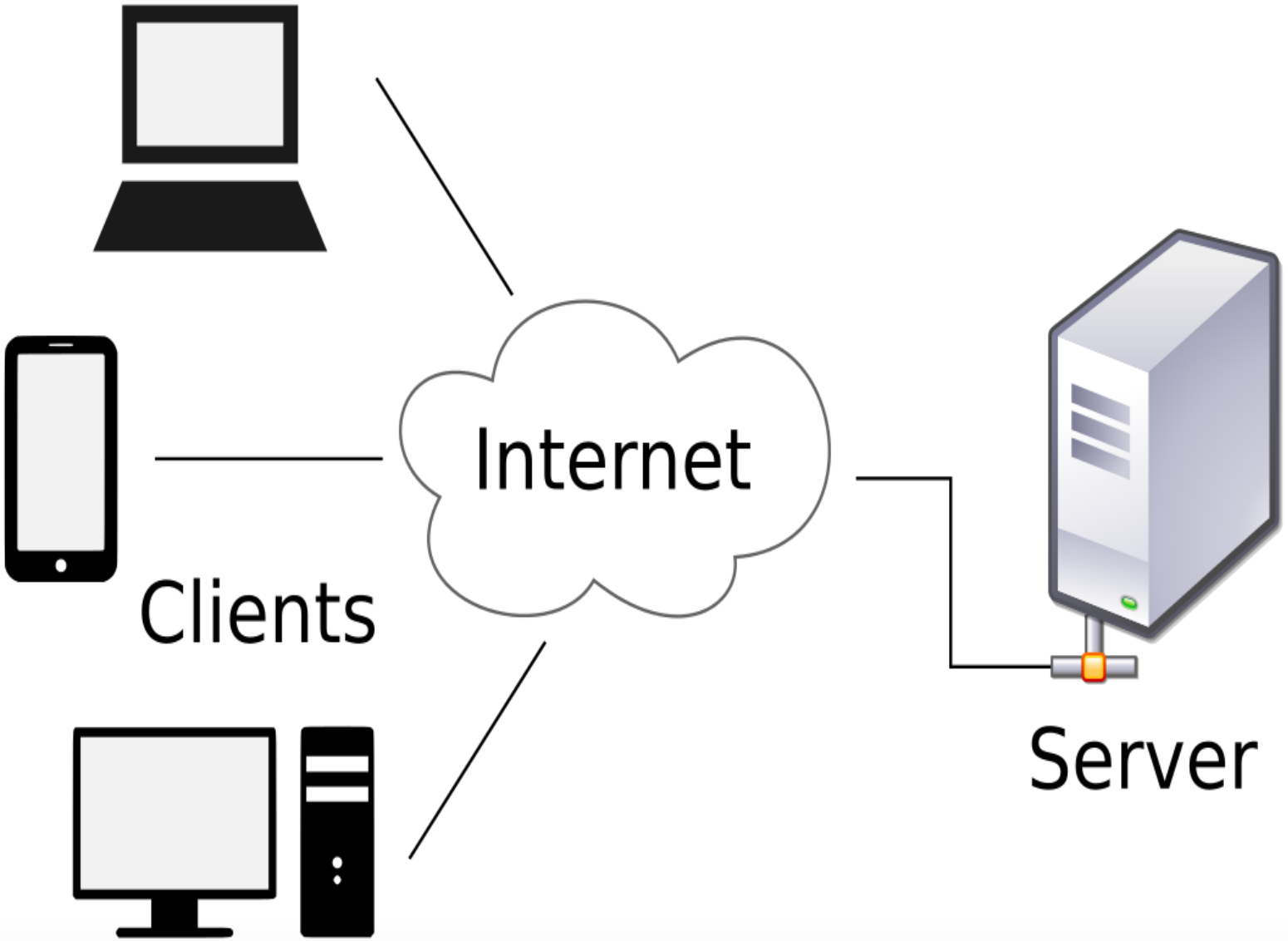
Connections

Philipp A.Upravitelev
upravitelev@gmail.com

2017-12-02

Databases


Internet
Clients
Server

r connectors

```
library('RMySQL')
con <- dbConnect(MySQL(max.con = 100, fetch.default.rec = 1000),
                 user = "students",
                 password = "pmsar2017",
                 dbname = "pmsar",
                 host = "mysql.cyvwua6jlfyv.us-west-2.rds.amazonaws.com",
                 port = 3306)
```

```
library('RMySQL')
```

```
## Loading required package: DBI
```

```
con <- dbConnect(MySQL(max.con = 100, fetch.default.rec = 1000),
                 user = "students",
                 password = "pmsar2017",
                 dbname = "pmsar",
                 host = "mysql.cyvwua6jlfyv.us-west-2.rds.amazonaws.com",
                 port = 3306)
```

dbGetQuery

```
# dbWriteTable(con, 'hp_spells', spells,  row.names = FALSE)
# dbSendQuery(con, 'drop table hp_spells')
# dbWriteTable(con, 'got_chars', got_chars, row.names = FALSE)

dbGetQuery(
  conn = con,
  statement = 'select * from hp_spells limit 5')
```


```
# dbWriteTable(con, 'hp_spells', spells,  row.names = FALSE)
# dbSendQuery(con, 'drop table hp_spells')
# dbWriteTable(con, 'got_chars', got_chars, row.names = FALSE)

dbGetQuery(
  conn = con,
  statement = 'select * from hp_spells limit 5')
```


```
##   incantation                    effect  type spell_length spell_multi
## 1       Accio          Summons an object Charm            5           0
## 2   Aguamenti     Shoots water from wand Charm            9           0
## 3   Alohomora       Opens locked objects Charm            9           0
```

sql example

```
dbGetQuery(
  conn = con,
  statement =
    "select
        type,
        count(*)
    from hp_spells
    where incantation rlike ' '
    group by type
    order by type")
```

```
##    type count(*)
## 1 Charm        3
## 2 Curse        2
## 3 Spell       14
```

programming

conditions

- if()
- ifelse()
- switch()

loops

- for()
- while()
- repeat()

lapply

- lapply()
- lapply() in data.table

Examples

```
res <- rnorm(n = 1000, mean = 0, sd = 1)
```

```
mx <- max(res)
```

```
qplot(res, geom = "histogram") + theme_minimal()
```

Existing functions

```
range.default
```

```
## function (..., na.rm = FALSE, finite = FALSE)
## {
##     x <- c(..., recursive = TRUE)
##     if (is.numeric(x)) {
##         if (finite)
##             x <- x[is.finite(x)]
##         else if (na.rm)
##             x <- x[!is.na(x)]
##         return(c(min(x), max(x)))
##     }
##     c(min(x, na.rm = na.rm), max(x, na.rm = na.rm))
## }
## <bytecode: 0x000000000be67ea0>
## <environment: namespace:base>
```

Creating functions

```
my_fun <- function(<arguments>) {
## Do something interesting
}
```

Structure of function's body

```
func_example <- function(x) {
  res <- rnorm(n = x, mean = 0, sd = 1)
  mx <- max(res)
  mx_squared <- mx ^ 2
  return(mx_squared)
}
```

```
func_example <- function(x) max(rnorm(n = x, mean = 0, sd = 1)) ^ 2
```

List in the result

```
func_example <- function(x) {
  res <- rnorm(n = x, mean = 0, sd = 1)
  mx <- max(res)
  mx_squared <- mx ^ 2
  res_list <- list("true_mean" = 0,
                   "sample_mean" = mean(res),
                   "sample_max" = mx,
                   "max_squared" = mx_squared)
  return(res_list)
}
```

```
tmp <- func_example(1000)
```

```
class(tmp)
```

```
## [1] "list"
```

List in the result pt.2

```
str(tmp)
```

```
## List of 4
##  $ true_mean  : num 0
##  $ sample_mean: num -0.0556
##  $ sample_max : num 3.69
##  $ max_squared: num 13.6
```

```
tmp[4]
```

```
## $max_squared
## [1] 13.62801
```

```
tmp$max_squared
```

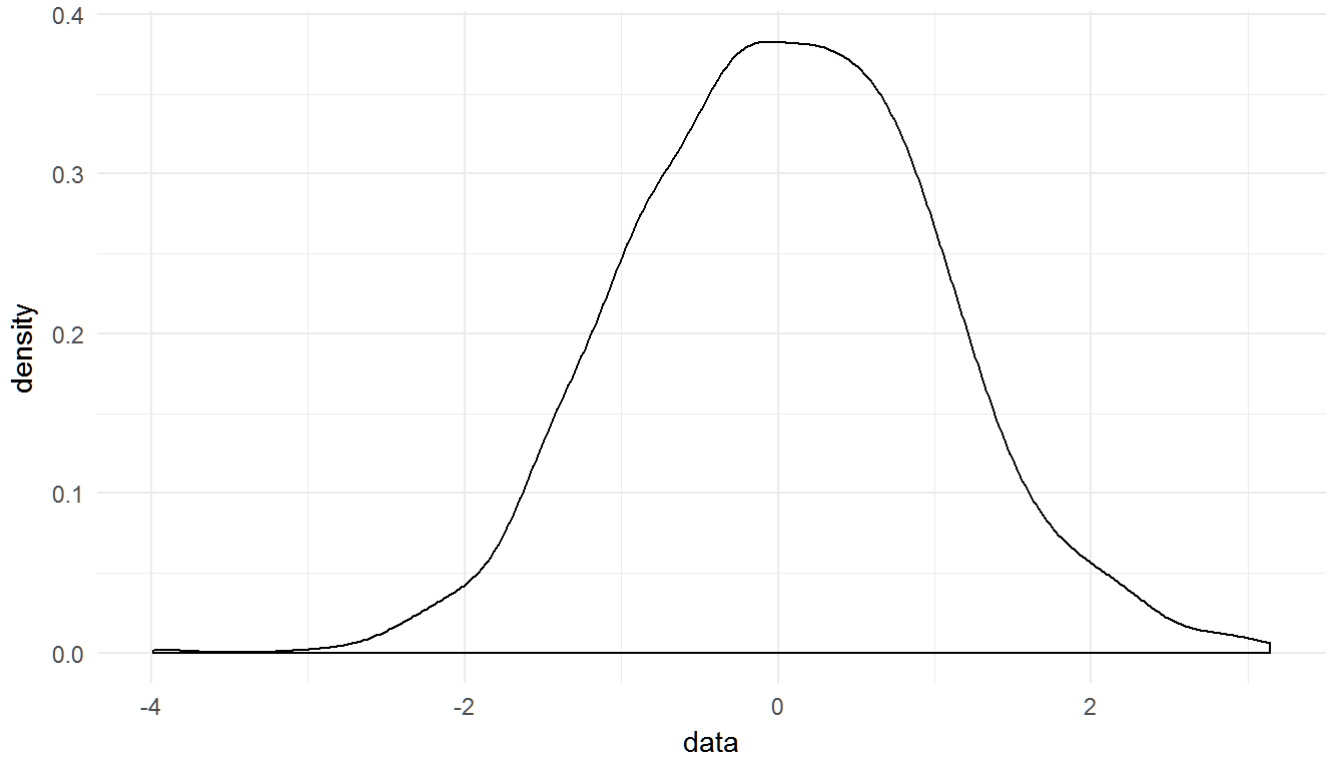
```
## [1] 13.62801
```

Default arguments

```
plot_rnorm <- function(x, type = "gaussian") {
  data <- switch(type,
                 "gaussian" = rnorm(n = x, 0, 1),
                 "uniform" = runif(n = x, 0, 1),
                 "lognormal" = rlnorm(n = x, 0, 1),
                 stop("unknown distribution type"))
  qplot(data, geom = "density") + theme_minimal()
}
```

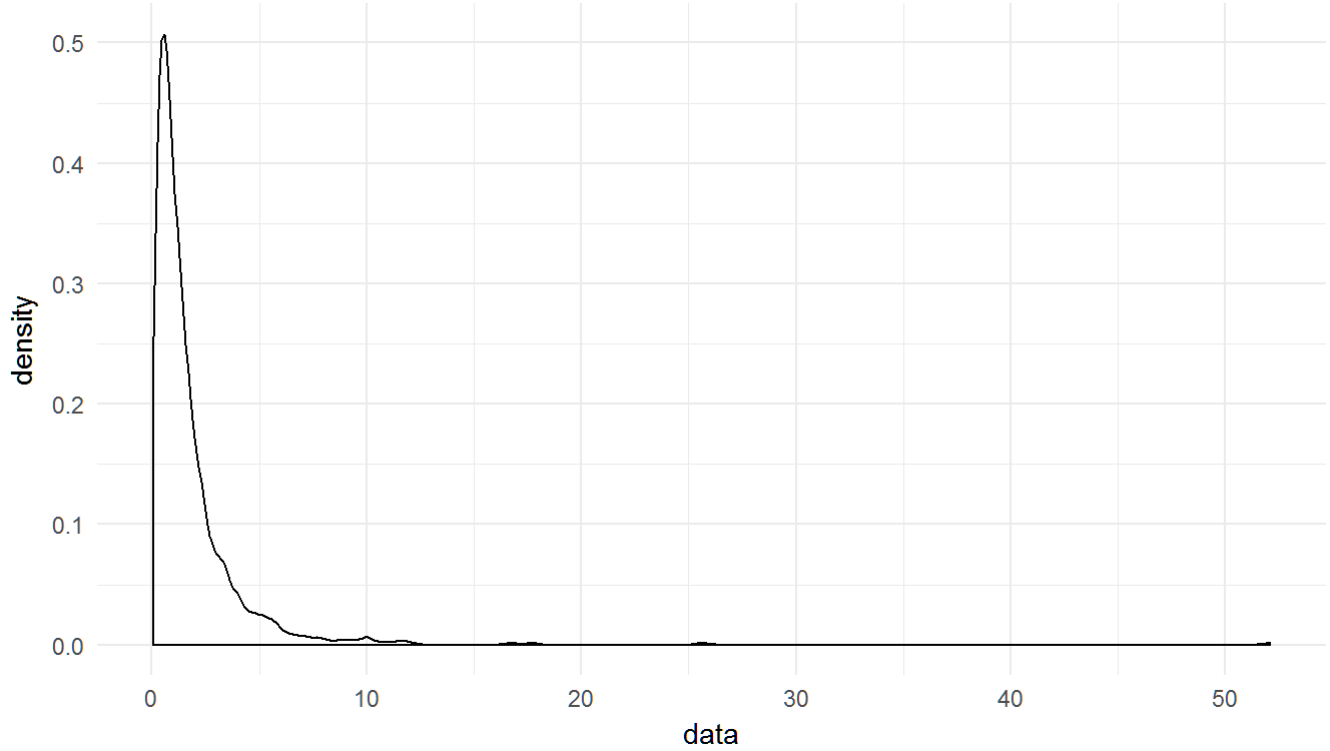
Default arguments pt.2

```
plot_rnorm(1000)
```



Default arguments pt.3

```
plot_rnorm(1000, type = "lognormal")
```



Default arguments pt.4

```
plot_rnorm(1000, type = "blabla")
```

```
## Error in plot_rnorm(1000, type = "blabla"): unknown distribution type
```

Anonymous functions

```
lapply(data, function(arglist)  { body })
```

Example of anonymous

```
nicknames <- c("Nagibator", "Дохлый Гуру", "KILLER QUEEN", "ДеффачКа")

nicknames_simplifier <- function(x) {
  temp <- tolower(x)
  temp <- gsub("\\s", "_", temp)
  temp
}
```

```
temp <- nicknames_simplifier(nicknames)
temp
```

```
## [1] "nagibator"    "дохлый_гуру"  "killer_queen" "деффачка"
```

Example of anonymous pt.2

```
lapply(nicknames, function(x) {
  temp <- tolower(x)
  temp <- gsub("\\s", "_", temp)
  temp
})
```

```
temp <- lapply(nicknames, function(x) gsub("\\s", "_", tolower(x)))
unlist(temp)
```

```
## [1] "nagibator"    "дохлый_гуру"  "killer_queen" "деффачка"
```

Markdown

Code chunks

```{r chunk, eval = F, include=T, wrapper = T}

you_script

```