# S3 Classes and Methods in R

Reza Mohammadi

June 17, 2013

university of groningen

faculty of mathematics and natural sciences

## Classes and Methods in R

R has two systems of classes and methods:

- **S3** are informal classes.
  'S3' classes and methods for the S language were introduced and have been implemented in R. Many R functions, such as *print*, *plot* and *summary*, are S3 generics.
- **S4** are formal classes.
  Recommended for advanced users.
  Only used in some packages e.g. Matrix.

## CLASSES AND METHODS IN R

- **Class** is the definition of an object. (See *?Classes*)

  ```
  > x <- matrix(1, 5, 5)
  > class(x)
  [1] "matrix"
  ```

- **Method** is a function that performs specific calculations on objects of a specific class. (See *?Methods*)
- Functions such as *print*, *plot*, and *summary* adapt their action according to different type of objects. They are known as **generic** functions.

## Classes and Methods in R

- Classes and methods allow users to connect new objects with old, familiar functions.
- When we create complex object from new function, creating new class with methods can improve usability of the function and results.

## Creating S3 Classes

To create a new class, simply assign a new class to an object before returning it from a function

```
functionName = function(input){
                  .
                  .
                  .
  class(output) <- "className"
  return(output)
  }
```

## Building S3 Methods

- Creating new class called "Class"

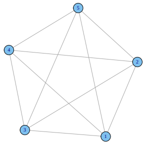
```
> generate = function(p = 5){
+   G                  <- matrix(0, p ,p)
+   G[upper.tri(G)] <- 1
+   class(G)           <- "graph"
+   return(G)
+   }
```

- Suppose we want to allow users to apply "Method" to an object of class "Class"

## Building S3 Methods

- We create a new function called "Method.Class", which R will then invoke whenever "Method" is applied to an object of class "Class"

```
> plot.graph = function(x, ...){
+   G <- graph.adjacency(x, mode = "undirected")
+   plot.igraph(G, ...)
+   }
> x <- generate(p = 5)
> class(x)
> [1] graph
> plot(x)
```



## USING "..." IN GENERIC FUNCTIONS

The "..." argument is often used in generic functions likes *print*, *summary*, and *plot*

```
> plot.graph = function(x, ...){
+     G <- graph.adjacency(x)
+     plot.igraph(G, ...)
+ }
>
> plot(x)
> plot(x, layout = layout.circle)
```

## Advantages and disadvantages

### Advantages

- Users can apply familiar R functions to new objects
- Saves the user time in finding or visualizing important information

### Disdvantages

- Using methods for classes (especially for *print*) takes the user one step away from the true R object.

Tip : to learn about R object we can use :

```
> str(objName) # prints summary information
```

## USEFUL LINKS

http://www.ci.tuwien.ac.at/Conferences/useR-2004/Keynotes/Leisch.pdf

http://cran.r-project.org/doc/Rnews/Rnews_2003-1.pdf