A pratical tutorial to S4 programming

Laurent Gatto*

Cambridge Center for Proteomics
University of Cambridge

June 20, 2013

*lg390@cam.ac.uk

Contents

1 Introduction

This document introduces the `R` object-oriented programming paradigm using the microarray as a use case. The introduction is purely practical and does not aim for an exhaustive guide to `R` object-oriented programming. We will concentrate on the S4 system and only mention the older S3 system and the recent S4 reference class infrastructure. See the appropriate literature, `?ReferenceClasses` or our more thorough introduction to OO programming[1] and references therein for mote details.

In section 2, we present a solution on how to represent microarray data in `R` using simple data types and conclude with some issues with this implementation. In section 3, we introduce fundamental concepts of OO programming and introduce how OO programming is implemented in S4 (and S3) system.

2 The microarray example

We assume the reader is familiar with the concept of microarrays and the type of data that is obtained from such an experiment. Before embarking in any serious programming task, in particular when modelling data and defining data structures (using a OO class or not), to carefully think about how to best represent and store the data.

Exercise 1: Based on your understanding of microarrays and the kind of data that is to be used to computational analysis, think of what is going to be needed to describe the experiment and what the types of data structures available in `R` (`data.frame`, `matrix`, `vector`, ...) are most appropriate. Ideally, one would want everything (data, meta-data, ...) to be stored together as a single variables.

There are of course multiple valid solutions to the above question. Below are three pieces of information that consider essential along with their respective `R` data

[1] https://github.com/lgatto/roo

structure.

- We choose to represent the microarray probes

3 Using OO programming

Object-oriented programming is based on two important concepts, abstraction and encapsulation.

4 The `MArray` class

5 `MArray` methods

Session information

All software and respective versions used to produce this document are listed below.

- R Under development (unstable) (2013-06-16 r62969), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_GB.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_GB.UTF-8`, `LC_COLLATE=en_GB.UTF-8`, `LC_MONETARY=en_GB.UTF-8`, `LC_MESSAGES=en_GB.UTF-8`, `LC_PAPER=C`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_GB.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: knitr 1.2
- Loaded via a namespace (and not attached): digest 0.6.3, evaluate 0.4.3, formatR 0.7, stringr 0.6.2, tools 3.1.0