanced R Programming
Date:	2018/12/10, 8–12
Teacher:	Krzysztof Bartoszek
Allowed aids:	The extra material is included in the zip file **exam_material.zip**
Grades:	A= $[18-20]$ points
	B= $[16-18)$ points
	C= $[12-16)$ points
	D= $[9-12)$ points
	E= $[8-9)$ points
	F= $[0-8)$ points
Instructions:	Name your solution files as:
	[your exam account]_[own file description].[format]
	The R code should be complete and readable code, possible to run by copying directly into a script. Comment directly in the code whenever something needs to be explained or discussed. Follow the instructions carefully.
	There are **THREE** problems (with sub–questions) to solve.

Problem 1 (5p)

a) (2p) Provide the names of R's object–oriented systems. Very briefly write how they differ between each other.
b) (3p) The behaviour of many functions, e.g. `print()` or `plot()` is different when applied to different types of input. How is this achieved?

Problem 2 (10p)

READ THE WHOLE QUESTION BEFORE STARTING TO IMPLEMENT! Remember that your functions should **ALWAYS** check for correctness of user input!

a) (3p) In this task you should use object oriented programming in S3 or RC to write code that manages a library's books.

The first task is to implement a function called `build_library()` that returns an object corresponding to the library. The `build_library()` function should take one numeric argument: `book_capacity`. The object should contain a data structure that contains what books are present in the library. A book is described by its title, author and it has to possess a **unique ID** that you have to assign **yourself**. Furthermore, a book has to have a status variable saying if it is borrowed or available. The library object should have two fields: `number_of_books` (the number of books in possession of the library) and `number_of_loaned` (the number of books that have been borrowed). At creation the value of both should be `0`. Of course the number of books cannot exceed `book_capacity`.

```
## S3 and RC call to build_library() function
my_library <- build_library(book_capacity=100)
```

b) (3p) Now implement a function called `acquire_book()` to add books to the library. The function should take two arguments, the title of the book and its author. This user provided data should then be remembered in the data structure that stores the library's contents. Run the function a number of times (say 120 times) to add books to the library. Remember to update the **number_of_books** field. You decide yourself how to react when the number of books exceeds `book_capacity` (but of course the library cannot accept books then). The titles and authors of all the books can be anything, including the same for all the books.

```
## S3 and RC call
my_library <-acquire_book(my_library,"title","author")

## if using RC you may also call in this way
## my_library$acquire_book("title","author")
```

c) (3p) Now implement two functions called `borrow_book()` and `return_book()`. Both functions should take as an argument the book's ID. The functions should check if the library has the given book in its possession, if it has not been already borrowed and react appropriately. Do not forget to update the field `number_of_loaned` and also the book's status. Run the function a number of times (say 20 times) to borrow and return books.

Provide some example calls to your code.

d) (1p) Implement a plot **OR (NO NEED TO DO BOTH!)** print function to present the books the library has in its possession.

Problem 3 (5p)

a) (3p) In multivariate statistics a key matrix algebra operation is calculating the value of the *quadratic form* for a given square, $p \times p$, matrix $\mathbf{\Sigma} \in \mathbb{R}^p \times \mathbb{R}^p$ and vector of length p, $\vec{x} \in \mathbb{R}^p$. The formula for the value of the quadratic form is

$$\vec{x}^T \mathbf{\Sigma} \vec{x} = \sum_{i=1}^{p} \sum_{j=1}^{p} \vec{x}[i] \mathbf{\Sigma}[i,j] \vec{x}[j],$$

Write your own function that takes as its input a matrix and vector and returns the value of the quadratic form. Do not forget that your function should check for correctness of input and react appropriately.
b) (1p) What is the complexity of your solution in terms of the number of required multiplication operations?
c) (1p) The same can be achieved using the `%*%` operator in R code, i.e. the value of the quadratic form will be calculated as `x%*%Σ%*%x`. Implement a unit test that compares your implementation with direct R matrix–vector multiplication.