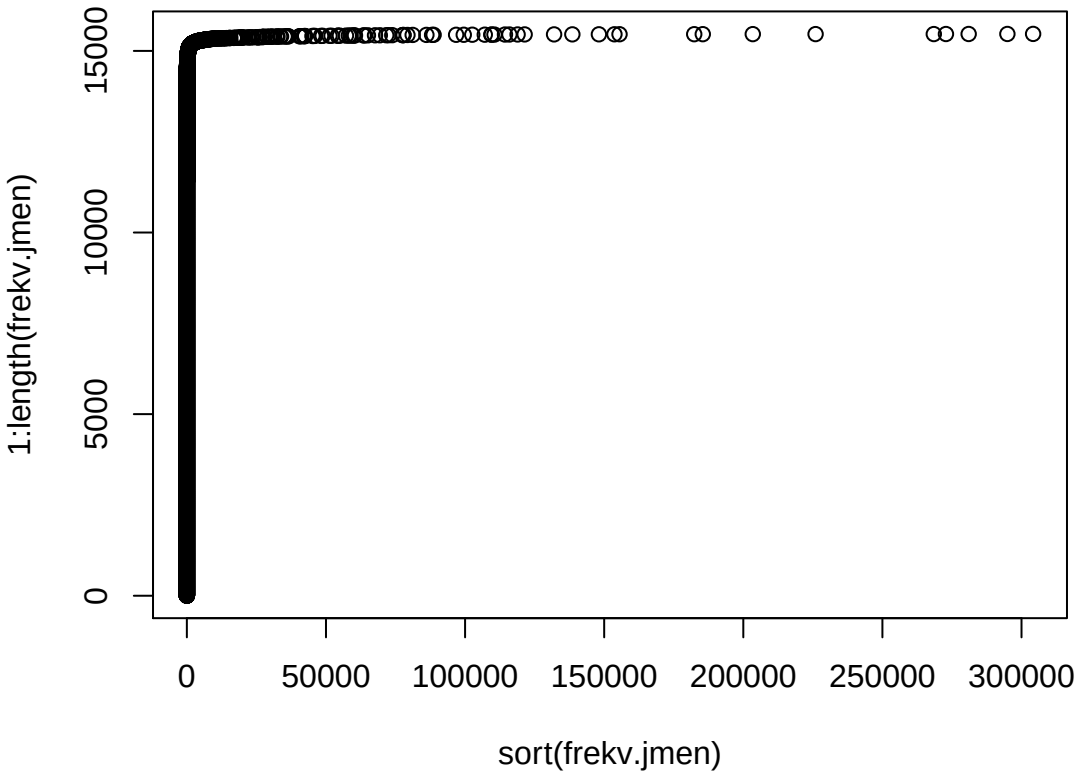
Procvičování 9a

1. Pohrajeme si ještě s těmi jmény, co jsme s nimi pracovali posledně. Tentokrát vám nebudu diktovat, jaké objekty máte v jakém bodě vytvářet a jak je pojmenovat. Spíš po vás budu chtít něco zjistit, případně zobrazit, a jaké objekty si k pomoci vytvoříte nechám na vás. Taky to bude trochu o práci s nápovědou. Pokud dataset s frekvencemi jmen českých občanů *jmena.txt* nemáte importovaný z minula, importujte jej do **R**. Najdete jej ve studijních materiálech, nebo jej můžete načíst přímo z `http://www.sci.muni.cz/~syrovat/jmena.txt`.Tabulka obsahuje v řádcích všechna jednoslovná křestní jména českých obyvatel, ve sloupcích pak jejich frekvenci v jenotlivých ročnících od r. 1896 do 2013.

```
# Problem s encodovanim v LyXu se mi nepodarilo vyresit, proto zase odstranim nejaka
# jmena. Tim padem moje vysledky nebudou sedet s vasimi, ale na script to vliv zadny nema.

# Nastaveni pracovniho adresare:
setwd("D:/My Dropbox/predmety/uvod do R/2014/cv09")
# nacteni jmen bez jmen radku, zabranim vytvoreni faktoru ze jmen a kontrole jmen sloupcu:
jm.data<- read.delim('jmena.txt', sep= '\t', check.names= F, stringsAsFactors= F)
# odstraneni radku s duplikovanymi jmeny:
jm.data<- jm.data[!duplicated(jm.data[, 1]), ]
# definovani jmen radku:
jm.data<- data.frame(jm.data, row.names= 1, check.names= F)

## Error: row names contain missing values

# R hlasi, ze mezi jmeny jsou missing values (NA), odstranim i ty:
jm.data<- jm.data[!is.na(jm.data[, 1]), ]
# definovani jmen radku podruhe:
jm.data<- data.frame(jm.data, row.names= 1, check.names= F)
# z nejakeho duvodu se mi v datasetu objevily NA hodnoty u 278 jmen, odstranim tyto radky
jm.data<- jm.data[!is.na(rowSums(jm.data)), ]
```

2. Celý dataset obsahuje hromadu balastních jmen. Mám na mysli jména, jež jsou velmi vzácná, aby se s nimi dalo něco dělat. Jedná se o jména cizinců, či o špatně zadaná jména. Odstraníme tedy vzácná jména. Nejprve si ale namalujeme dotchart, abychom si udělali obrázek o frekvencích jmen. Dotchart (Cleveland dot plot) je jednou z grafických možností pro posouzení rozložení hodnot. Spočívá ve vynešení vlastních hodnot na osu X podle jejich pořadí (Y). Pro přehlednost můžeme hodnoty seřadit. Můžeme použít funkci `dotchart()`, která ale není moc neflexibilní. Raději si dotchart vytvoříme sami: zobrazte seřazené frekvence jmen na ose X a jejich pořadí na ose Y.

```
# frekvence jmen si mohu neprve spocitat a ulozit do nejakeho objektu:
frekv.jmen<- rowSums(jm.data)
plot(sort(frekv.jmen), 1:length(frekv.jmen))
```



3. Je vidět, že opravdu velká většina z oněch 16 tisíc jmen je velmi vzácná a teprve u posledního 1-2 tisíce jmen jejich frekvence narůstá, můžeme si tedy dovolit jich většinu odstranit. Kolik ale? Hranice by měla ležet někde v ohybu té křivky, kterou tvoří body v našem dotchartu. Já jsem zvolil arbitrární kritérium 0,5%. "Odstraňujme vzácná jména postupně od těch nejvzácnějších a zastavme s odstraňováním těsně předtím, než bychom dosáhli 0,5% odstraněných jedinců." Tak docílíme toho, že nám v datasetu zůstane naprostá většina dat (alespoň 99,5%), a zároveň ho vyčistíme od balastu. Vypočítal jsem, že to odpovídá všem jménům s frekvencí menší nebo rovnou 33. To můžete ověřit. Zjistěte, kolik jmen a kolik jedinců nám z datasetu tímto postupem vypadne, kolik tito jedinci tvoří procent přesně, a z datasetu je odstraňte. Od teď budeme pracovat už jen s tímto menším a čistším datasetem.

```
# kolik jmen vypadne (nezapomente, ze muj dataset neni uplny):
sum(frekv.jmen <= 33)

## [1] 13963

# kolik jedincu vypadne:
sum(jm.data[frekv.jmen <= 33, ])

## [1] 47302
```

```
# kolik je to procent:
sum(jm.data[frekv.jmen <= 33, ]) / sum(jm.data) * 100

## [1] 0.524

sum(jm.data)

## [1] 9026312

# odstraneni jmen s freq <= 33:
jm.data33<- jm.data[frekv.jmen > 33, ]
```

4. Už víme, že nejběžnějším jménem je Jiří. Jak je to s dalšími jmény, která začínají na "J"? Zjistěte 10 nejběžnějších jmen začínajících písmenem "J". K tomu budete potřebovat nějakou funkci, která dokáže v textového řetězce vyextrahovat první písmenko. Tu najděte. Jen napovím, že pro řetězec se používá termín "string".

```
# googleni fraze "extract character from string r" mi prvni odkaz odkazuje
# na napovedu fce substr()

# vytazeni prvniho pismenka ze jmen:
jm1<- substr(rownames(jm.data), 1, 1)

# vytazeni jmen zacinajicich na J:
jm.dataJ<- jm.data[jm1 == "J", ]

# serazeni a vypsani 10 nejbeznejsich Jmen:
rownames(jm.dataJ)[order(-rowSums(jm.dataJ))[1:10]]

##  [1] "JIXM"      "JAN"       "JANA"      "JOSEF"     "JAROSLAV"
##  [6] "JAKUB"     "JAROSLAVA" "JITKA"     "JARMILA"   "JIXINA"

# (JIXM = JIRI)
```

5. Teď, když si umíme vytáhnout jména stejného začátečního písmene, bychom se mohli podívat, jak si stojí naše jména s frekvencí v množině jmen stejného začátečního písmene: Zjistěte pořadí Vašeho jména mezi jmény začínajícími stejným písmenem. (Pokud Vaše jméno vypadlo v bodu 2, zjistěte pořadí nějakého jiného oblíbeného jména).

```
# nejprv e si vytahneme jmena stejneho zacatecniho pismenem u me to je V:
jm.dataV<- jm.data[jm1 == "V", ]

# kvuli problemum s diakritikou se moje jmeno prelozilo jako VMT.
# hledam tedy rank jmena frekvence jmena VMT
rank(-rowSums(jm.dataV))["VMT"]

## VMT
##   8
```

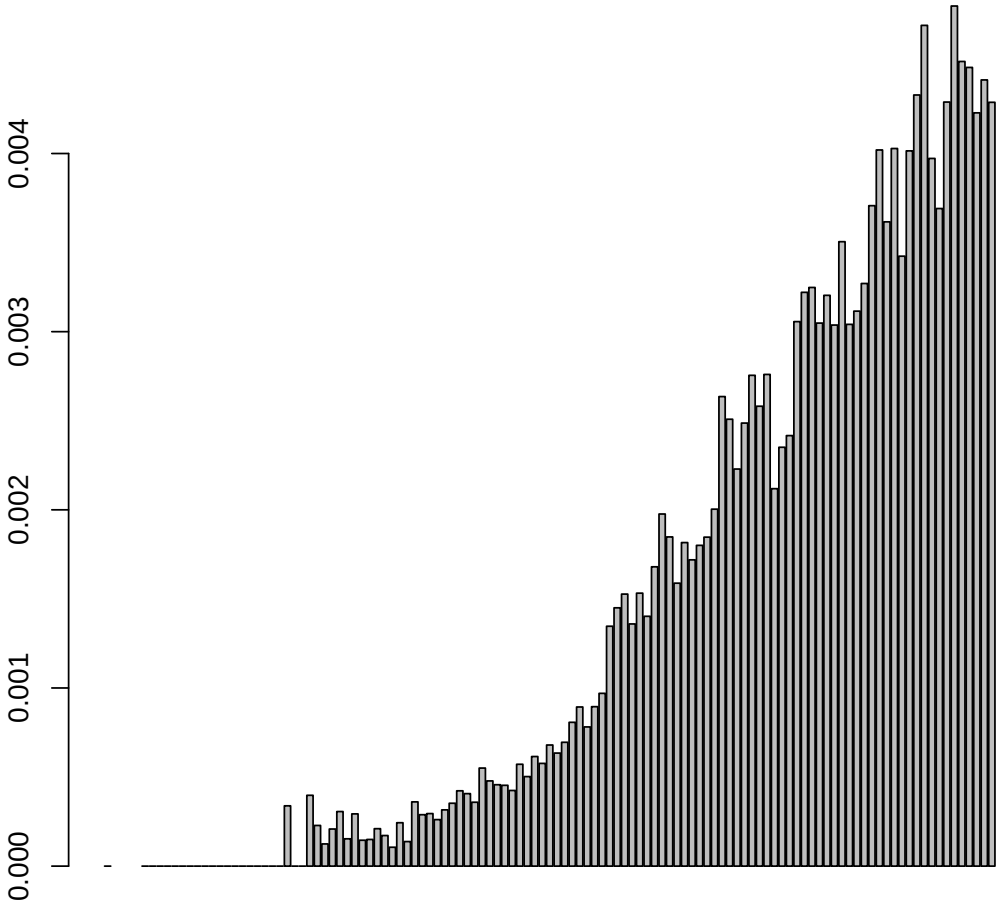
```
# zajimave by bylo i podivat se na prvnich nekolik, rekneme 10, nejbeznejsich jmen
# zacinajicich danym pismenem, nechat si vypsat jejich frekvence a poradi:
jm.V.info<- data.frame(freq= rowSums(jm.dataV),
                       poradi= rank(-rowSums(jm.dataV)))
jm.V.info[order(-jm.V.info$freq)[1:10], ]

##             freq poradi
## VLRA      114330      1
## VLADIMMR   88149      2
## VERONIKA   86094      3
## VOJTLCH    48810      4
## VLASTA     46014      5
## VLASTIMIL  27291      6
## VLADIMMRA  18259      7
## VMT        17447      8
## VENDULA    14232      9
## VLADISLAV  13617     10

#
```

6. Protože data jsou strukturována podle roku narození, můžeme zjistit něco o trendech v používání jmen během posledních desetiletí. Namalujte sloupcový graf relativního zastoupení Vašeho jména v populaci v jednotlivých letech. Výsledkem má být graf se sloupečky, kde počet sloupečků odpovídá počtu let, pro které jsou frekvence jmen v datasetu zaznamenané, a výška sloupců relativnímu zastoupení daného jména mezi jedinci narozenými v patřičném roce. Namalujte stejný graf pro tři různá jména, u kterých byste čekali odlišný vývoj. Sloupečky ve sloupečkovém grafu se označují termínem "bar".

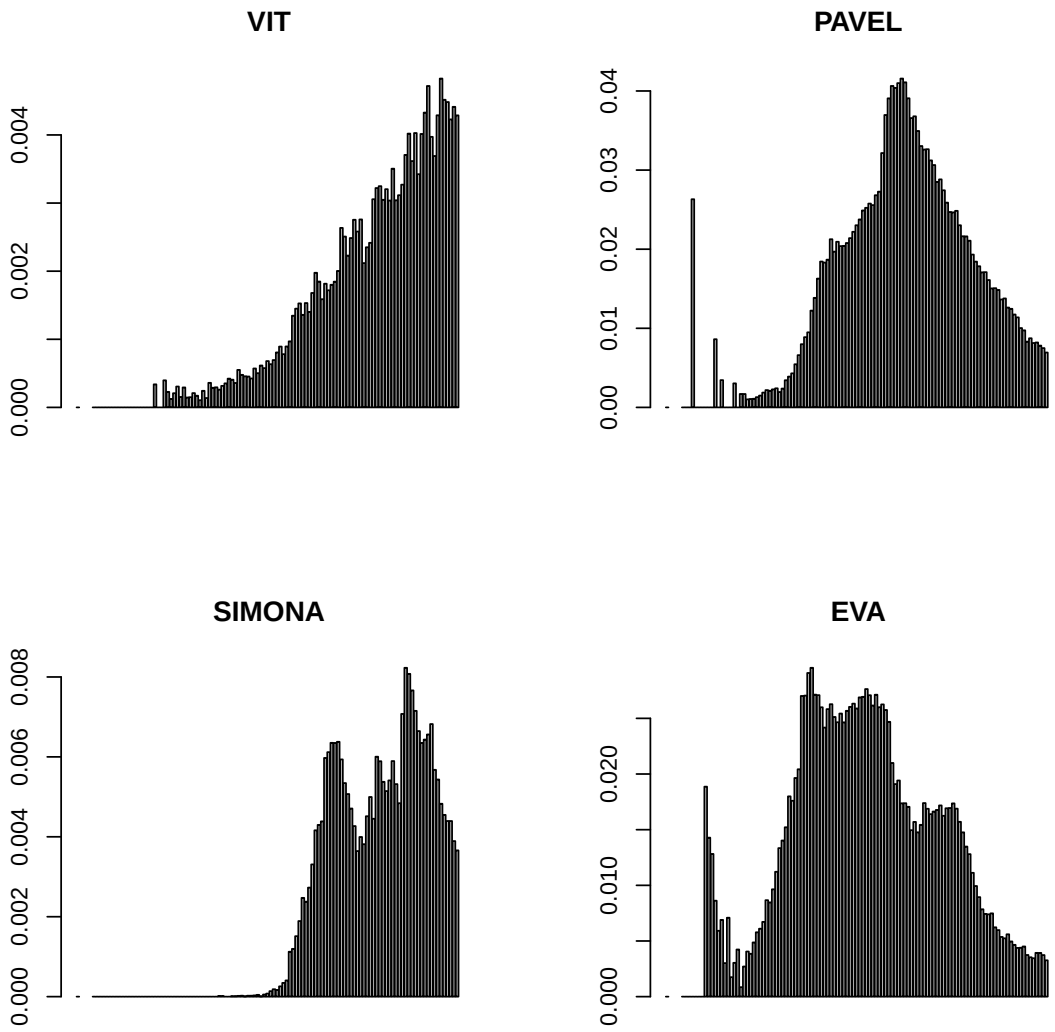
```
# nejprve musime najit funkci, ktera maluje sloupeckove grafy. hledame neco jako "bar graph",
# nebo muzeme zkusit stesti a v RStudiu napiseme 'bar' a stiskneme klavesu Tabulator pro vyhledani
# funkci, ktere zacinaji na 'bar'. nakonec najdeme funkci barplot()
# relativni zastoupeni jmena v populaci daneho rocniku je podil jedincu toho rocniku, kteri nesou
# ku poctu vsech jedincu daneho rocniku.
# v mem pripade tedy:
# jm.data["VMT", ] / colSums(jm.data)
barplot(jm.data["VMT", ] / colSums(jm.data))

## Error: 'height' must be a vector or a matrix

# ha, R hlasi, ze 'height', cili vyska tech sloupecku, musi byt ve formatu vektoru nebo matice.
# co mu to vlastne cpeme? podivame se na strukturu:
str(jm.data["VMT", ] / colSums(jm.data))

## 'data.frame': 1 obs. of  120 variables:
##  $ 0   : num 0
##  $ 1896: num NaN
##  $ 1897: num NaN
##  $ 1898: num NaN
##  $ 1899: num NaN
##  $ 1900: num 0
##  $ 1901: num 0
```

```
##  $ 1902: num 0
##  $ 1903: num 0
##  $ 1904: num 0
##  $ 1905: num 0
##  $ 1906: num 0
##  $ 1907: num 0
##  $ 1908: num 0
##  $ 1909: num 0
##  $ 1910: num 0
##  $ 1911: num 0
##  $ 1912: num 0
##  $ 1913: num 0
##  $ 1914: num 0
##  $ 1915: num 0
##  $ 1916: num 0
##  $ 1917: num 0
##  $ 1918: num 0
##  $ 1919: num 0.000339
##  $ 1920: num 0
##  $ 1921: num 0
##  $ 1922: num 0.000398
##  $ 1923: num 0.000228
##  $ 1924: num 0.000125
##  $ 1925: num 0.000208
##  $ 1926: num 0.000306
##  $ 1927: num 0.000153
##  $ 1928: num 0.000293
##  $ 1929: num 0.000145
##  $ 1930: num 0.00015
##  $ 1931: num 0.00021
##  $ 1932: num 0.000172
##  $ 1933: num 0.000106
##  $ 1934: num 0.000244
##  $ 1935: num 0.000138
##  $ 1936: num 0.000361
##  $ 1937: num 0.000289
##  $ 1938: num 0.000295
##  $ 1939: num 0.000261
##  $ 1940: num 0.000316
##  $ 1941: num 0.000353
##  $ 1942: num 0.000423
##  $ 1943: num 0.000407
##  $ 1944: num 0.000359
##  $ 1945: num 0.000551
##  $ 1946: num 0.000478
##  $ 1947: num 0.000457
##  $ 1948: num 0.000454
##  $ 1949: num 0.000425
##  $ 1950: num 0.000572
##  $ 1951: num 0.000502
##  $ 1952: num 0.000615
```

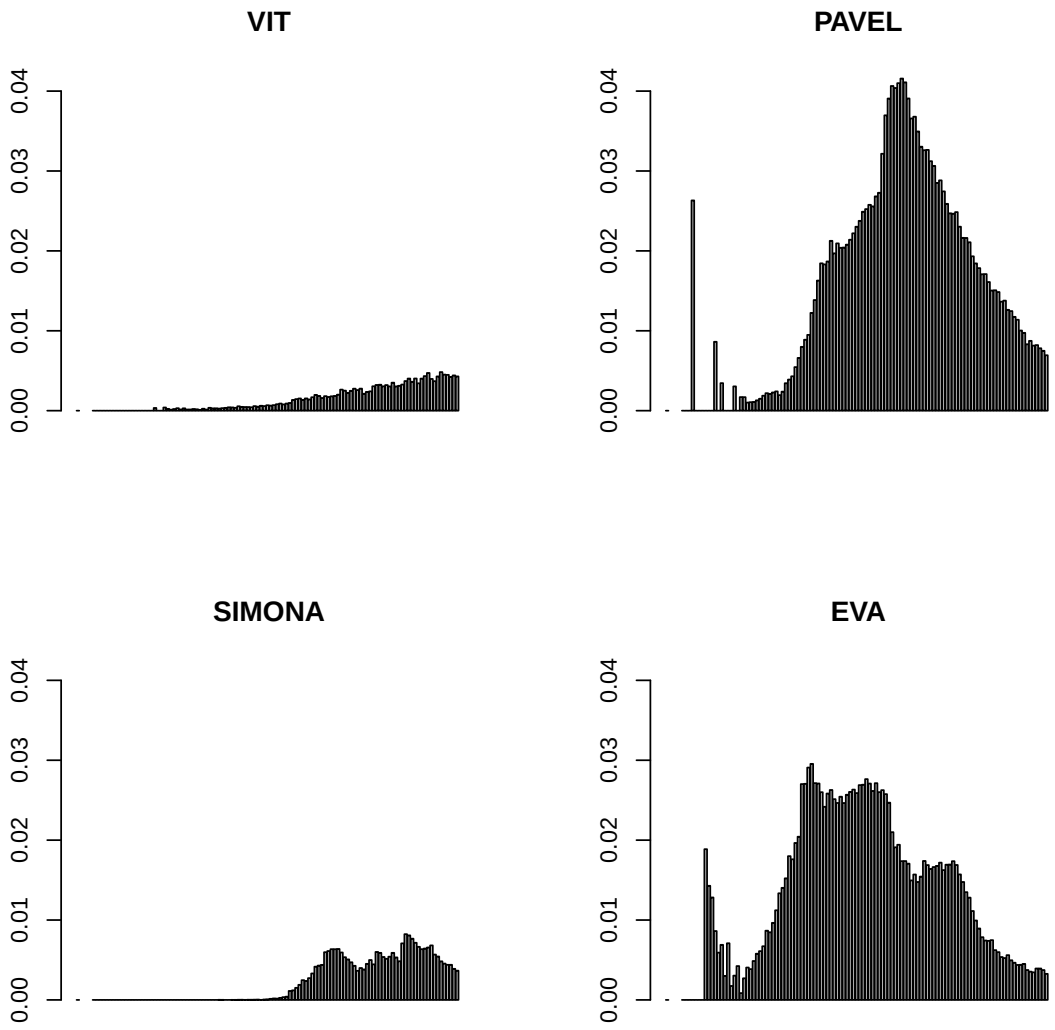
```
##  $ 1953: num 0.000577
##  $ 1954: num 0.00068
##  $ 1955: num 0.000635
##  $ 1956: num 0.000695
##  $ 1957: num 0.000807
##  $ 1958: num 0.000893
##  $ 1959: num 0.000781
##  $ 1960: num 0.000895
##  $ 1961: num 0.00097
##  $ 1962: num 0.00135
##  $ 1963: num 0.00145
##  $ 1964: num 0.00153
##  $ 1965: num 0.00136
##  $ 1966: num 0.00153
##  $ 1967: num 0.0014
##  $ 1968: num 0.00168
##  $ 1969: num 0.00198
##  $ 1970: num 0.00185
##  $ 1971: num 0.00159
##  $ 1972: num 0.00182
##  $ 1973: num 0.00172
##  $ 1974: num 0.0018
##  $ 1975: num 0.00185
##  $ 1976: num 0.002
##  $ 1977: num 0.00264
##  $ 1978: num 0.00251
##  $ 1979: num 0.00223
##  $ 1980: num 0.00249
##  $ 1981: num 0.00275
##  $ 1982: num 0.00258
##  $ 1983: num 0.00276
##  $ 1984: num 0.00212
##  $ 1985: num 0.00235
##  $ 1986: num 0.00242
##  $ 1987: num 0.00306
##  $ 1988: num 0.00322
##  $ 1989: num 0.00325
##  $ 1990: num 0.00305
##  $ 1991: num 0.0032
##  $ 1992: num 0.00304
##  $ 1993: num 0.0035
##   [list output truncated]

# je to tedy dataframe o 120 promennych a jednom radku. musime teda ty hodnoty prevest na vektor:
barplot(as.numeric(jm.data["VMT", ] / colSums(jm.data)))
```



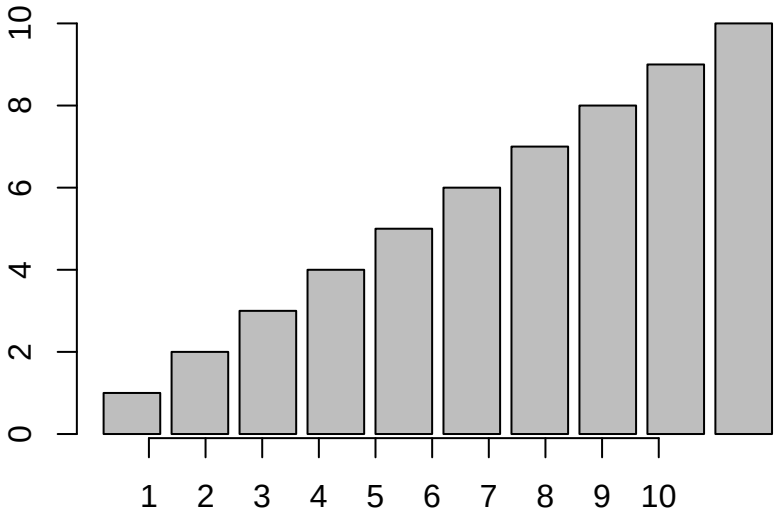

```
# pridam jeste 3 dalsi jmena:
par(mfrow=c(2,2))
barplot(as.numeric(jm.data["VMT", ] / colSums(jm.data)), main= "VIT")
barplot(as.numeric(jm.data["PAVEL", ] / colSums(jm.data)), main= "PAVEL")
barplot(as.numeric(jm.data["SIMONA", ] / colSums(jm.data)), main= "SIMONA")
barplot(as.numeric(jm.data["EVA", ] / colSums(jm.data)), main= "EVA")
```



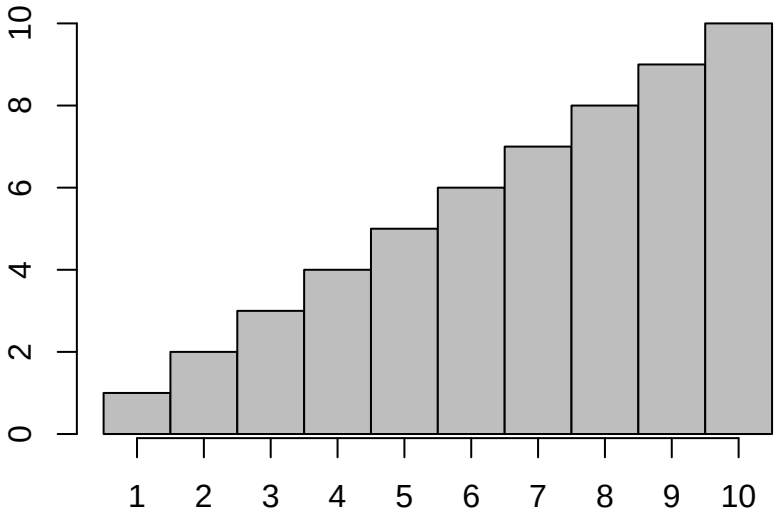

```
# mohli bychom i naskalovat osu Y, aby obrazky byly vzajemne porovnatelne.
# to se dela tak, ze R rekneme rozsah dane osy. minimum je jasne, to bude 0.
# maximum pak bude maximalni hodnota vsech zobrazovanych relativnich zastoupeni.
# nejvyssi sloupecky ma Pavel, tak muzeme pouzit maximum od neho:
Pavel.max<- max(as.numeric(jm.data["PAVEL", ] / colSums(jm.data)), na.rm= T)

par(mfrow=c(2,2))
barplot(as.numeric(jm.data["VMT", ] / colSums(jm.data)), main= "VIT", ylim= c(0,Pavel.max))
barplot(as.numeric(jm.data["PAVEL", ] / colSums(jm.data)), main= "PAVEL", ylim= c(0,Pavel.max))
barplot(as.numeric(jm.data["SIMONA", ] / colSums(jm.data)), main= "SIMONA", ylim= c(0,Pavel.max))
barplot(as.numeric(jm.data["EVA", ] / colSums(jm.data)), main= "EVA", ylim= c(0,Pavel.max))
```



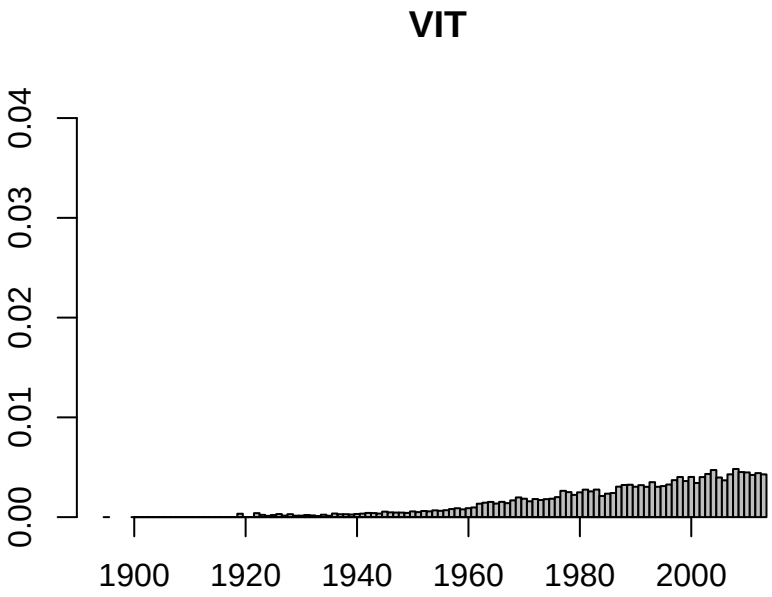

```
# mozna bychom jeste mohli chtit zobrazit osu X...
# to jsou prece jmena sloupcu naseho datasetu.
# tak jen R rict, ktera jmena na ktera mista chceme zobrazit.
# protoze ale R automaticky pridava mezeru mezi sloupecky, nesedely by nam roky
# s prislusnymi sloupecky
# jak vidno na malem prikladu:
barplot(1:10)
axis(1, at= 1:10, labels= 1:10)
```



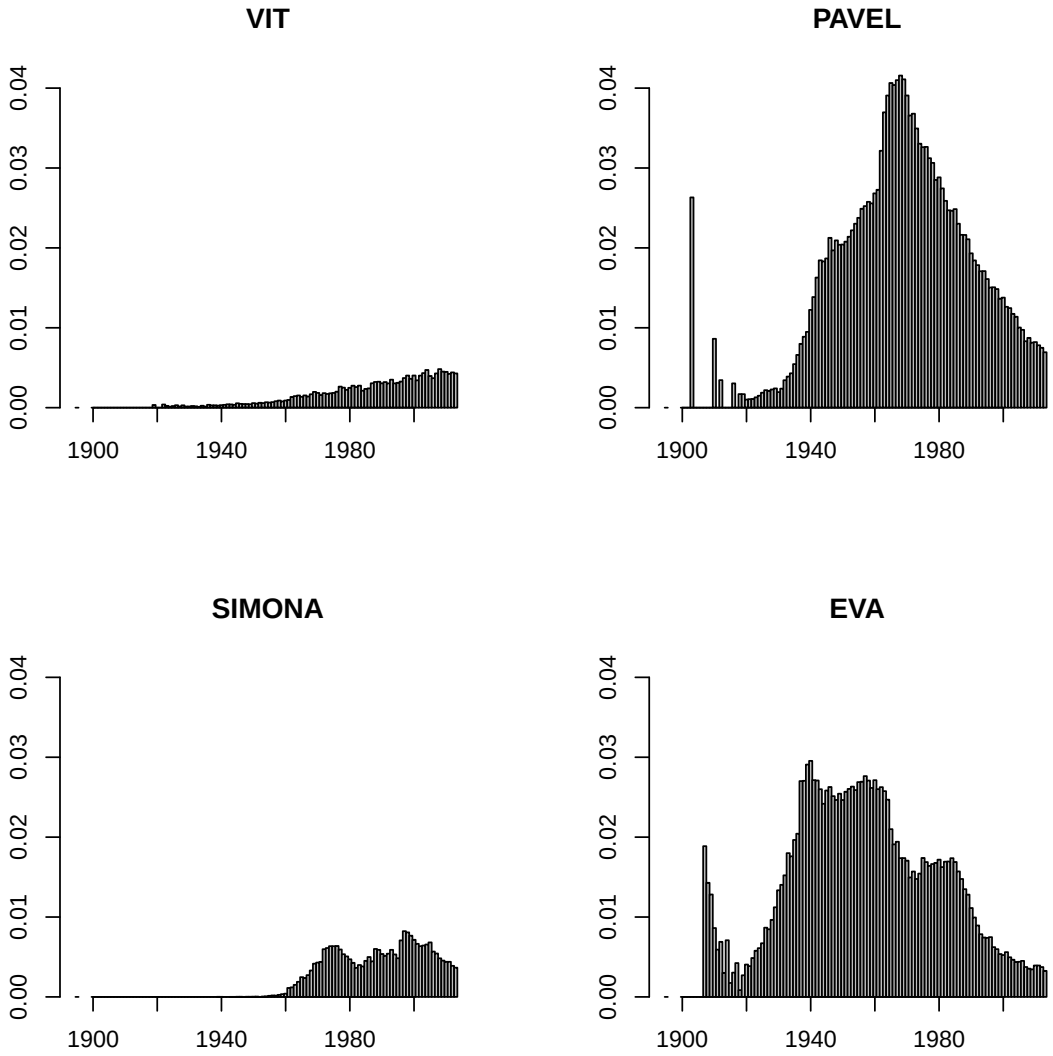

```r
# musime tedy odstranit pridavanou mezeru a jeste odecist 0.5, aby popisek byl zobrazen
# pod stredem sloupce:
barplot(1:10, space= 0)
axis(1, at= 1:10-0.5, labels= 1:10)
```




```r
# urcite se nam na osu X nevejde kazdy rok, kazdych 20 let by mohlo stacit.
poradi.roku<- seq(6, ncol(jm.data), by= 20)
# (sesty rok je 1900)

barplot(as.numeric(jm.data["VMT", ] / colSums(jm.data)), main= "VIT", ylim= c(0,Pavel.max),
  space= 0)
axis(1, at= poradi.roku-0.5, labels= names(jm.data)[poradi.roku])
```




```
# vsimnete si, ze R automaticky vynecha popisky, ktere by se prekryvaly:
par(mfrow=c(2,2))
barplot(as.numeric(jm.data["VMT", ] / colSums(jm.data)), main= "VIT", ylim= c(0,Pavel.max),
  space= 0)
axis(1, at= poradi.roku-0.5, labels= names(jm.data)[poradi.roku])
barplot(as.numeric(jm.data["PAVEL", ] / colSums(jm.data)), main= "PAVEL", ylim= c(0,Pavel.max),
  space= 0)
axis(1, at= poradi.roku-0.5, labels= names(jm.data)[poradi.roku])
barplot(as.numeric(jm.data["SIMONA", ] / colSums(jm.data)), main= "SIMONA", ylim= c(0,Pavel.max),
  space= 0)
axis(1, at= poradi.roku-0.5, labels= names(jm.data)[poradi.roku])
barplot(as.numeric(jm.data["EVA", ] / colSums(jm.data)), main= "EVA", ylim= c(0,Pavel.max),
  space= 0)
axis(1, at= poradi.roku-0.5, labels= names(jm.data)[poradi.roku])
```




```
# no a nakonec jen dodatek, ze na namalovani tech 4 grafu jsem vlastne pouzil
# jednu sady prikazu (namaluj graf, pridej osu) ctyrikrat, jen na jinou cast datasetu.
# kdo jste byli na posledni hodine, jiz patrne vidite, ze by se to dalo zavrit do smycky,
# ve ktere bychom R rekli, aby nam ten stejny obrazek namalovalo pro dana 4 jmena.
# to necham uz na vas.
```