Univerzitet u Beogradu – Elektrotehnički fakultet (ETF)
Katedra za signale i sisteme

Tehnike obrade biomedicinskih signala 19M051TOBS

Dr Nadica Miljković, vanredni profesor
kabinet 68, nadica.miljkovic@etf.bg.ac.rs

VRSTE PODATAKA I OSNOVNE FUNKCIJE

Topovi podataka u R-u

- Tipovi podataka u R-u se dele na:
  - Atomske/osnovne klase objekata (ima ih 5): numeričke, logičke, karaktere, *integer* (celobrojne) i kompleksne;
  - nizove, liste;
  - faktore;
  - nedostajuće vrednosti;
  - *data frame*-ove i matrice.
- Svi R objekti mogu imati atribute koji se koriste da se opiše sadržaj (fizički/matematički smisao objekta).
- Neki od atributa (npr. dimenzija) mogu menjati svojstva R objekata.

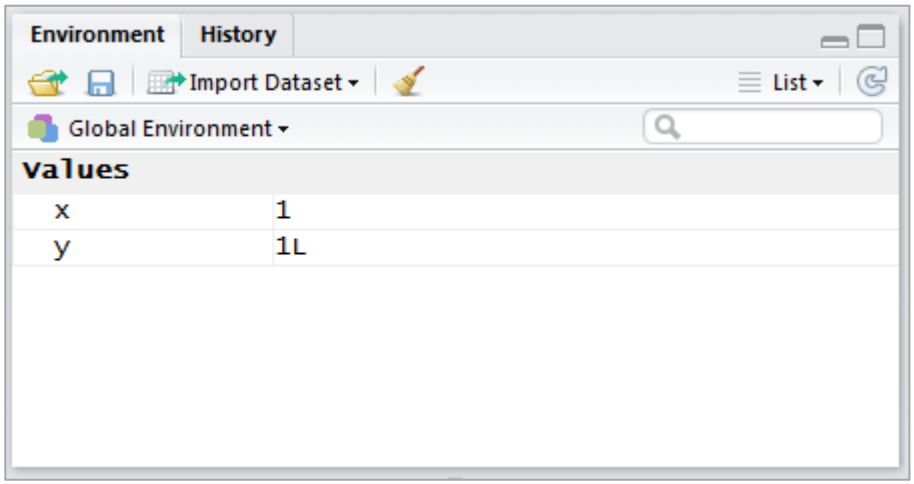
Atributi

- Metapodaci (eng. *metadata)* u R-u su atributi. Svaki R objekat može imati atribute.
- Atributi mogu biti:
  - imena, imena dimenzija;
  - dimenzije (nizova, matrica, ...);
  - klasa (numerička, *integer*, karakter, ...);
  - dužina;
  - drugi metapodaci koje definiše korisnik.
- Da bi se pogledali metapodaci neke promenljive/objekta može se koristiti funkcija *attributes()*.
- Ako metapodaci ne postoje funkcija će vratiti vrednost NULL.

Numeričke promenljive

```
> x <- 1
> x
[1] 1
> print(x)
[1] 1
> y <- 1L
> y
[1] 1
> print(y)
[1] 1
> 1 / 0
[1] Inf
> 1 / Inf
[1] 0
> 0 / 0
[1] NaN
>
```

- Brojevi su u R-u predstavljeni preko numeričkih promenljivih, odnosno svi brojevi su predstavljeni kao realni.
- Ako neki podatak mora eksplicitno biti unet kao celobrojna veličina tj. *integer*, onda je potrebno uneti i oznaku za *integer* (L).
- U R-u postoji kao broj (numerički podatak) definisan i Inf, pa je deljenje nulom moguća operacija! Postoji takođe i NaN (eng. *Not A Number*) i može se dobiti kao rezultat pojedinih operacija.

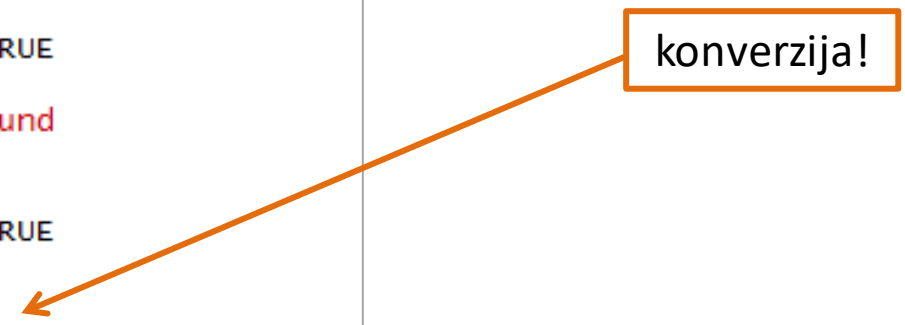


Nizovi

- U R-u postoji i niz (eng. *vector*) kao tip podataka.
  - Niz se može napraviti pomoću funkcije *vector()* – uključujući i prazan niz.
- Da li postoji i neki drugi način da se napravi niz?
  - Da, primenom *c()* funkcije.
- Nizovi mogu sadržati isključivo elemente koji pripadaju istim klasama.
- Za razliku od nizova, lista (eng. *list*) može sadržati i elemente koji pripadaju različitim klasama.
- ...

Kreiranje nizova

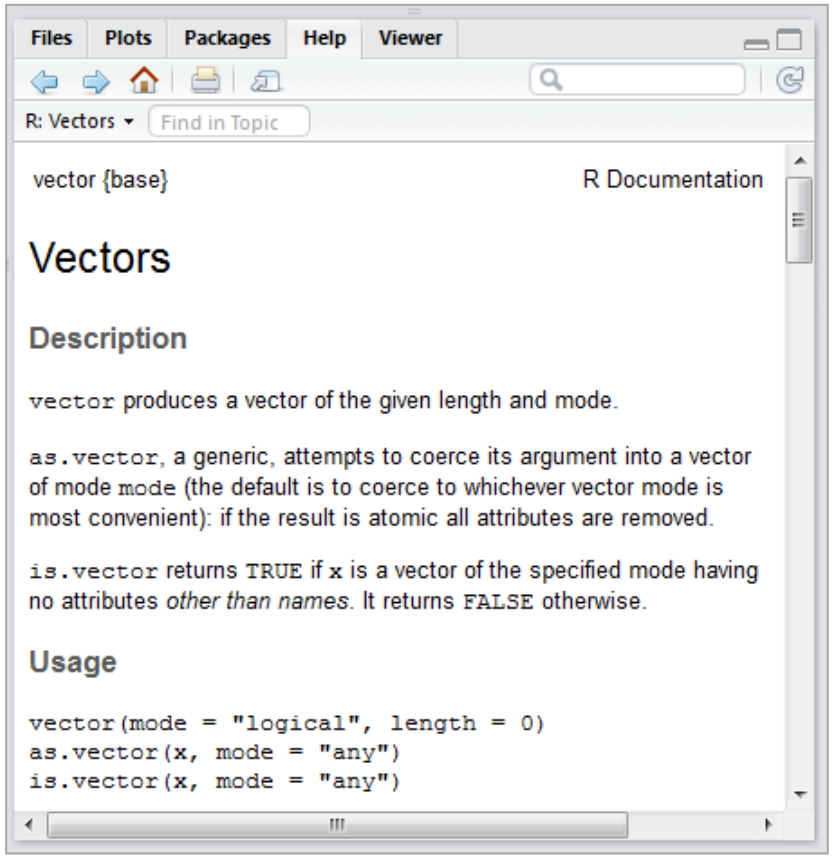
```
> c(1, 2, 3, 4)
[1] 1 2 3 4
> x <- c("jedan", "dva", "tri", "četiri")
> x
[1] "jedan"  "dva"    "tri"    "četiri"
> y <- c(TRUE, TRUE, FALSE, TRUE)
> y
[1]  TRUE  TRUE FALSE  TRUE
> Y
Error: object 'Y' not found
> y1 <- c(T, T, F, T)
> print(y1)
[1]  TRUE  TRUE FALSE  TRUE
> z <- c(TRUE, 1, "dva")
> z
[1] "TRUE" "1"    "dva"
>
```



- Najčešće korišćena funkcija (pored *vector()* funkcije za inicijalizaciju niza) za kreiranje nizova je *c()* od eng. *combine*.
- NAPOMENA: Bolje je koristiti TRUE umesto T, ali ponekad je ugodnije imati validan kraći zapis *boolean* promenljive.
- **Primetiti da su *Y* i *y* dve različite promenljive.**

Funkcija *vector()*

```
> vector("numeric", length = 10)
 [1] 0 0 0 0 0 0 0 0 0 0
> vector("numeric", 10)
 [1] 0 0 0 0 0 0 0 0 0 0
>
```



- Na slici je prikazano kako se funkcija *vector()* može koristiti za incijalizaciju/kreiranje nizova.
- Kako proveriti koji su ulazni parametri funkcije *vector()*?
  - *HINT*: koristiti automatsko popunjavanje koda ili “?” operator

Konverzija podataka

```
> c(1, 2, 3, 4)
[1] 1 2 3 4
> x <- c("jedan", "dva", "tri", "četiri")
> x
[1] "jedan"  "dva"    "tri"    "četiri"
> y <- c(TRUE, TRUE, FALSE, TRUE)
> y
[1]  TRUE  TRUE FALSE  TRUE
> Y
Error: object 'Y' not found
> y1 <- c(T, T, F, T)
> print(y1)
[1]  TRUE  TRUE FALSE  TRUE
> z <- c(TRUE, 1, "dva")
> z
[1] "TRUE" "1"    "dva"
>
```


implicitna konverzija

- Kada se više objekata različitih klasa pojavi unutar jednog niza, onda dolazi do konverzije, jer
  - nizovi su predstavljeni samo sa objektima koji pripadaju istoj klasi podataka.
- Ako dođe do “mešanja” podataka različitih klasa, kao na slici, onda dolazi do implicitne konverzije (eng. *implicit coercion*).
- **Logic -> Numeric -> Character**

Konverzija podataka

```
> x <- 1:4
> x
[1] 1 2 3 4
> as.logical(x)
[1] TRUE TRUE TRUE TRUE
> as.character(x)
[1] "1" "2" "3" "4"
> as.complex(x)
[1] 1+0i 2+0i 3+0i 4+0i
> y <- c("p", "r", "o", "g", "a", "m")
> y
[1] "p" "r" "o" "g" "a" "m"
> as.complex(y)
[1] NA NA NA NA NA NA
Warning message:
NAs introduced by coercion
> |
```

upozorenje prilikom eksplicitne konverzije

- Ako postoji potreba za konverzijom podataka i ako korisnik želi da ima kontrolu nad tom konverzijom onda se koristi eksplicitna konverzija (eng. *explicit coercion*).
- Za ove potrebe koristi se funkcija **as.** kao što je prikazano na slici.
- Ako postoji nekakav "problem" prilikom konverzije (R nema dovoljan broj podataka na osnovu kojih može izvršiti konverziju), kao rezultat eksplicitne konverzije može se javiti NAs rezultat (+ upozorenje/*Warning*).
- Da bi se za neki objekat proverilo kojoj klasi pripada koristi se *class()* funkcija.

Matrice (funkcija *matrix()*)

```
> x <- matrix(nrow = 3, ncol = 2)
> x
     [,1] [,2]
[1,]   NA   NA
[2,]   NA   NA
[3,]   NA   NA
> dim(x)
[1] 3 2
> attributes(x)
$dim
[1] 3 2

> x1 <- matrix(1:6, nrow = 3, ncol = 2)
> x1
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> 
```

- Matrice u R-u su nizovi koji imaju kao atribut/metapodatak dimenziju.
- Dimenzija (eng. *dimension*) je niz *integer*-a dužine 2 (broj vrsta i broj kolona). Pogledati upotrebu funkcije *attributes()* na slici.
- Ako bi se pristupalo pojedinim elementima, treba imati na umu da su matrice *column-wise*. Šta to znači? Znači da se prvo popunjavaju kolone, pa tek onda redovi (pogledati kod na slici).
- Matrice se mogu kreirati i iz nizova dodavanjem atributa dimenzije, ali se mogu kreirati i "spajanjem" po vrstama i/ili kolonama.

Drugi načini za inicijalizaciju matrice

```
> x2 <- 1:30
> print(x2)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
> dim(x2)
NULL
> dim(x2) <- c(5, 6)
> print(x2)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    6   11   16   21   26
[2,]    2    7   12   17   22   27
[3,]    3    8   13   18   23   28
[4,]    4    9   14   19   24   29
[5,]    5   10   15   20   25   30
> 
```

```
> x3 <- 1:4
> print(x3)
[1] 1 2 3 4
> y3 <- 10:13
> print(y3)
[1] 10 11 12 13
> cbind(x3, y3)
     x3 y3
[1,]  1 10
[2,]  2 11
[3,]  3 12
[4,]  4 13
> rbind(x3, y3)
   [,1] [,2] [,3] [,4]
x3    1    2    3    4
y3   10   11   12   13
> 
```

- Funkcija *dim()* za rezultat daje dimenziju nekog elementa, ali ona može i dodeliti dimenziju nekom elementu (nizu, matrici i sl.).
- *cbind()* i *rbind()* funkcije omogućavaju kreiranje matrice iz nizova.

Liste

```
> x4 <- list("2017", TRUE, 2, 3 + 7i)
> print(x4)
[[1]]
[1] "2017"

[[2]]
[1] TRUE

[[3]]
[1] 2

[[4]]
[1] 3+7i
```

```
> x5 <- vector("list", length = 5)
> x5
[[1]]
NULL

[[2]]
NULL

[[3]]
NULL

[[4]]
NULL

[[5]]
NULL

> x6 <- vector("list", 6)
>
```

- Lista je vrsta niza koja može sadržati elemente različitih klasa.
- Liste se mogu kreirati pomoću *list()* funkcije, a moguće je kreirati i praznu listu pomoću *vector()* funkcije.
- Kada se koriste liste? Primer u biomedicinskom inženjerstvu? Podaci o ispitaniku (godine, pol, obrazovanje, biomarkeri …)

Faktori – kategoričke promenljive

```
> radniDani <- factor(c("ponedeljak", "utorak", "sreda", "četvrtak", "petak"))
> radniDani2 <- factor(c('ponedeljak', 'utorak', 'sreda', 'četvrtak', 'petak'))
> print(radniDani)
[1] ponedeljak utorak     sreda      četvrtak   petak
Levels: četvrtak petak ponedeljak sreda utorak
> radniDani2
[1] ponedeljak utorak     sreda      četvrtak   petak
Levels: četvrtak petak ponedeljak sreda utorak
> table(radniDani)
radniDani
  četvrtak      petak ponedeljak      sreda     utorak
         1          1          1          1          1
> unclass(radniDani)
[1] 3 5 4 1 2
attr(,"levels")
[1] "četvrtak"   "petak"      "ponedeljak" "sreda"      "utorak"
> radniDani2 <- factor(c('ponedeljak', 'utorak', 'sreda', 'četvrtak', 'petak'), levels = c('ponedeljak', 'uto
rak', 'sreda', 'četvrtak', 'petak'))
> radniDani2
[1] ponedeljak utorak     sreda      četvrtak   petak
Levels: ponedeljak utorak sreda četvrtak petak
> |
```

redosled nivoa po abecednom redu

sa argumentom *levels* se menja redosled nivoa

- Faktori se koriste da bi se predstavili podaci podeljeni u kategorije (eng. *cathegorical data*).
- Kao *integer* koji ima oznaku... Gde se koriste ovakve vrste podataka? Primer je prikazan na slici: dani u sedmici ili bilo koji drugi kategorički podaci.
- Da bi se inicijalizovali faktori koristi se *factor()* funkcija. Može i *as.factor()*.
- NAPOMENA: Prilikom učitavanja podataka iz nekog fajla, uobičajeno je da R podatke koji "liče" na faktore predstavi automatski ovim tipom podataka.

Nedostajuće vrednosti

```
> merenja <- c(1, 0.9, NA, 0.95, 1.01)
> is.na(merenja)
[1] FALSE FALSE  TRUE FALSE FALSE
> is.nan(merenja)
[1] FALSE FALSE FALSE FALSE FALSE
> merenja2 <- c(NaN, 0.9, NA, 0.95, 1.01)
> is.na(merenja2)
[1]  TRUE FALSE  TRUE FALSE FALSE
> is.nan(merenja2)
[1]  TRUE FALSE FALSE FALSE FALSE
>
```

- Nedostajuće vrednosti se u R-u označavaju sa NA ili NaN.
- Gde nedostajuće vrednosti mogu biti od koristi? U principu, one nisu korisne, ali su neizbežne. Nekada mogu poslužiti za opis skupa podataka koji se analizira.
- Pravila koja se odnose na nedostajuće vrednosti su:
  - NA vrednosti mogu imati svoju klasu (mogu biti karakteri, *integer* i sl.) i
  - NaN vrednost jeste NA, ali konverzija iz NA –> NaN nije moguća
- Funkcije za testiranje nedostajućih vrednosti su *is.na()* i *is.nan()* i njihov rezultat je logička (*boolean*) promenljiva.

Nedostajuće vrednosti

Photo by Irina on Unsplash

Nedostajuće vrednosti i analiza podataka

- Uglavnom predstavljaju problem.
- Ne mogu se izbeći.
- Mora se problem rešiti sa pažnjom.
- Posebno ih treba pažljivo analizirati/isključiti/obraditi u kliničkim ispitivanjima.
- Potrebno je utvrditi razlog nedostajućih podataka, ali u najvećem broju slučajeva se analiza završava njihovim opisom i planom tretmana.
- Više u Kang, H. (2013). The prevention and handling of the missing data. *Korean journal of anesthesiology*, *64*(5), 402. doi: 10.4097/kjae.2013.64.5.402 i u Oblaković, M., Sokolovska, V., & Dinić, B. (2015). Tretmani nedostajućih podataka. *Primenjena psihologija*, *8*(3), 289-309.
- Postoji više razloga zašto podaci nedostaju:
  - MCAR (eng. *Missing Completely at Random*) – u potpunosti slučajno raspodeljeni nedostajući podaci, npr. ako su izgubljeni prilikom prenosa (snaga testa je manja, ali ne utiče na rezultat analize)
  - MAR (eng. *Missing at Random*) – slučajno raspodeljeni nedostajući podaci, nedostajanje podataka zavisi od neke varijable u skupu podataka
  - MNAR (eng. *Missing Not at Random*) – podaci koji nedostaju po slučajnom rasporedu, a u vezi su sa analizom i rezultatom istraživanja. Ovo može biti vrlo nezgodno i mora se uvesti model.

Kako tretirati nedostajuće vrednosti?

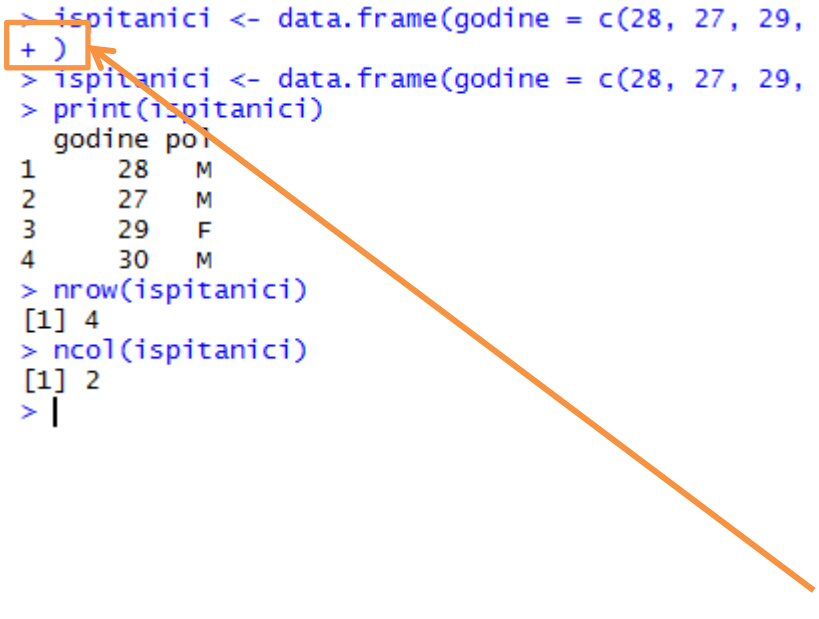
- Isključivanje nedostajućih vrednosti
  - Može biti vrlo kompleksno.
  - Koliko promenljivih imate o jednom pacijentu?
  - Šta tačno nedostaje, a šta se briše?
  - Zato postoji isključivanje u celini (eng. *listwise deletion*) i isključivanje po parovima (eng. *pairwise deletion*)
- Zamena srednjom vrednošću (eng. *mean substitution*)
- Zamena pomoću regresije (eng. *regression imputation*) – koriste se postojeće vrednosti da se napravi predikcija i onda se na osnovu predikcije popunjavaju nedostajuće vrednosti. Mora se pažljivo primeniti i treba imati na umu da je metoda prilično kritikovana.
- Zamena slučajnim vrednostima (eng. *hot deck imputation*) – najčešće se koristi zamena vrednošću date varijableiz nasumično odabranog slučaja izbora
- Zamena poslednjom izmerenom vrednošću (eng. *last observation carried forward*) – najčešće kod merenja ili analize vremenskih serija
- I druge metode

Data frame tip podataka

- Za skladištenje tabelarnih podataka u R-u se koristi *data frame*.
- Postoji posebno dizajniran paket dplyr (https://github.com/hadley/dplyr) koji omogućava pojednostavljen rad sa podacima tipa *data frame*. O ovom paketu kasnije ...
- *Data frame* je lista u kojoj svaki element liste ima jednaku dužinu.
- Za razliku od matrica i nizova, ali slično kao i sa listama, *data frame* može imati podatke koji pripadaju različitim tipovima atomskih promenljivih.
- Atributi (opisi promenljivih u vrstama) se najčešće dodeljuju svakom *data frame*-u.
- Funkcije koje se najčešće koriste za rad sa ovim tipom podataka su:
  - *row.names()*
  - *read.table()*
  - *read.csv()*
  - *data.frame()*
- Da bi se *data frame* konvertovao u matricu koristi se funkcija *data.matrix()*, ali i *as.matrix()*.
- **U obradi signala *data frame* podaci se najčešće koriste!**

Data frame podaci

```
> ispitanici <- data.frame(godine = c(28, 27, 29, 30), pol = factor(c('M', 'M', 'F', 'M'))
+ )
> ispitanici <- data.frame(godine = c(28, 27, 29, 30), pol = factor(c('M', 'M', 'F', 'M')))
> print(ispitanici)
  godine pol
1     28   M
2     27   M
3     29   F
4     30   M
> nrow(ispitanici)
[1] 4
> ncol(ispitanici)
[1] 2
> 
```



Šta znači oznaka “+”?

Nazivi promenljivih i objekata

```
> IDsubjekta = 1:4
> IDsubjekta <- 1:4
> names(IDsubjekta)
NULL
> names(IDsubjekta) <- c("ID1", "ID2", "ID3", "ID4")
> names(IDsubjekta)
[1] "ID1" "ID2" "ID3" "ID4"
>
```

- Za povećanu čitljivost koda i razumevanje učitanih (ili simuliranih) podataka, međurezultata i rezultata, poželjno je koristiti atribut imena promenljivih.
- Za to se koristi funkcija *names()*. Ova funkcija služi i za proveru atributa neke promenljive, ali i za dodeljivanje atributa imena (slično kao i za atribut dimenzije).
- Liste mogu imati imena, ali mogu i matrice.

Atributi matrica

```
> matrica <- matrix(1:6, nrow <- 2, ncol <- 3)
> matrica <- matrix(1:6, nrow = 2, ncol = 3)
> dimnames(matrica) <- list(c("vrsta1", "vrsta2"), c("kolona1", "kolona2"))
Error in dimnames(matrica) <- list(c("vrsta1", "vrsta2"), c("kolona1",  :
  length of 'dimnames' [2] not equal to array extent
> dimnames(matrica) <- list(c("vrsta1", "vrsta2"), c("kolona1", "kolona2", "kolona3"))
> matrica
       kolona1 kolona2 kolona3
vrsta1       1       3       5
vrsta2       2       4       6
> print(matrica)
       kolona1 kolona2 kolona3
vrsta1       1       3       5
vrsta2       2       4       6
> colnames(matrica) <- c("k1", "k2", "k3")
> rownames(matrica) <- c("v1", "v2")
> matrica
   k1 k2 k3
v1  1  3  5
v2  2  4  6
>
```

- Matrice mogu imati imena i kolona i vrsta.
- U tu svrhu se koristi funkcija *dimnames()*, ali mogu se koristiti i funkcije *colnames()* i *rownames()*.
- Zašto je došlo do greške u kodu sa slike?

UČITAVANJE PODATAKA

Funkcije za “čitanje” podataka

- *read.table()*, *read.csv()* – koju vrstu promenljive daju ove funkcije kao rezultat “čitanja” podataka? (***data.frame*!**) Ove funkcije služe za čitanje tabelarnih podataka ≠ *write.table()*
- *readLines()* – čita bilo koji dokument, ali kao rezultat daje tekst ≠ *writeLines()*
- *source() ≠ dump()* – koristi se za R kod – učitavanje koda, najčešće se koristi za učitavanje funkcija.
- *dget() ≠ dput()* – učitavanje R objekata
- *load()* - učitavanje binarnih podataka ≠ *save()*
- *unserialize()* (- || -) ≠ *serialize()* – ovo su *low-level* funkcije. (Ovde ih nećemo koristiti!)

read.table() funkcija

- Ova funkcija se najčešće koristi za čitanje tabu(e)larnih podataka.
- Ulazni parametri ove funkcije su:
  - *file*: ime fajla (uključujući *http*, *source*),
  - *header*: T/F vrednost,
  - *sep*: skup karaktera (string) koji označava kako su razdvojene kolone,
  - *colClasses*: tip podataka,
  - *nrows*: broj vrsta u skupu podataka,
  - *comment.char*: karakter koji označava komentar,
  - *skip*: broj linija koje treba "preskočiti" na početku čitanja fajla i
  - *stringsAsFactors*: da li bi trebalo promenljive tipa karaketra kodovati kao faktore (kategorijske promenljive) i ovaj parametar je tipa *Boolean*.
- Postoje i drugi ulazni parametri.
- Moguće je podesiti globalno neki od parametara, a ne pri svakom pozivu ove funkcije: za to se koristi komanda *options(stringAsFactors = FALSE)*.
- Prilikom korišćenja *read.csv()* funkcije treba primetiti da argumenti imaju drugačije podrazumevane (eng. *default*) vrednosti!

read.table() funkcija

Data Input

Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

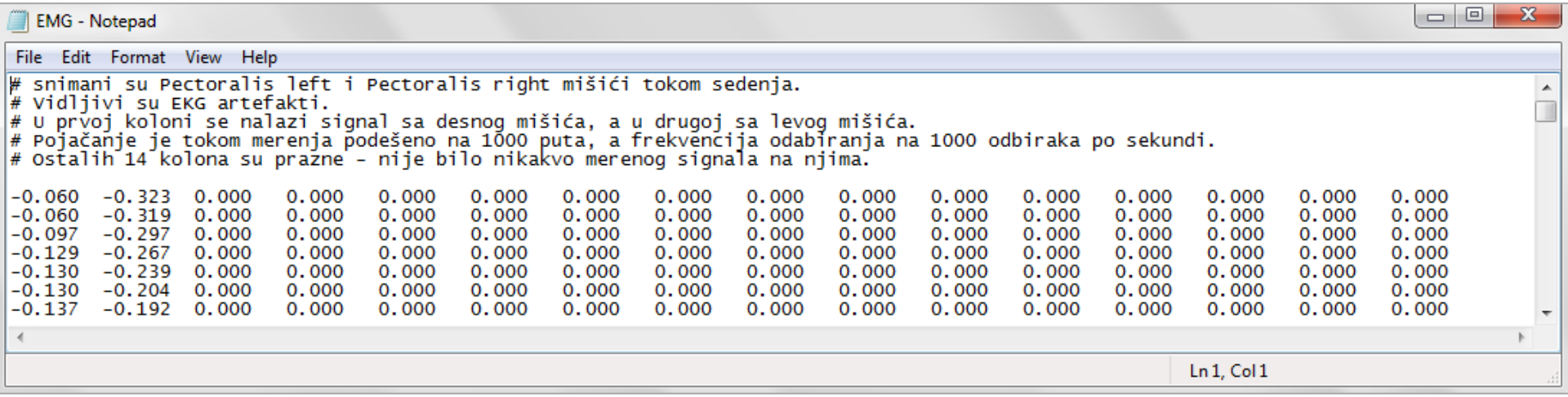
```
read.table(file, header = FALSE, sep = "", quote = "\"'",
           dec = ".", numerals = c("allow.loss", "warn.loss", "no.loss"),
           row.names, col.names, as.is = !stringsAsFactors,
           na.strings = "NA", colClasses = NA, nrows = -1,
           skip = 0, check.names = TRUE, fill = !blank.lines.skip,
           strip.white = FALSE, blank.lines.skip = TRUE,
           comment.char = "#",
           allowEscapes = FALSE, flush = FALSE,
           stringsAsFactors = default.stringsAsFactors(),
           fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)

read.csv(file, header = TRUE, sep = ",", quote = "\"",
         dec = ".", fill = TRUE, comment.char = "", ...)
```

R dokumentacija za ovu funkciju data je na slici.

CASE STUDY

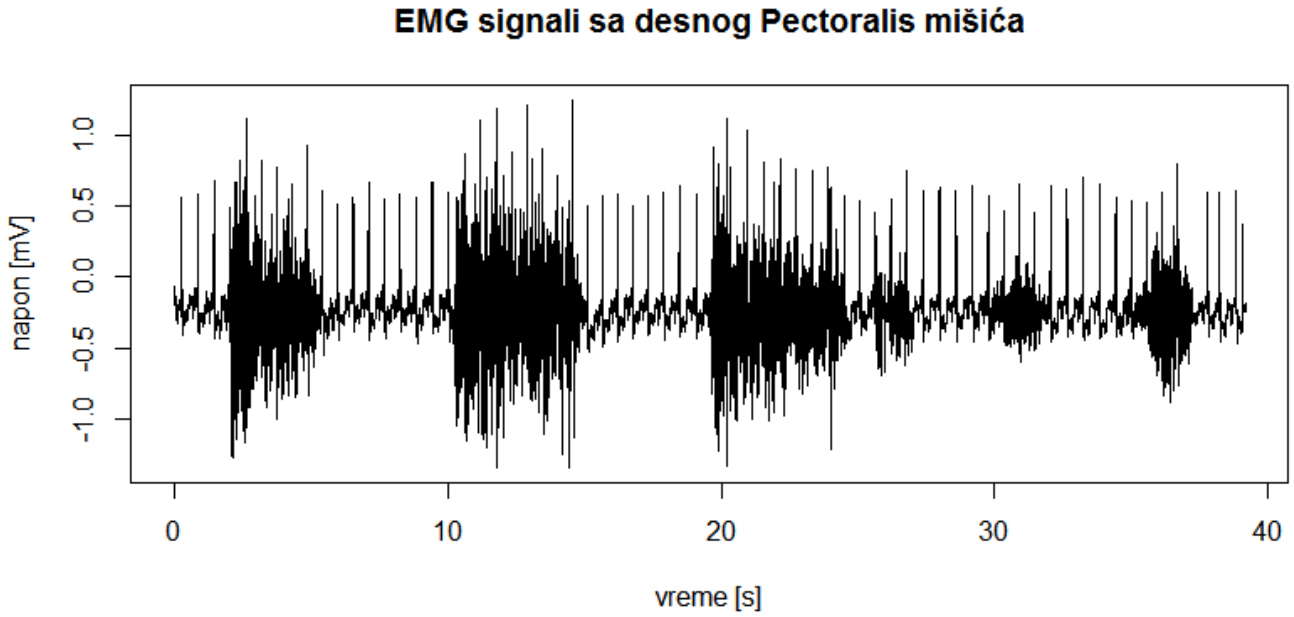
read.table() funkcija

```
> EMGsignali <- read.table("EMG.txt")
> head(EMGsignali)
      V1     V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16
1 -0.060 -0.323  0  0  0  0  0  0  0   0   0   0   0   0   0   0
2 -0.060 -0.319  0  0  0  0  0  0  0   0   0   0   0   0   0   0
3 -0.097 -0.297  0  0  0  0  0  0  0   0   0   0   0   0   0   0
4 -0.129 -0.267  0  0  0  0  0  0  0   0   0   0   0   0   0   0
5 -0.130 -0.239  0  0  0  0  0  0  0   0   0   0   0   0   0   0
6 -0.130 -0.204  0  0  0  0  0  0  0   0   0   0   0   0   0   0
>
```



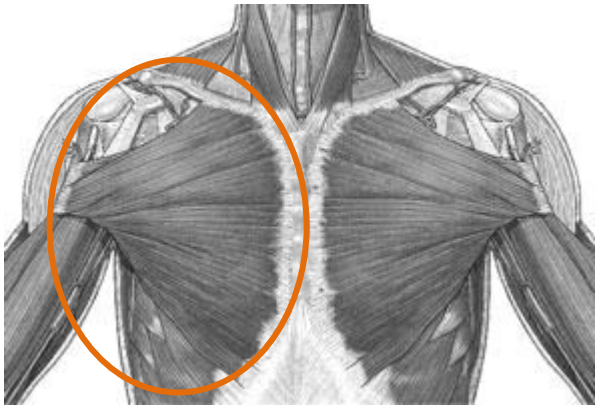
- Na slici su učitani podaci iz tekstualnog fajla “EMG.txt”. Prikaz podataka u *Notepad*-u je prikazan na slici.
- Za prikaz prvih 6 vrsta podatka (na slici *data.frame*-a) koristi se funkcija *head()*.
- Analogno, za poslednjih 6 vrsta koristi se *tail()*.

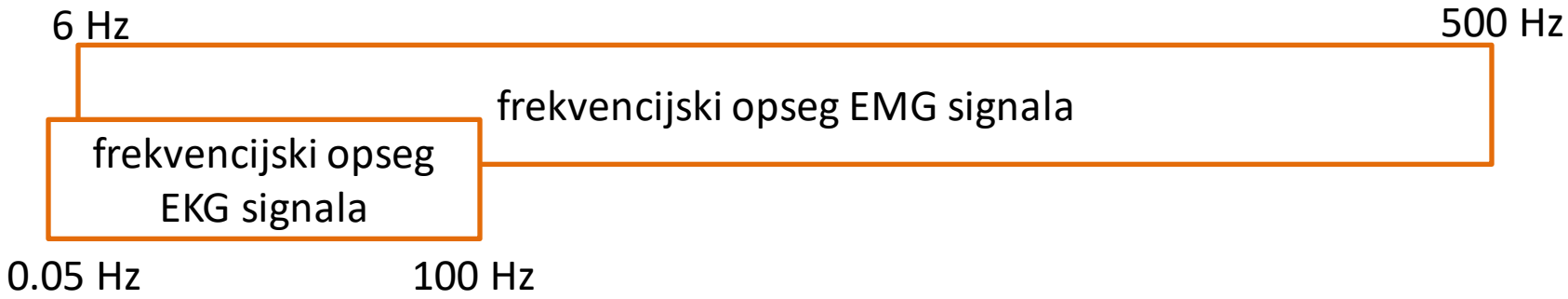
Više o podacima koji se koriste



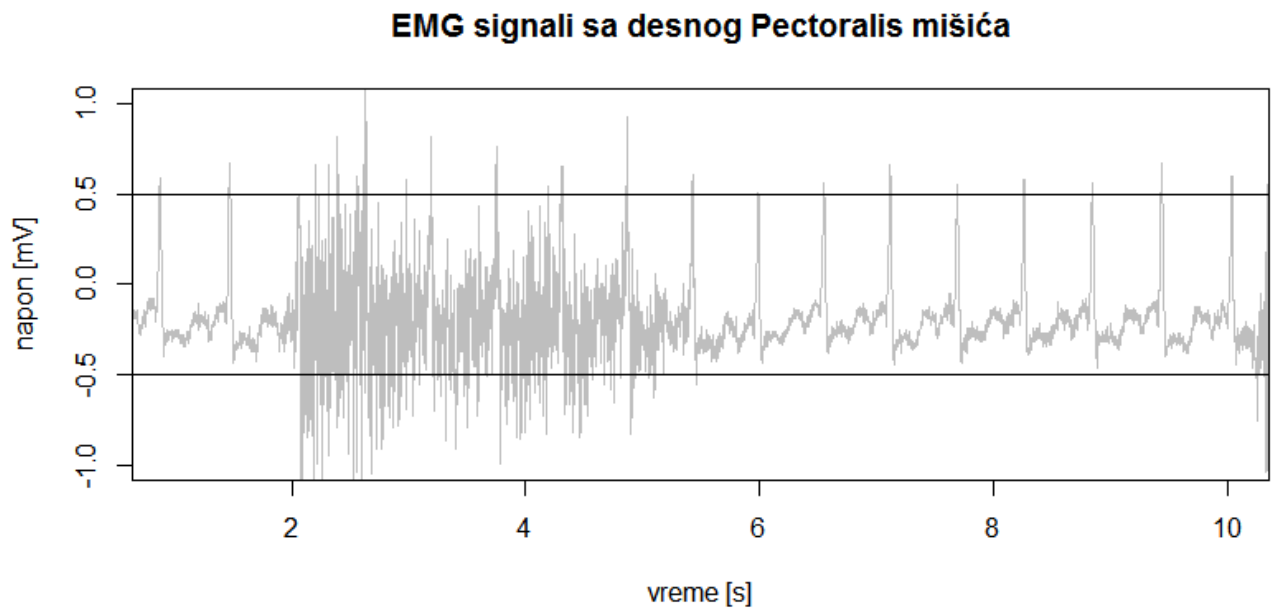
- Elektromiografski (EMG, https://en.wikipedia.org/wiki/Electromyography) signali koji se mere u blizini grudnog koša mogu biti kontaminirani elektrofiziološkim artefaktima:
  - EKG signal – usled blizine srca i
  - signal disanja.
- Mogu se javiti i drugi artefakti. Koji? Na primer artefakt disanja ili šuma napajanja na 50/60 Hz.
- Koliko mišićnih kontrakcija je vidljivo na slici? Tri. Koji artifakt je najuočljiviji? EKG artefakt.
- Podaci korišćeni ovde su mereni na Elektrotehničkom fakultetu u Beogradu i dostupni su na sajtu 13E051TOBS predmeta: http://automatika.etf.rs/images/FAJLOVI_srpski/predmeti/izborni_kursevi_os/biomedicinsko_inzenjerstvo/TOBS/vezbe/EMG.txt.

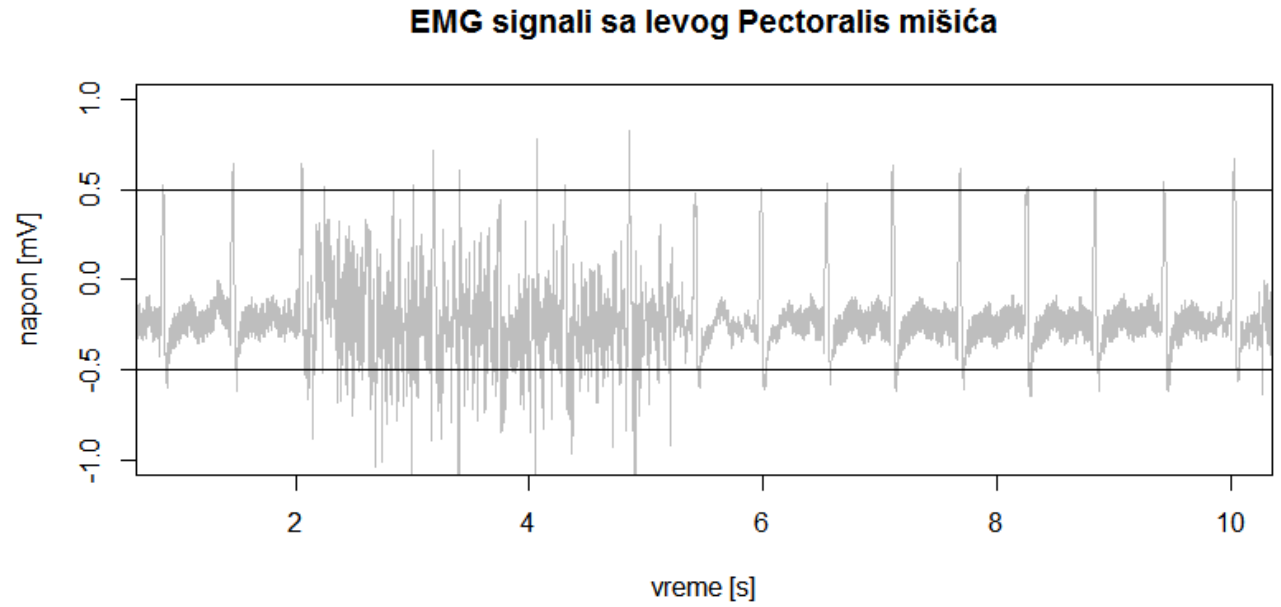
EMG signali i EKG artifakt





- Prikazana je lokacija mišića *Pectoralis major* na kome je meren EMG signal.
- Na donjem panelu su prikazani frekvencijski opsezi šuma i signala.

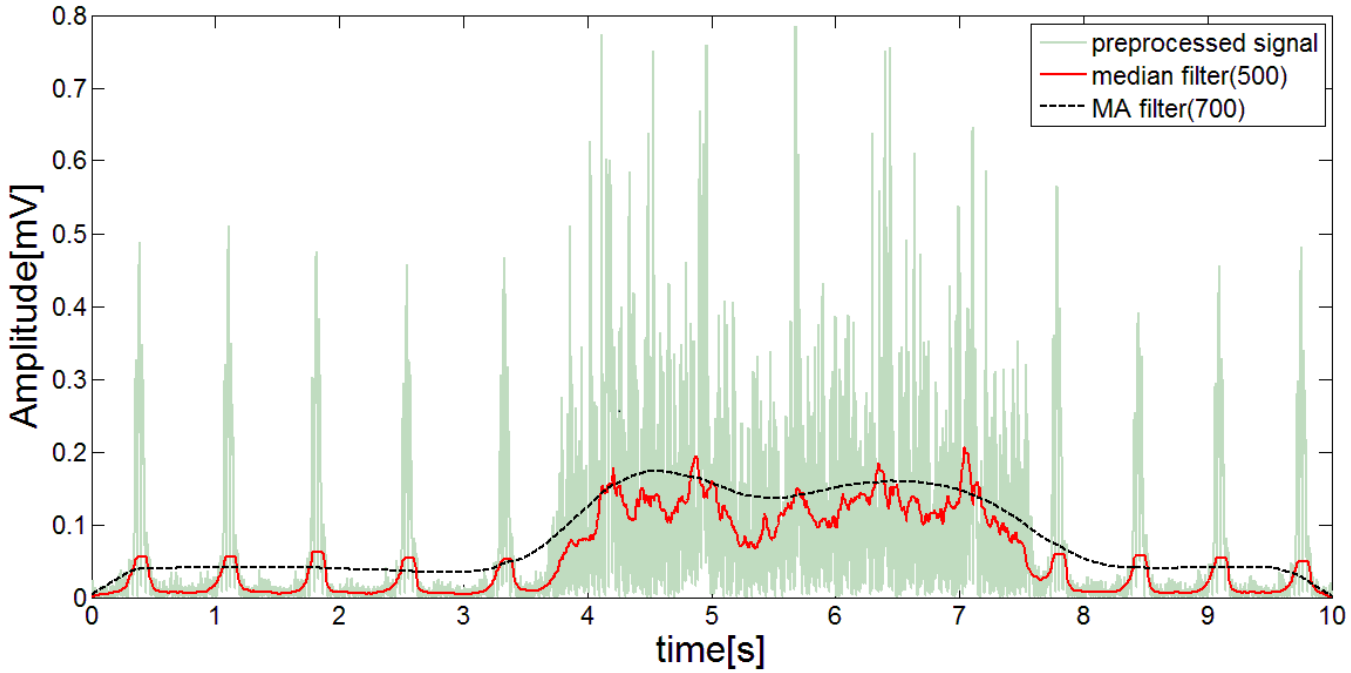




Gde je veći šum?

- Da li je šum veći na EMG signalu koji je meren sa levog ili desnog mišića?
- Zašto?
- Da li bi se ova razlika videla da je grafik drugačije prikazan?

Filtriranje EKG iz EMG



- Ne može se koristiti *high pass* filter. Zašto? Zato što se preklapaju frekvencijski opsezi.
- Jedno od rešenja je filtriranje signala u vremenskom domenu.
- Postoji linearan MA (eng. *Median Average*) i nelinearan *Median* filter.
- Primer učinka ovih filtara je prikazan na slici. Da li su ovi filtri uspešno otklonili šum? Ne nisu, jer su i dalje vidljivi otkucaju srca.
  - Postoji niz drugih metoda koje se koriste *Independent Component Analysis*, premeštanje elektroda, adaptivni filtri i druge.
- Slika je preuzeta i izmenjena iz rada:
  - N. Popović, N. Miljković, O. Djordjević, T. B. Šekara. Artifact cancellation using median filter, moving average filter, and fractional derivatives in biomedical signals, *Proc of the International Conference on Fractional Differentiation and its Applications*, ISBN 978-86-7892-830-7, pp. 150-161, Novi Sad, Serbia, July 18-20, 2016.

ŠTA AKO SU PODACI VEĆEG FORMATA?

Učitavanje relativno većih fajlova

```
> pomoćniSignal <- read.table("EMG.txt", nrows = 20)
> tip <- sapply(pomoćniSignal, class)
> EMGsignalFinalno <- read.table("EMG.txt", colClasses = tip)
> tip
       V1        V2        V3        V4        V5        V6        V7        V8        V9       V10
"numeric" "numeric" "numeric" "numeric" "numeric" "numeric" "numeric" "numeric" "numeric" "numeric"
      V11       V12       V13       V14       V15       V16
"numeric" "numeric" "numeric" "numeric" "numeric" "numeric"
> |
```

- Kako bi se čitanje podataka iz relativno velikih fajlova završilo u konačnom intervalu, potrebno je:
  - proceniti memoriju podataka koji su smešteni u fajl i
  - pročitati *Help* stranicu za funkciju koja se koristi za čitanje podataka.
- Moguće je podesiti i tip podataka, kako bi se ubrzao proces čitanja (argument *colClasses*) -> koristi se pomoćna promenljiva kako bi se smanjila memorija koja se koristi na računaru.
- Postoje i druge opcije, ali prevazilaze deo gradiva predviđen za 19E051TOBS predmet.

Velika količina podataka...

- Potrebno je imati na umu sledeće:
  - Koliko memorije je dostupno na računaru?
  - Da li se koriste druge aplikacije osim R-a?
  - Da li postoji još korisnika koji su ulogovani?
  - Koji operativni sistem se koristi?
  - Da li se koristi 32 bitni ili 64 bitni operativni sistem?
  - …

Modified photo by Nick Fewings on Unsplash

Procena potrebne memorije

- Tzv. *dirty way* procene memorije ...
- Neka podaci imaju 30 000 vrsta i 20 kolona numeričkih podataka.
  - Pretpostavka je da su snimani EMG signali sa 20 mišića u trajanju od 30 sekundi sa frekvencijom odabiranja od 1000 odbiraka u sekundi.
- Otprilike, memorija koja je potrebna za smeštanje ove količine podataka je:
  - 30 000 x 20 x 8 B/numeric =
  - 4800000 B (/ 2^{20} MB) =
  - 4.58 MB -> uporediti ovu vrednost sa RAM-om računara (treba da bude manji od RAM/2 ...)
    - Koliko je to GB?
- Šta će se desiti, ako je ovaj broj veći od RAM-a na računaru?
  - Ako se to desi, ne ostaje puno opcija: *kill R process*, *reboot computer* ...

Dovoljno RAM-a ili ne?

Photo: Random Access Memory (HDR), Computer Part, 3-exposure HDR tone-mapped by Doug Miller; Fickr https://www.flickr.com/photos/sensual_shadows_photography/131643270/; CC-BY 2.0 Generic.

READR PAKET I TEKSTUALNI PODACI

readr paket

- Ovaj paket je razvijen sa ciljem da se unaprede neke opcije postojećih osnovnih funkcija *read.csv()* i *read.table()*.
- Analogne funkcije ovom paketu su *read_csv()* i *read_table()*.
- Ove funkcije su brže od osnovnih, a imaju i *progress meter* ... Ovde ih nećemo koristiti, ali za slučaj da postoji potreba učitavanja "većih" podataka ...
- Na slici je data ilustracija *progress meter* za pretragu fajlova na računaru sa Windows 7 operativnim sistemom.

Rad sa tekstulanim podacima

- Prednosti rada sa tekstualnim podacima:
  - moguće je manuelno menjati podatke,
  - metapodaci su sačuvani (ovo delimično narušava čitljivost podataka u odnosu na npr... ".csv" fajlove),
  - u slučaju potrebe (**nadam se nikada!**) može se pregledati i popraviti deo u kome je došlo do korupcije podataka,
  - *Unix* filozofija?
- Mane rada sa tekstualnim podacima:
  - ovi podacu zauzimaju više memorije od ostalih formata i
  - delimično su "čitljivi" – zbog metapodataka.

Unix filozofija ?

- Svaka oblast inženjerstva i dizajna ima svoju tehničku “kulturu”.
- Ova kultura je uglavnom zasnovana na “nepisanim”, ali (u praksi) primenjivanim pravilima.
- Univerzalni interfejs u tekstualnoj formi, koji se primenjuje u *Unix*-u, predstavlja jedno od nepisanih pravila.
- Takođe, ova pravila uključuju i da tip podataka određuje neku konačnu aplikaciju, a ne algoritam.
- Više o tome u knjizi: The art of Unix programming, Eric Steven Raymond, 2003, http://nakamotoinstitute.org/static/docs/taoup.pdf.

Rad sa drugim formatima podataka?

Modified photo by Rupert Britton on Unsplash

dput() i *dget()* funkcije

```
> setwd("C:/Users/Nadica Miljkovic/Desktop")
> podaci <- data.frame(redniBroj = 1:4, ispitaniciID = "ID")
> podaci
  redniBroj ispitaniciID
1         1           ID
2         2           ID
3         3           ID
4         4           ID
> dput(podaci)
structure(list(redniBroj = 1:4, ispitaniciID = structure(c(1L,
1L, 1L, 1L), class = "factor", .Label = "ID")), .Names = c("redniBroj",
"ispitaniciID"), row.names = c(NA, -4L), class = "data.frame")
> dput(podaci, file = "podaci.R")
> učitaniPodaci <- dget("podaci.R")
> učitaniPodaci
  redniBroj ispitaniciID
1         1           ID
2         2           ID
3         3           ID
4         4           ID
>
```

```
> dump(c("podaci", "učitaniPodaci"), file = "proba.R")
> source("proba.R")
> str(podaci)
'data.frame':   4 obs. of  2 variables:
 $ redniBroj   : int  1 2 3 4
 $ ispitaniciID: Factor w/ 1 level "ID": 1 1 1 1
> str(učitaniPodaci)
'data.frame':   4 obs. of  2 variables:
 $ redniBroj   : int  1 2 3 4
 $ ispitaniciID: Factor w/ 1 level "ID": 1 1 1 1
> ?rm
> ?str
>
```

- U slučaju da se fajlovi sa snimljenim objektima koriste isključivo u R-u, pogodno je koristiti ove funkcije.
- Fajlovi imaju ekstenziju ".R".
- Metapodaci (npr. atributi objekata) su sačuvani.
- Za snimanje većeg broja objekata koriste se *dump()* i *source()* funkcije.

Binarni format podataka

```
> podaci <- data.frame(redniBroj = 1:4, ispitaniciID = "ID")
> string <- "Ovi podaci odgvaraju ispitanicima u studiji br. D023789628"
> save(podaci, string, file = "mojiPodaci.rda")
> load("mojiPodaci.rda")
> save.image(file = "sviPodaci.RData")
> load("sviPodaci.RData")
>
```

- Najčešće se koristi zbog efikasnosti (zauzima relativno mali memorijski prostor u odnosu na ostale formate).
- Može se koristiti i kada ne postoji način da se podaci predstave u tekstualnom formatu (gubi se preciznost numeričkih podataka prilikom konverzije u tekstualni format).
- Funkcije koje se koriste su:
  - *save()* – snimaju se individualni R objekti,
  - *save.image()* – ako se snima veći broj R objekata i
  - *serialize()* – koristi se da bi se individualni R objekat konvertovao u binarni format (može za snimanje u fajl, ali i za prenos preko komunikacione mreže).
- Ovaj slajd je informativan, na predmetu 19E051TOBS se neće raditi sa binarnim podacima!

REZIME

Rezime

Take-home messages

- Postoji 5 osnovnih tj. atomskih podataka u R-u. Postoje i nizovi, liste, matrice, kategoričke promeljive, *data frame*. Postoje i NA i NaN. Ovi podaci mogu imati svoje atribute.
  - NAPOMENA: Postoji i tip podataka kompleksni brojevi.
- Moguće je izvršiti eksplicitnu i implicitnu konverziju podataka.
  - Uvek je poželjnije izvršiti eksplicitnu konverziju.
- Nedostajući podaci se moraju na odgovarajući način identifikovati i tretirati.
- Najčešće se koriste *read.table()* i *read.csv()* funkcije, a moguće je koristiti i readr paket za čitanje tekstualnih podataka.
- EMG signali snimani u blizini grudno koša mogu biti izloženi EKG šumu.
- Za veće količine podataka, potrebno je proceniti memoriju podataka i memoriju računara ili koristiti pakete i funkcije posebne namene.
- Literatura korišćena za programiranje u R-u: Peng, R. D. (2016). *R programming for data science* (pp. 86-181). Leanpub.

Dodatno, o indeksu

- R i Matlab broje od 1 (eng. *default base index*).
- Međutim, R neće javiti grešku ako pozovete niz sa indeksom 0. Neće javiti ni ako je indeks negativan, više o tome kasnije ...
- C i Python broje od 0.
- Veći broj programskih jezika broji od 0.
- Kompletna lista programskih jezika i njihovog podrazumevanog osnovnog indeksa na: https://en.wikipedia.org/wiki/Comparison_of_programming_languages_%28array%29#Array_system_cross-reference_list:

Programming language	Default base index	Specifiable index type[9]	Specifiable base index	Bound check	Multidimensional	Dynamically-sized	Vectorized operations
Ada	index type[10]	yes	yes	checked	yes	init[11]	some, others definable[12]
ALGOL 68	1	no[13]	yes	varies	yes	yes	user definable
APL	1	?	0 or 1[14]	checked	yes	yes	yes
AWK	1	yes, implicitly	no	unchecked	yes, as delimited string	yes, rehashed	no
BASIC	0	?	no	checked	no	init[11]	?
C	0	no	no[15]	unchecked	partially	init,[11][16] heap[17]	no
Ch	0	no	no	checked	yes, also array of array[18]	init,[11][16] heap[17]	yes