R_intro_2

Karel Fišer, 2017

Last intro

R

```
>
> x <- 2
> x <- c(1, 2, 3)
> x[2]
> 5 + x
> mean(x)
> q()
```

Create vectors

```
> c(2, 4, 6) # (c)ombine elements into a vector.

> 2:6                    # Sequence in a range.
> seq(2, 6, by = 2)      # Sequence in a range.
> seq(2, 6, by = 0.5)

> rep(1:2, times = 3)
> rep(1:2, each = 3)

> rnorm(3)
> # sample(1:10, size = 10, replace = TRUE)
```

Use functions to work with vectors

```
> x <- c(1, 2, 3, 5, 4, 4)

> sort(x)    # Return x sorted.
> table(x)   # See counts of values.
> rev(x)     # Return x reversed.
> unique(x)  # See unique values.
```

And do not forget to assign the result to a variable.

Select vector elements

By position:

```
> x[4]       # The fourth element.
> x[-4]      # All but the fourth.
> x[2:4]     # Elements two to four.
> x[-(2:4)] # All elements except two to four.
> x[c(1, 5)]# Elements one and five.
```

The position of element is its **index**.

Select vector elements

By value:

```
> x[x == 4]
> x[x < 4]
> x[x %in% c(1, 2, 5)]
```

Conditions:

```
> a == b      # are equal
> a != b      # are NOT equal
> a > b       # greater than
> a >= b      # greater than or equal to
```

Vectors – content types

```
> x <- c(2, 10, 3)        # numeric
> y <- c("ball", "dog")   # character
> z <- c(TRUE, TRUE)      # logical
> factor_x <- factor(x)   # factor (categorical)
```

or integer

… mind the coercion logical > integer > numeric > caracter

```
# factor is integer vector with attributes
```

Your turn

create vectors of all types

Variables – dimensions

1D – atomic vector, list

2D – matrix, data frame

nD – array

Variables – homogeneity

homogenous – vector, matrix, arrays

heterogenous – list, data frame

Variable constructors

```
> # atomic vector, same types, 1d:
> x <- c(2, 10, 3)

> # matrix, same column lengths, same types, 2d:
> m <- matrix(1:20, nrow=5,ncol=4)

> # data frame, same column lengths, 2d:
> d <- data.frame(c("blue", "red"), c(4, 8))

> # list, 1d (list is a vector, but possibly
  branching):
> l <- list(3, c(TRUE, FALSE))
```

Rectangular data

Pretty much all of the data we deal with is rectangular: has columns and rows.

Statistically, we want variables and observations.

Variables go in columns, observations go in rows (with occasional exceptions)

Hadley Wickham

Your turn

create variables other than vectors

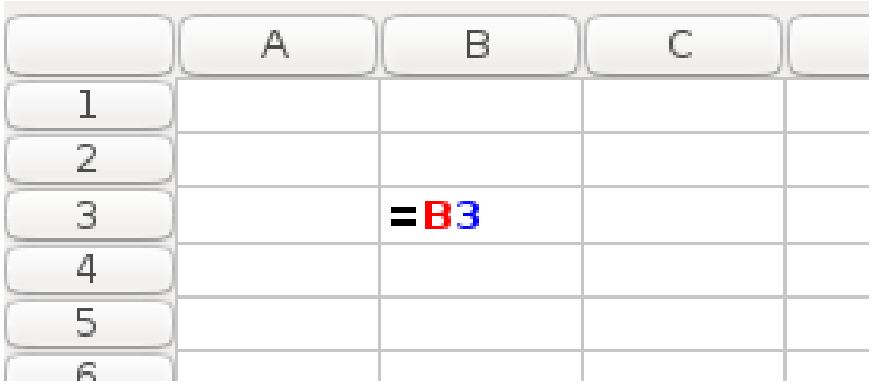
Indexing

select elements by position

```
> x[1]
> x[3:6]
```

```
  m[rows , columns]
```

```
> m[3, 2]
> m[1, 2:4]
> m[1, c(2,4)]
> m[2, ]
> m[,-3]
```



Named variables

```
> x <- c(a=2, b=10, c=3)
> names(x)



> d <- data.frame(colour=c("blue", "red"),
  legs=c(4, 8))
> colnames(d)
> rownames(d)
> rownames(d) <- c("cat", "spider")
> d
```

Select elements by names

```
> d[, 2]
> d[, "legs"]
> d["cat",]
```

access column by **d$column_name** notation

```
> d$legs # only for columns
> d$legs[2]
```

Indexing list

```
> goulash <- list(1, second=2, "dog", 4,
  c(50, 51), 3:7, last=list(three=c(1, 2,
  3), veggie="Paprika"))


> str(goulash)


> goulash[5]          # returns list
> goulash[[5]]        # returns list content
> goulash[[5]][1]
> goulash$last$three[2]
```

Your turn

- sum third element of x with
  first value in second column of d

- same using names

Exploring variables

```
> length(x)


> d2 <- data.frame(a=1:1000, b=1000:1, c=11:1010)
> dim(d2)
> str(d2)
> head(d2)
> tail(d2, n=1)


> class(x)
> is.data.frame(x)
```

Altering and combining

```
> rownames(d) <- c("cat", "spider")
> d[2, 2] <- 7
> d[,1] <- c("ginger", "black")


> lives <- c(9, 1)
> d[, 3] <- lives
> d <- data.frame(d, lives)
> d <- cbind(d, lives)
```

Variable manipulation

```
> t(m)
> c(x, 2:5); cbind(x, z); rbind(x, z)
> rbind(m[1,], x) # repeating shorter vector!


> z <- c(1, 2, 3)
> class(z)
> z <- as.character(z)
> class(z)


> # missing values are NA in R
> data[data=="NotDetermined"] <- NA
```

iris dataset

```
> ?iris
```

This famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

```
> data() # datasets in R
> str(iris)
> data <- iris
```

Ordering

data frame:

```
> data
```

a column of data frame:

```
> data$Sepal.Length
```

indexes of ordered vector:

```
> order(data$Sepal.Length)
```

ordered column of data frame (vector):

```
> data$Sepal.Length[order(data$Sepal.Length)]
```

ordered data frame:

```
> data[order(data$Sepal.Length),]
```

Ordering

```
> data[order(data$Sepal.Length),] # ordered
  data.frame

> # order by Sepal.Length (ascending) and
  Sepal.Width (descending):
> data[order(data$Sepal.Length,
  -data$Sepal.Width),]

> sort(data$Sepal.Length) # returns sorted
  vector, not its indices in sorted order
```

Variable manipulation

```
> # change col order:
> data <- data[, c(5, 1:4)]


> # new vector based on two columns:
> Sepal_square <-
  data$Sepal.Length*data$Sepal.Width


> # add new variable/col to the data:
> data <- cbind(data, Sepal_square)
```

Finding patterns

```
> unique(data$Species)

> # find pattern "virginica":
> which(data$Species=="virginica")
> grep("virginica", data$Species)

> # find and replace:
> gsub("virginica", "sp.", data$Species)
> data$Species <- gsub("virginica", "sp.",
  data$Species)
```

Your turn

- make "Petal_square" column and sort data on it

- change all "setosa" to "set" and "versicolor" to "ver"

Genes in human genome

get all genes in org.Hs.eg.db

```
> library(Homo.sapiens)
> genes <- select(Homo.sapiens, keys =
  keys(Homo.sapiens, keytype = "GENENAME"),
  columns = c("SYMBOL","TXCHROM"), keytype =
  "GENENAME")
> dim(genes)
[1] 61117     3
```

Kinases in human genome

select genes with “kinase” in their name

```
> kinases <- genes[grep("kinase", genes$GENENAME), ]
> dim(kinases)
[1] 977   3
```

Kinases on chromosomes

how many kinases on each chromosome

```
> kinases_on_chr <- table(kinases$TXCHROM)
> kinases_on_chr[order(kinases_on_chr)]
> tail(kinases_on_chr[order(kinases_on_chr)],
  5)
```

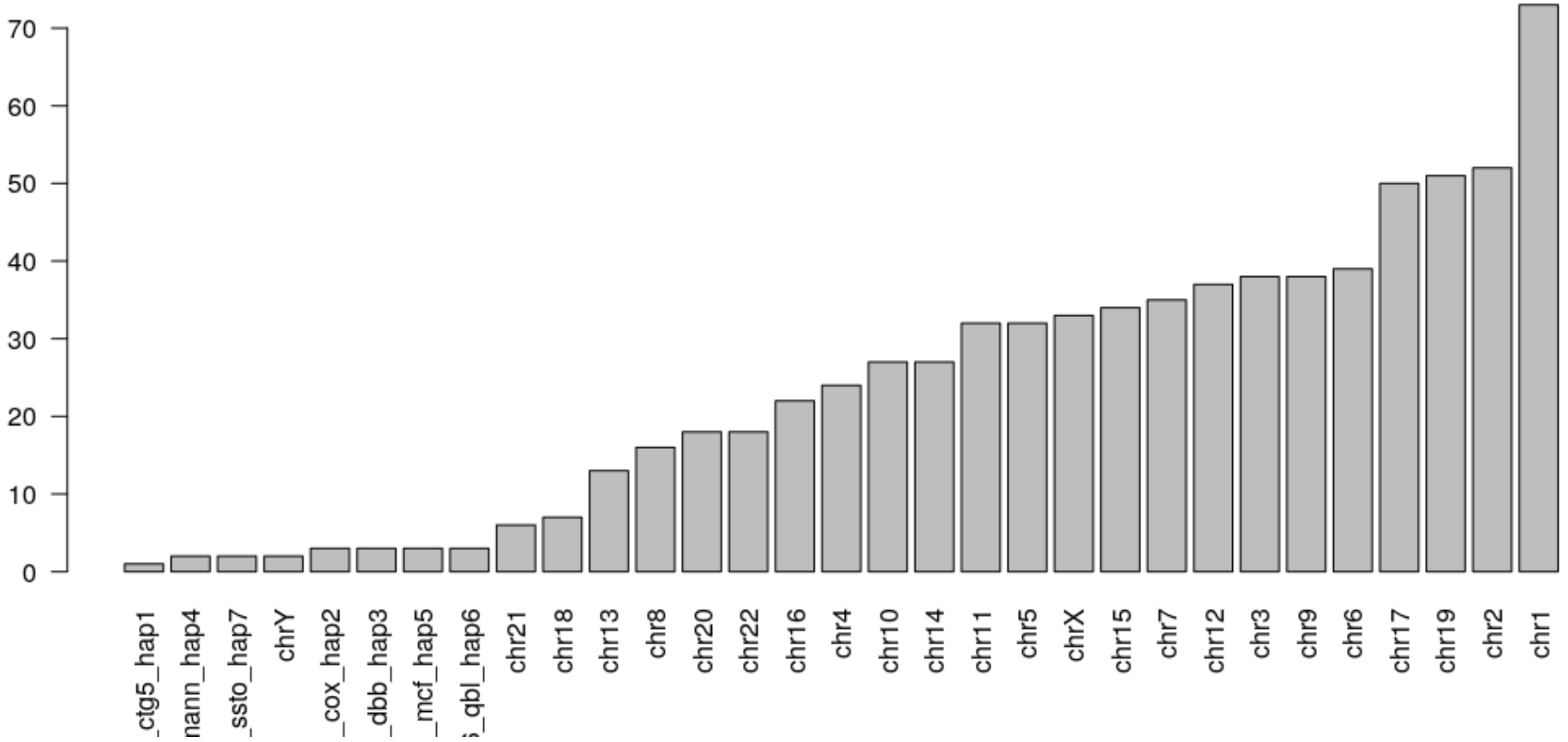
```
chr6 chr17 chr19  chr2  chr1
  39    50    51    52    73
```

Kinases on chromosomes

5 lines of code …

```
> library(Homo.sapiens)
> genes <- select(Homo.sapiens, keys = keys(Homo.sapiens, keytype =
  "GENENAME"), columns = c("SYMBOL","TXCHROM"), keytype = "GENENAME")
> kinases <- genes[grep("kinase", genes$GENENAME), ]
> kinases_on_chr <- table(kinases$TXCHROM)
> barplot(kinases_on_chr[order(kinases_on_chr)], las = 2)
```

Kinases on chromosomes



http://www.bioinformin.net