

# Osnovna statistična analiza v R-ju

Aleš Žiberna



# Osnovna statistična analiza v R-ju

Aleš Žiberna

12. julij 2016

CIP - Kataložni zapis o publikaciji  
Narodna in univerzitetna knjižnica, Ljubljana

004.42:311.1(075.8)(0.034.2)

ŽIBERNA, Aleš

Osnovna statistična analiza v R-ju [Elektronski vir] / Aleš Žiberna. - El.  
knjiga. - Ljubljana : Fakulteta za družbene vede, Založba FDV, 2016

ISBN 978-961- 235-779- 5 (pdf)

284426496

*Moji družini.*

*Ženi Tamari, za ljubezen in podporo.*

*Hčerkam Hani, Živi in Zoji, ker so svojo otroško igrivostjo  
moji sončki.*



# Predgovor

Pričujoči učbenik je namenjen študentom, ki že poznajo osnove univariatne in bivariatne statistike ter multiple regresije in se želijo naučiti omenjene metode uporabljati v programskem paketu **R** ali pa želijo predvsem spoznati osnove uporabe statističnega programskega paketa **R**. V učbeniku je torej precej natančno predstavljena uporaba **R**-ja, tako za opravljanje splošnih opravil, ki se uporabljajo pri večini analiz (priprava podatkov, risanje grafov), kot tudi za izvedbo univariatnih in bivariatnih statističnih analiz ter multiple regresije, sama statistična teorija pa je obravnavana le toliko, kolikor se mi je zdelo glede na predznanje študentov (več o tem v naslednjem odstavku) nujno potrebno.

V prvi vrsti je učbenik namenjen študentom predmetov Statistika in analiza podatkov na magistrskem programu Družboslovna informatika na Fakulteti za družbene vede (v nadaljevanju FDV) Univerze v Ljubljani (v nadaljevanju UL), Tehnični in informacijski sistemi na dodiplomskem programu Družboslovna informatika na FDV UL in Multivariatna analiza na magistrskem programu Uporabna statistika na UL (medfakultetni program), njegova uporaba pa se načrtuje tudi pri predmetu Osnove statistične obdelave s pomočjo informacijske tehnologije na Fakulteti za socialno delo UL. Tako učbenik predvideva znanje opisne statistike in statističnega sklepanja s področij univariatne in bivariatne statistike ter multivariatne regresije v obsegu, kot se obravnava pri predmetih Statistika in Statistika II z računalniško analizo podatkov na FDV UL. Obravnavane metode zato niso širše predstavljene, razen na mestih, kjer potrebno predznanje presega tisto, kara so študenti obravnavali pri prej omenjenih predmetih (predvsem pri neparametričnih testih).

Učbenik, predvsem njegovo prvo, najobsežnejše poglavje, je namenjen tudi vsem, ki bi radi spoznali **R**, ne glede na to, na katerem področju ga bodo kasneje uporabljali. **R** se namreč lahko uporablja na mnogo področjih, od že omenjene osnovne statistične analize do multivariatne analize, biostatistike, statistične analize finančnih podatkov, podatkovnega rudarjenja, strojnega učenja, računalniške analize besedil, avtomatskega zajema podatkov s spleta ...

**R** (<http://www.r-project.org>) je odprtokodno programsko okolje za statistične analize. **R** ima kar nekaj lastnosti, ki ga naredijo zelo privlačnega:

- je brezplačen,

- podpira večino statističnih metod (preko paketkov),
- se veliko uporablja v raziskovanju (inštituti, univerze ...),
- deluje na večini operacijskih sistemov (Windows, Mac OS/OS X, Linux ...),
- omogoča programiranje,
- je zelo primeren za simulacije in generiranje slučajnih spremenljivk.

Medtem ko večina študentov ceni predvsem prvo izmed naštetih lastnosti (brezplačnost), pa je njegov razcvet v veliki meri povezan s preostalimi.

Predvsem za začetnike pa ima **R** tudi nekaj slabosti. Glavna slabost je, da v osnovni različici nima grafičnega vmesnika (vsaj ne v klasičnem smislu, kjer bi preko menijev in oken zahtevali izvedbo analiz), zato vsaj na začetku učenje poteka počasneje kot pri programih z grafičnim vmesnikom (na primer SPSS, PSPP, Stata ...). Tu je sicer treba omeniti, da obstaja tudi nekaj grafičnih vmesnikov za **R**, med katerimi bi izpostavil predvsem Rcommander (<http://www.rcommander.com/>), saj je najboljše in omogoča izvajanje največjega nabora analiz, še posebej s pomočjo velikega števila paketkov, ki razširjajo njegovo funkcionalnost. Tega vmesnika, kot rečeno, v tem učbeniku ne obravnavam, je pa vsaj za osnovno statistično analizo ta opisan v učbeniku: Kastelec, Damijana in Katarina Košmelj. 2009. *Statistična analiza podatkov s programoma Excel 2003 in R*. Ljubljana: Biotehniška fakulteta. URL [http://www.bf.uni-lj.si/fileadmin/groups/2763/%C5%A1tudijsko\\_gradivo/SAP\\_2003.pdf](http://www.bf.uni-lj.si/fileadmin/groups/2763/%C5%A1tudijsko_gradivo/SAP_2003.pdf).

V učbeniku se omejim na uporabo preko ukazne vrstice. Glavni razlog za to je, da lahko le s tako uporabo izkoristimo vse prednosti **R**-ja, taka uporaba pa je tudi nujna, med drugim za analize meri, simulacije, avtomatsko izdelavo poročil ter mnogo drugih opravil.

Učbenik je razdeljen na tri osnovna poglavja. Prvo je Uvod v **R** in je namenjeno spoznavanju **R**-ja. Začne se z uvodnim primerom statističnih analiz, nadaljuje pa se z razmeroma sistematičnim pregledom področij, ki so pomembna za uporabo tega programskega okolja. Področja vsebujejo tako osnovne podatkovne strukture in osnove programiranja kot tudi delo z datotekami, risanje in pripravo poročil na podlagi opravljenih analiz. Pravzaprav to poglavje vsebuje tiste "nestatistične" vsebine, ki so potrebne za osnovno uporabo **R**-ja. Poznavanje večine izmed teh vsebin (z izjemo programiranja v **R**-ju – podpodpoglavji 1.4.2 in 1.4.3) je ključno za učinkovito izvajanje in poročanje o takorekoč vseh statističnih analizah in torej tudi za razumevanje preostalih dveh poglavij. Pri tem se prvo poglavje v veliki meri lahko bere "po potrebi", se pravi kot priročnik, kjer se prebere določeno poglavje, ko se pokaže potreba po posameznih znanjih. Z vidika uporabe **R**-ja je to pravzaprav najzahtevnejše poglavje, zato vsebuje tudi vrsto vaj z rešitvami (rešitve so na koncu poglavja).

Drugo poglavje je vsebinsko najširše, saj v njem obravnavamo kar veliko tako univariatnih kot bivariatnih metod. Pod bivariatne metode sicer spadata tudi enofaktorska analiza variance in bivariatna regresija, vendar pa ju, skupaj z njunima multivariatnima različicama, obravnavamo ločeno v naslednjem poglavju. V tem poglavju obravnavamo tako osnovne opisne statistike kot tudi inferenčno statistiko. Znotraj inferenčne statistike obravnavamo pri srednjih vrednostih tako izračune intervalov zaupanja kot preverjanje domnev, pri bivariatni povezanosti pa le preverjanje domnev. Pri večini metod in statistik predstavljena le uporaba v **R**-ju in podani primeri, izjema so neparametrični testi, kjer so obravnavane metode tudi teoretično predstavljene.

V tretjem poglavju sta predstavljeni analiza variance in linearna regresija. Pri obeh začnemo z najenostavnejšim (bivariatnim) primerom, nato pa dodajamo dodatne elemente (več neodvisnih spremenljivk, interakcije med učinki ...). Pri analizi varianc predstavimo tudi njeno neparameterično različico. Pri obeh metodah je posebna pozornost namenjena tudi preverjanju predpostavk obeh metod.

Vsako od treh glavnih poglavji se zaključijo z viri za poglobljanje in vprašanji za ponavljanje. Na koncu učbenika sledi še sklepno poglavje, v katerem povzamemo, kaj vse je bilo v učbeniku obdelano. Nekaj podpoglavij vsebuje tudi zahtevnejše snov, ki za enostavno uporabo ni potrebna. Naslovi takih podpoglavij so označeni z zvezdico (★).

Velik del učbenika predstavlja tudi prikaz dela v **R**-ju s pomočjo izpisov iz "konzole" oziroma ukazne vrstice. Izpisi iz ukazne vrstice so vizualno ločeni z zaobljenimi okvirji s sivim ozadjem in z uporabo drugačne pisave, in sicer *poševne pisave s fiksno širino črk* (Courier)<sup>1</sup> za ukaze in običajne pisave s fiksno širino črk za izpise. Vsaka vrstica, kjer nastopa ukaz, se začne z znakom `>`. Ukaz se lahko nadaljuje čez več vrstic. V tem primeru se naslednja vrstica začne z dvema presledkoma. Vrstice, ki se ne začnejo z znakom `>` in niso nadaljevanje ukaza iz prejšnje vrstice, vsebujejo izpise **R**-jevih funkcij. V **R**-ju je komentar vse, kar sledi znaku `#`, pojavlja pa se lahko na začetku vrstice ali kasneje. Vsa **R**-jeva koda, uporabljena v tem učbeniku, je na voljo na spletnem naslovu [www2.arnes.si/~aziber4/R/](http://www2.arnes.si/~aziber4/R/).

Primer izpisa iz **R**-ja s komentarjem je videti takole.

```
> 2 + 1 #tole je ukaz za seštevanje dveh števil  
[1] 3
```

V učbeniku uporabljam še dve posebni grafični obliki. Prva je opozorilo.

<sup>1</sup> Enaka pisava se sicer uporablja tudi znotraj besedila za **R**-jevo kodo.

**Opozorilo!**

V opozorilih opozarjam na še posebej pomembne stvari, katerih neupoštevanje lahko vodi do resnih in pogosto nepričakovanih posledic.

Druga oblika so dodatna pojasnila.

Dodatna pojasnila so pojasnila, ki jih lahko večina bralcev brez hujših posledic izpusti, vseeno pa lahko pripomorejo k boljšem razumevanju snovi.

Pri pripravi učbenika je bil uporabljen **R** različice 3.2.2. Različico **R**-ja lahko preverite med drugim z izpisom spremenljivke *R.version.string*.

```
> R.version.string  
[1] "R version 3.2.2 (2015-08-14)"
```

# Kazalo

<b>Predgovor</b>	<b>i</b>
<b>Kazalo</b>	<b>v</b>
<b>Slike</b>	<b>viii</b>
<b>Tabele</b>	<b>x</b>
<b>1 Uvod v R</b>	<b>1</b>
1.1 Uvodni primer . . . . .	1
1.2 Osnovne informacije . . . . .	14
1.2.1 Osnovne računske operacije . . . . .	14
1.2.2 Spremenljivke . . . . .	15
1.2.3 Uporaba funkcij in pomoči . . . . .	16
1.2.4 Paketki . . . . .	18
1.2.5 Drugi osnovni podatki . . . . .	19
1.3 Podatkovne strukture . . . . .	20
1.3.1 Osnovni podatkovni tipi . . . . .	20
1.3.2 Vektor . . . . .	22
1.3.3 Nominalne in ordinalne spremenljivke . . . . .	26
1.3.4 Seznam . . . . .	27
1.3.5 Matrika . . . . .	30
1.3.6 Polje – Array . . . . .	35
1.3.7 Podatkovni okvir – Data frame . . . . .	35
1.3.8 Vaje . . . . .	38
1.4 Funkcije in programiranje . . . . .	38
1.4.1 Nekaj koristnih funkcij . . . . .	39
1.4.2 Definiranje funkcij ★ . . . . .	43
1.4.3 Programski tok ★ . . . . .	44
1.4.4 Vaje . . . . .	47
1.5 Delo z datotekami . . . . .	47
1.5.1 Tekstovne datoteke . . . . .	48
1.5.2 Shranjevanje in branje objektov . . . . .	50
1.5.3 Branje in pisanje datotek drugih programov . . . . .	51

1.5.4	Vaje . . . . .	55
1.6	Risanje . . . . .	56
1.6.1	Visokonivojske funkcije za risanje . . . . .	56
1.6.2	Nizkonivojske funkcije za risanje . . . . .	58
1.6.3	Shranjevanje slik . . . . .	60
1.6.4	Vaje . . . . .	62
1.7	Priprava dokumentov z rezultati iz <b>R</b> -ja . . . . .	63
1.8	Rešitve vaj . . . . .	66
1.8.1	Podatkovne strukture . . . . .	66
1.8.2	Funkcije in programiranje . . . . .	67
1.8.3	Risanje . . . . .	70
1.9	Viri za poglobljanje znanja . . . . .	71
1.9.1	Spletni viri . . . . .	71
1.9.2	Knjižni viri . . . . .	73
1.10	Vprašanja za ponavljanje . . . . .	75
<b>2</b>	<b>Univariatna in bivariatna statistika</b>	<b>77</b>
2.1	Uporabljeni podatki . . . . .	77
2.2	Osnovne statistike . . . . .	78
2.3	Preverjanje domnev o srednjih vrednostih in pripadajoči intervali zaupanja . . . . .	86
2.3.1	Preverjanje domneve o srednji vrednosti in pripadajoči interval zaupanja . . . . .	88
2.3.2	Preverjanje domneve o razliki med srednjima vrednostma na odvisnih vzorcih in pripadajoči interval zaupanja . . . . .	91
2.3.3	Preverjanje domneve o razliki med srednjima vrednostma na neodvisnih vzorcih in pripadajoči interval zaupanja . . . . .	95
2.4	Preverjanje domnev o deležih in pripadajoči intervali zaupanja . . . . .	98
2.4.1	Preverjanje domneve o vrednosti deleža in pripadajoči interval zaupanja . . . . .	99
2.4.2	Preverjanje domnev o razliki med deležema na neodvisnih vzorcih in pripadajoči intervali zaupanja . . . . .	101
2.5	Frekvenčne in kontingenčne tabele . . . . .	103
2.5.1	Frekvenčne tabele . . . . .	103
2.5.2	Kontingenčne tabele . . . . .	105
2.5.3	Povezanost spremenljivk . . . . .	109
2.6	Korelacija . . . . .	112
2.7	Viri za poglobljanje znanja . . . . .	118
2.8	Vprašanja za ponavljanje . . . . .	119
<b>3</b>	<b>Analiza variance in linearna regresija</b>	<b>121</b>
3.1	Uporabljeni podatki . . . . .	121
3.2	Analiza variance (ANOVA) . . . . .	122
3.2.1	Enofaktorska analiza variance za neodvisne vzorce . . . . .	123

---

3.2.2	Večfaktorska analiza variance za neodvisne vzorce ★ . . . . .	131
3.2.3	Enofaktorska analiza variance za odvisne vzorce ★ . . . . .	138
3.3	Linearna regresija . . . . .	141
3.3.1	Dodatne spremenljivke . . . . .	141
3.3.2	Bivariatna regresija . . . . .	143
3.3.3	Nelinearna regresija ★ . . . . .	147
3.3.4	Multipla regresija . . . . .	150
3.3.5	Vključevanje nominalnih/ordinalnih spremenljivk . . . . .	155
3.3.6	Interakcije med spremenljivkami . . . . .	160
3.3.7	Preverjanje predpostavk . . . . .	163
3.3.8	V razmislek ★ . . . . .	174
3.3.9	Izračun "na roke" ★ . . . . .	176
3.4	Viri za poglobljanje znanja . . . . .	183
3.5	Vprašanja za ponavljanje . . . . .	184
<b>4</b>	<b>Za konec</b>	<b>187</b>
	<b>Literatura</b>	<b>191</b>

# Slike

1.1	Strukturni stolpci in krog za spol . . . . .	3
1.2	Strukturni stolpci za povezanost dveh nominalnih spremenljivk . . . . .	4
1.3	Histograma za ekstravertiranost in emocionalno stabilnost . . . . .	5
1.4	Razsevni grafikon za ekstravertiranost in emocionalno stabilnost . . . . .	8
1.5	Dendrogram – Wardovo razvrščanje s kvadrirano evklidsko razdaljo na podlagi spremenljivk emocionalne stabilnosti . . . . .	10
1.6	Graf povprečij po skupinah, dobljenih z metodo voditeljev . . . . .	12
1.7	Rešitev razvrščanja v prostoru glavnih komponent . . . . .	14
1.8	Štirje obrazi . . . . .	19
1.9	Strukturna stolpca in krog na eni sliki . . . . .	58
1.10	Slika z veliko dodanimi elementi . . . . .	60
1.11	Rešitev vaje 1 (Risanje) . . . . .	70
1.12	Rešitev vaje 2 (Risanje) . . . . .	72
1.13	Rešitev vaje 3 (Risanje) . . . . .	73
2.1	Porazdelitev bruto plače . . . . .	89
2.2	Porazdelitev zaupanja v Državni zbor in Evropski parlament . . . . .	92
2.3	Porazdelitev razlike med zaupanjem v Državni zbor in zaupanjem v Evropski parlament . . . . .	93
2.4	Porazdelitev bruto plače pri moških in ženskah . . . . .	96
2.5	Razsevni grafikon – izobrazba in bruto plača . . . . .	113
2.6	Razsevni grafikoni med vsemi spremenljivkami, ki merijo zaupanje v institucije . . . . .	115
3.1	Porazdelitve uporabljenih spremenljivk . . . . .	125
3.2	Porazdelitev bruto plače po krajih bivanja . . . . .	127
3.3	Porazdelitve uporabljenih spremenljivk . . . . .	132
3.4	Porazdelitev bruto plače po krajih bivanja in spolu . . . . .	134
3.5	Porazdelitve dodatnih spremenljivk . . . . .	143
3.6	Odnos med bruto plačo in številom let šolanja . . . . .	144
3.7	Porazdelitev rezidualov . . . . .	146
3.8	Diagnostični grafikoni za linearno regresijo . . . . .	147
3.9	Exponentna zveza – transformacija . . . . .	149
3.10	Ocenjevanje nelinearne zveze . . . . .	151

---

3.11	Odnos med bruto plačo in tipičnim številom delovnih ur na teden . . .	151
3.12	Reziduali v odvisnosti od vrednosti neodvisnih spremenljivk . . . . .	154
3.13	Porazdelitev rezidualov . . . . .	155
3.14	Diagnostični grafikoni za linearno regresijo . . . . .	156
3.15	Diagnostični grafikoni za linearno regresijo – model z interakcijo . . .	164
3.16	Histogram rezidualov . . . . .	165
3.17	Grafikon za ocenjevanje heteroskedastičnosti . . . . .	165
3.18	Grafikoni delnih ostankov . . . . .	172
3.19	Ceres grafikoni . . . . .	173
3.20	Diagnostični grafikoni za linearno regresijo . . . . .	175
3.21	Odnos med izobrazbo in bruto plačo po spolu . . . . .	177

# Tabele

1.1	Frekvenčna tabela za spol . . . . .	2
1.2	Hierarhično razvrščanje - povprečja po skupinah . . . . .	11
1.3	Metoda voditeljev – povprečja po skupinah . . . . .	12
1.4	Primerjava končnih razvrstitev po enotah . . . . .	13

# 1. poglavje

## Uvod v R

Prvo poglavje je namenjeno spoznavanju **R**-ja. Začne se z uvodnim primerom statističnih analiz, nato pa se nadaljuje z razmeroma sistematičnim pregledom področij, ki so pomembna za uporabo tega programskega okolja. Sistematični pregled se začne z osnovnimi informacijami, kjer velja še posebej izpostaviti navodila za uporabo funkcij in pomoči ter paketkov. Tem sledi pregled podatkovnih struktur, torej osnovnih načinov, na katere lahko v **R**-ju shranjujemo podatke in rezultate (in jih nato nadalje obdelujemo). Nadaljuje se s pregledom nekaterih (s statističnega vidika) najkoristnejših funkcij, za napredne uporabnike pa tudi pravila za pisanje novih funkcij in napotki za programiranje. Zelo pomembno podpoglavje je tudi podpoglavje o delu z datotekami, saj preko le teh "beremo" in shranjujemo podatke ter pogosto tudi rezultate statističnih obdelav. **R** je znan tudi po zelo kvalitetnih grafih in prav risanje ter shranjevanje grafov in drugih slik je opisano v naslednjem podpoglavju. Vsebinsko se poglavje zaključi s pregledom načinov za pripravo dokumentov z rezultati analiz iz **R**-ja, kjer se osredotočimo na izvoz tabel v dokumente, kompatibilne z urejevalniki besedila. Ker je to poglavje z vidika uporabe **R**-ja pravzaprav najzahtevnejše poglavje, vsebuje tudi vrsto vaj z rešitvami (po podpoglavjih), ki so podane na koncu poglavja.

### 1.1 Uvodni primer

Za predstavitev načina analize z **R**-jem začnimo z realnim primerom. Primer prikazuje uporabo metod, predstavljenih v tem učbeniku, poleg tega pa prikazuje tudi uporabo multivariatnih metod z **R**-jem.

Podatki, na katerih temelji primer, so bili zbrani v okviru raziskave *Kakovost merjenja egocentričnih socialnih omrežij* (Ferligoj in drugi 2000). Podroben opis raziskave je na voljo na spletni strani Arhiva družboslovnih podatkov (ADP) (<http://www.adp.fdv.uni-lj.si/opisi/egoomr00/>), s katere je (po predhodni registra-

ciji) mogoče tudi prenesti podatke. Za ta primer sem prenesel podatke v SPSS-ovem formatu (običajnem, ne "portable" (prenosljivem), torej datoteka s končnico ".sav"), saj ga **R** dobro bere. Kot sicer natančneje obravnavamo v podpodpoglavju 1.5.3, lahko **R** bere in zapisuje tudi podatke iz/v mnogih/-o drugih formatov, med katerimi posebej izpostavim Statin format (predvsem zaradi priročnosti pri pisanju). V podpodpoglavju 1.5.3 so opisani tudi razlogi za uporabo SPSS-ovega formata v tem učbeniku in napotki v primeru, da bralec bere podatke iz kakšnega drugega formata.

Pred začetka dela v **R**-ju preberemo funkcije (prej pripravljene, iz datoteke "UcbenikR-funkcije.R") in podatke. Ker želimo analizo izvesti samo na posameznikih, ki so bili osebno anketirani, opravimo še izbor enot.

```
> #naložimo datoteko s posebej pripravljenimi funkcijami
> source("UcbenikR-funkcije.R")
> #naložimo paketek za branje podatkov iz "tujih" zapisov
> library(foreign)
> #preberemo podatke
> ego1<-read.spss("egoomr00_f2.sav",to.data.frame=TRUE,
  use.value.labels = FALSE,use.missings=TRUE)
> #izbor samo tistih, ki so bili osebno anketirani
> ego<-ego1[ego1$MODE==1,]
> #kopiranje atributov (zaradi prejšne izbire enot)
> for(i in names(ego1))attributes(ego[,i])<-attributes(ego1[,i])
```

Naredimo še frekvenčno tabelo (lepo izpisana z  $\LaTeX$ -om je v tabeli 1.1) in grafa (slika 1.1). Najprej frekvenčno tabelo.

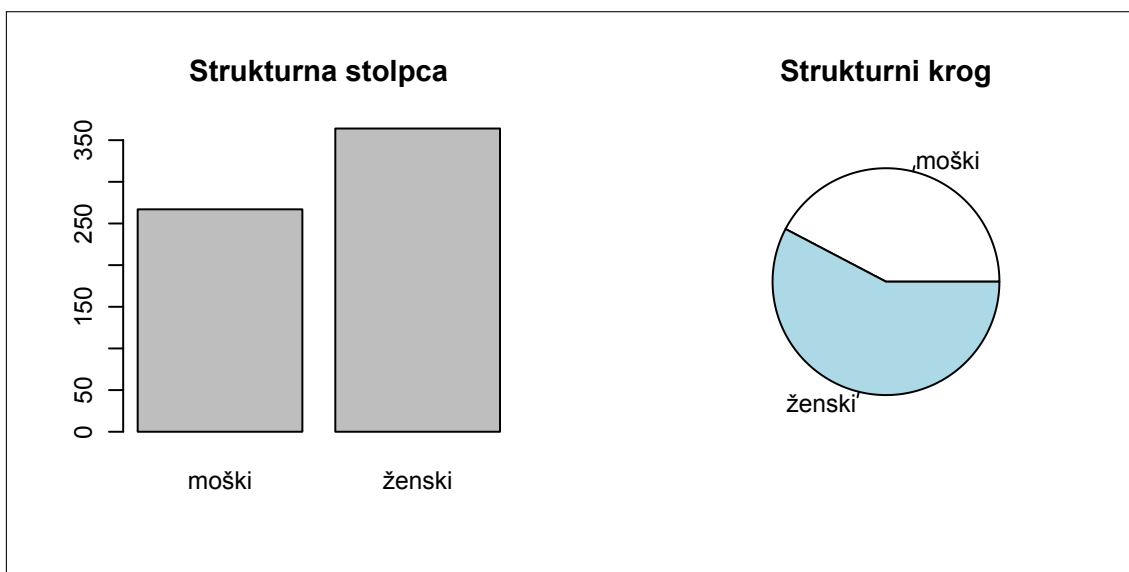
```
> ego$E_SPOL<-makeFactorLabels(ego$E_SPOL)
> table(ego$E_SPOL)
moški ženski
  267   364
> frekTab(ego$E_SPOL,dec=1)
      Frekvenca Kum. frek.    % Kumulativni %
moški      267      267 42.3      42.3
ženski     364      631 57.7     100.0
```

Tabela 1.1: Frekvenčna tabela za spol

	Frekvenca	Kum. frek.	%	Kumulativni %
moški	267	267	42.3	42.3
ženski	364	631	57.7	100.0

Sedaj pa narišimo še strukturalna stolpca in krog.

Slika 1.1: Strukturni stolpci in krog za spol



```
> par(mfrow=c(1,2)) #dva grafa na eno stran
> barplot(table(ego$E_SPOL),main="Strukturalna stolpca")
> pie(table(ego$E_SPOL),main="Strukturalni krog")
> par(mfrow=c(1,1))
```

Delamo lahko tudi dvodimenzionalne tabele in jih na enostaven način narišemo (slika 1.2).

```
> ego$IZOB<-makeFactorLabels(ego$IZOB)
> tbl<-table(spol=ego$E_SPOL,izobrazba=ego$IZOB)
> tbl
```

		izobrazba		
spol		nedokončana osnovna šola	osnovna šola	šola
moški		1	18	
ženski		5	36	

```
> tbl
```

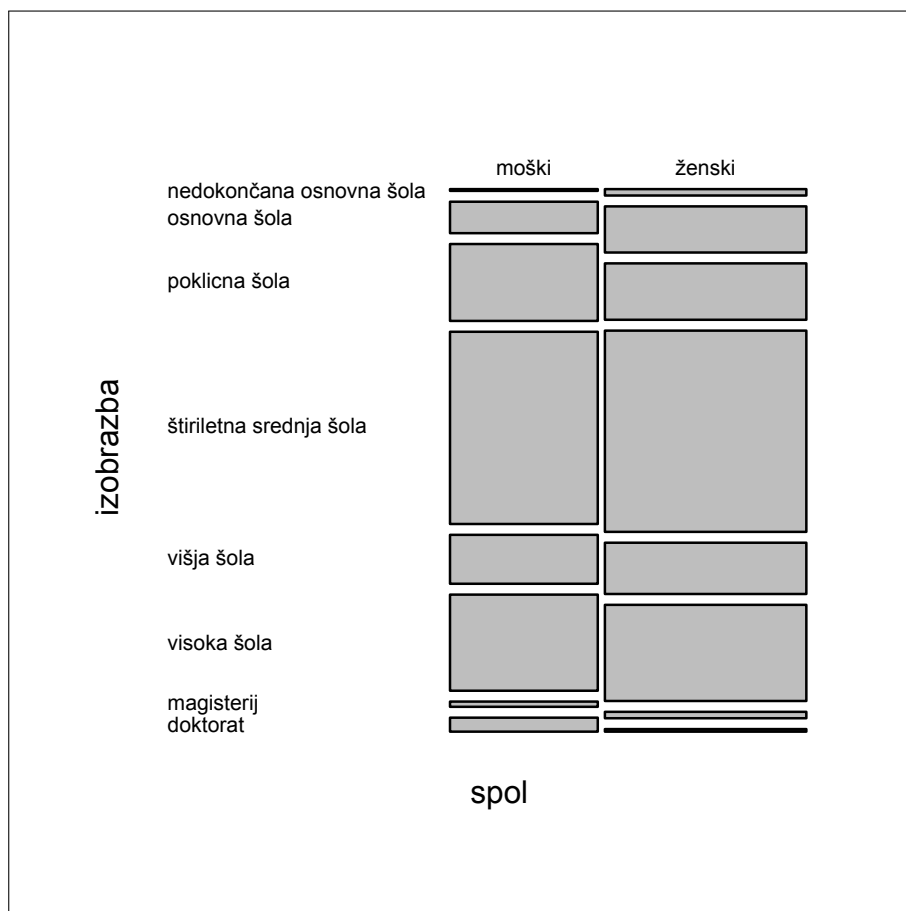
		izobrazba			
spol		poklicna šola	štiriletna srednja šola	višja šola	šola
moški		44		110	28
ženski		44		157	40

```
> tbl
```

		izobrazba		
spol		visoka šola	magisterij	doktorat
moški		55	3	8
ženski		75	5	2

```
> plot(tbl,las=1, main="")
```

Slika 1.2: Strukturni stolpci za povezanost dveh nominalnih spremenljivk



Sledi transformacija spremenljivk – "obračanje" lestvice in izračun Likertovih lestvic za ekstravertiranost in emocionalno stabilnost.

```

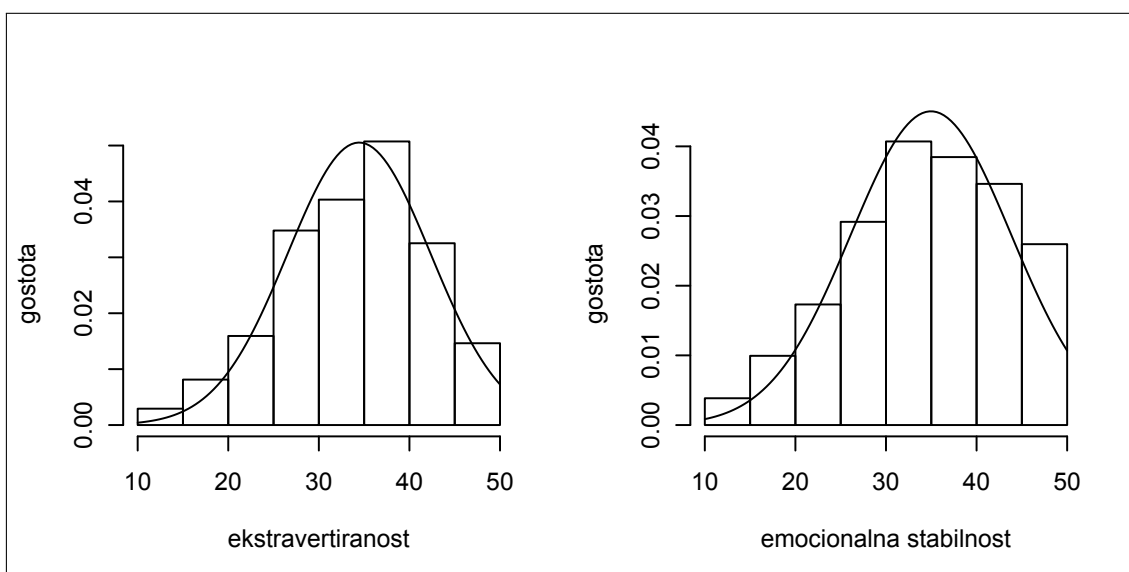
> #seznam spremenljivk, ki jim je treba obrniti lestvico
> spremZaRekod<-c("04", "07", "09", "010", "011", "012",
  "013", "014", "015", "017", "018", "019", "020")
> #imena za nove spremenljivke
> rekodiraneSprem<-paste(spremZaRekod,"R",sep="")
> #"obračanje" vrednosti
> ego[,rekodiraneSprem]<- 6 - ego[,spremZaRekod]
> spremEkst<-c("01", "02", "05", "08", "09R", "012R", "014R",
  "015R", "016", "018R")
> spremEmoc<-c("03", "04R", "06", "07R", "010R", "011R",
  "013R", "017R", "019R", "020R")
> #uvedemo novi spremenljivki
> ego$ekst_sco <- apply(ego[,spremEkst],1,sum)
> ego$emoc_sco <- apply(ego[,spremEmoc],1,sum)

```

Porazdelitev teh dveh spremenljivk je prikazana na sliki 1.3. Porazdelitev obeh spremenljivk je približno normalna, se pa nakazuje rahla asimetrija v levo.

```
> par(mfrow=c(1,2)) #dva grafa na eno stran
> hist(ego$ekst_sco, xlab="ekstravertiranost", main="",
      freq=FALSE,ylab="gostota")
> #dodamo normalno krivuljo
> curve(dnorm(x,mean=mean(ego$ekst_sco,na.rm=TRUE),
             sd=sd(ego$ekst_sco,na.rm=TRUE)),add=TRUE,xpd=NA)
> #raje z gostoto
> hist(ego$emoc_sco, xlab="emocionalna stabilnost", main="",
      freq=FALSE,ylab="gostota")
> #dodamo normalno krivuljo
> curve(dnorm(x,mean=mean(ego$emoc_sco,na.rm=TRUE),
             sd=sd(ego$emoc_sco,na.rm=TRUE)),add=TRUE,xpd=NA)
> #xpd=NA uporabimo, da se vidi tudi vrh krivulje
> par(mfrow=c(1,1))
```

Slika 1.3: Histograma za ekstravertiranost in emocionalno stabilnost



Izračunajmo še opisne statistike po skupinah glede na spol.

```
> #opisne statistike po skupinah
> #ekstravertiranost
> by(ego$ekst_sco, INDICES=ego$E_SPOL,
```

```

FUN = function(x)c(povprecje=mean(x,na.rm=TRUE),
sd=sd(x,na.rm=TRUE),n=sum(!is.na(x)))
ego$E_SPOL: moški
povprecje      sd      n
34.159696  7.711717 263.000000
-----
ego$E_SPOL: ženski
povprecje      sd      n
34.659091  8.031826 352.000000
> #emocionalna stabilnost
> by(ego$emoc_sco, INDICES=ego$E_SPOL,
FUN = function(x)c(povprecje=mean(x,na.rm=TRUE),
sd=sd(x,na.rm=TRUE),n=sum(!is.na(x))))
ego$E_SPOL: moški
povprecje      sd      n
36.306818  8.223621 264.000000
-----
ego$E_SPOL: ženski
povprecje      sd      n
33.986111  9.184723 360.000000

```

Pri ekstravertiranosti (prvi izpis) imajo ženske malce večje povprečje, vendar pa sta si povprečji precej podobni. Pri emocionalni stabilnosti imajo moški večje povprečje. Pri obeh spremenljivkah pa je variabilnost med ženskami večja kot med moškimi. V nadaljevanju za emocionalno stabilnost preverimo domnevo, da sta aritmetični sredini v obeh skupinah (pri obeh spolih) enaki.

```

> ##EMOCIONALNA STABILNOST
> #testiramo enakost varianc (več različnih testov)
> var.test(emoc_sco~E_SPOL,data=ego)
      F test to compare two variances

data:  emoc_sco by E_SPOL
F = 0.80167, num df = 263, denom df = 359, p-value =
0.05669
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.6411573 1.0063491
sample estimates:
ratio of variances
 0.801667
> #če želimo klasičen t-test (predpostavka enakih varianc)
> t.test(emoc_sco~E_SPOL, data=ego,var.equal = TRUE)

```

## Two Sample t-test

```

data:  emoc_sco by E_SPOL
t = 3.2579, df = 622, p-value = 0.001184
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.9218292 3.7195850
sample estimates:
mean in group moški mean in group ženski
 36.30682           33.98611

```

Ker je ena izmed predpostavk klasičnega t-testa tudi enakost varianc po skupinah, smo najprej preverili to domnevo. Domneve pri 5-odstotnem tveganju ne moremo zavrniti, zato izvedemo klasično različico t-testa. Ta nam pove, da lahko pri tveganju približno 0,1 % trdimo, da so moški (v Ljubljani) bolj emocionalno stabilni kot ženske.

Izračunajmo še korelacijo med ekstravertiranostjo in emocionalno stabilnostjo, preverimo domnevo, da je ta korelacija enaka 0, ter prikažimo odnos z njima na razsevnem grafikonu. Ta je prikazan na sliki 1.4, kjer smo z različnimi barvami in znaki označili tudi spol.

```

> plot(jitter(emoc_sco)~jitter(ekst_sco),
      pch=as.numeric(E_SPOL),col=as.numeric(E_SPOL),data=ego,
      ylab="emocionalna stabilnost",xlab="ekstravertiranost")
> #jitter malce razprši točke, da se ne prekrivajo
> legend(x=30,y=52.5,xjust=0.5,yjust=0,legend=c("moški",
      "ženski"),pch=1:2,col=1:2,horiz = TRUE,title="Spol",
      xpd=TRUE)

```

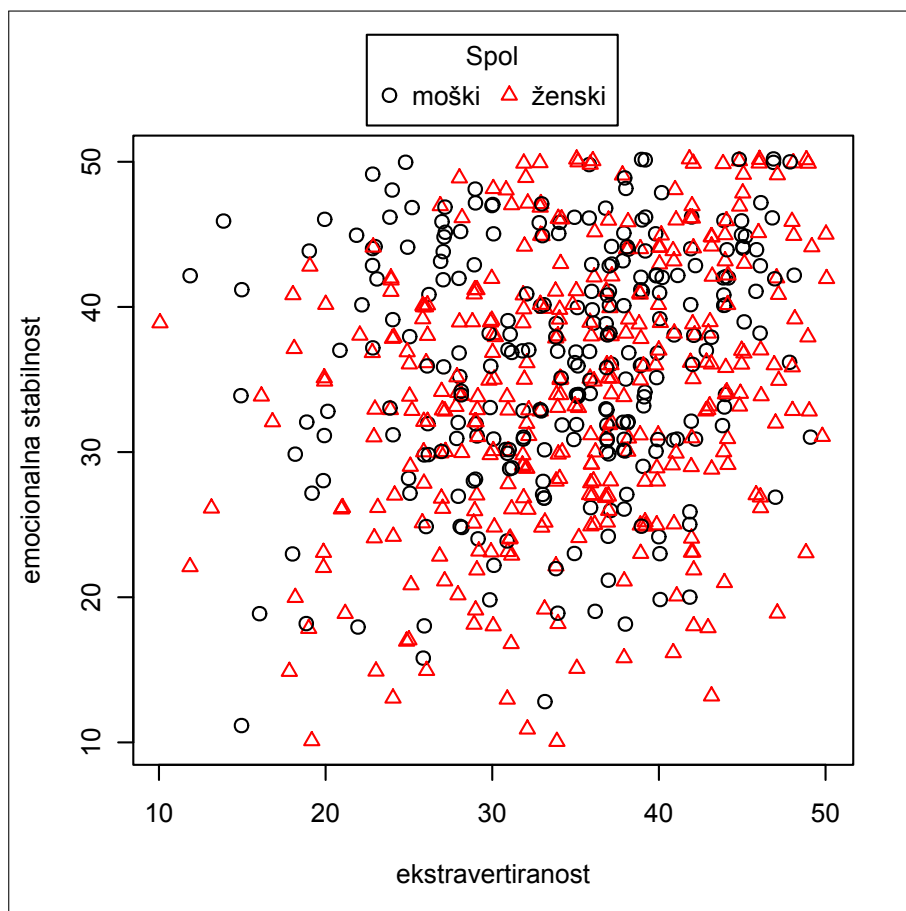
Izračunamo še korelacijo med novima spremenljivkama in preverimo domnevo, da je korelacija enaka 0.

```

> #Pearsonov koeficient korelacije
> cor(ego[,c("emoc_sco","ekst_sco")],method = "pearson",
      use="pairwise.complete.obs")
      emoc_sco ekst_sco
emoc_sco 1.0000000 0.2278617
ekst_sco 0.2278617 1.0000000
> cor.test(y=ego$emoc_sco,x=ego$ekst_sco)
      Pearsons product-moment correlation
data:  ego$ekst_sco and ego$emoc_sco
t = 5.7703, df = 608, p-value = 1.261e-08

```

Slika 1.4: Razsevni grafikon za ekstravertiranost in emocionalno stabilnost



```

alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1512119 0.3017878
sample estimates:
 cor
0.2278617
> #pravzarav bi bil dovolj samo ta drugi ukaz
> #prednost prvega je, da bi lahko izračunali tudi korelacijo
> # med več spremenljivkami

```

Korelacija je sicer razmeroma šibka in pozitivna, a močno statistično značilna ( $p < 0.001$ )<sup>2</sup>. V povprečju imajo osebe z višjo ekstravertiranostjo tudi višjo emocionalno stabilnost in obratno, a ta povezanost je, kot rečeno, šibka.

<sup>2</sup> Tako nizka stopnja tveganja ("močna" statistična značilnost) je predvsem posledica velikega vzorca.

Nadaljujemo z multivariatno analizo – razvrščanjem v skupine. Multivariatne analize sicer v učbeniku ne obravnavam.<sup>3</sup> V uvodni primer sem jo vključil, da bi vsaj nakazal, da **R** omogoča še bistveno več, kot je predstavljeno v tem učbeniku.

Začeli bomo z razvrščanje v skupine. Najprej izločimo enote z manjkajočimi vrednostmi, nato izračunamo kvadrirano evklidsko razdaljo in jo uporabimo za hierarhično razvrščanje z Wardovo metodo. Na podlagi dobljenega dendrograma na sliki 1.5 ocenimo, da v podatkih nastopata 2 skupini.

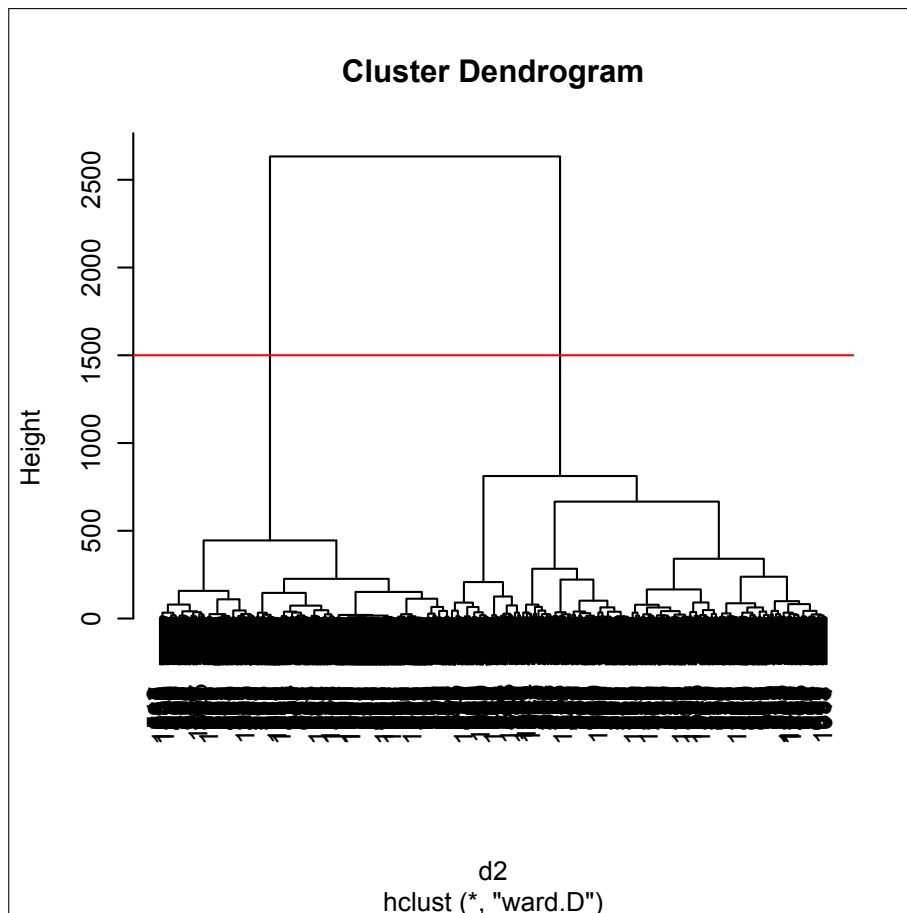
```
> #Hierarhično razvrščanje v skupine
> #odstranimo enote z manjkajočimi vrednostmi
> emoc<-na.omit(ego[,spremEmoc])
> #standardiziramo podatke s funkcijo scale
> Zemoc<-scale(emoc)
> #izračunamo evklidsko razdaljo na standardiziranih podatkih
> d<-dist(Zemoc)
> #za kvadrirano evklidsko razdaljo jo kvadriramo
> d2<-d^2
> #uporabimo Wardovo metodo
> ward<-hclust(d=d2,method="ward.D")
> #narišemo dendrogram
> plot(ward)
> abline(h=1500, col="red")
```

Ti dve skupini shranimo in na nestandardiziranih spremenljivkah izračunamo povprečja po skupinah. Povprečja so prikazana v tabeli 1.2.

```
> #izberemo 2 skupini (in shranimo pripadnost)
> wardCluK2<-cutree(ward,k=2)
> #število enot po skupinah
> table(wardCluK2)
wardCluK2
 1  2
352 272
> #izračunamo povprečja po skupinah
> aggregate(x=emoc, by=list(wardCluK2), FUN=mean)
  Group.1      03      04R      06      07R      010R
1         1 3.022727 2.428977 3.892045 2.650568 2.448864
2         2 4.000000 4.485294 4.621324 4.139706 4.117647
```

<sup>3</sup> Obstaja veliko splošnih učbenikov za multivariatno analizo (na primer Johnson in Wichern 2007; Tabachnick in Fidell 2007; Jesenko in Jesenko 2007), uporaba multivariatne analize v **R**-ju pa je predstavljena predvsem v Everitt (2005); Everitt in Hothorn (2011).

Slika 1.5: Dendrogram – Wardovo razvrščanje s kvadrirano evklidsko razdaljo na podlagi spremenljivk emocionalne stabilnosti



	011R	013R	017R	019R	020R
1	2.306818	3.301136	2.974432	3.536932	2.758523
2	3.746324	3.786765	4.220588	4.639706	4.518382

Dobili smo eno skupino s srednjo in eno z visoko emocionalno stabilnostjo. Poskusimo še, ali lahko z metodo voditeljev dobimo boljši rezultat. Povprečja po skupinah na nestandardiziranih spremenljivkah so prikazana v tabeli 1.3 in na sliki 1.6.

```
> #Razvrščanje v skupine z metodo voditeljev
> kMeansR10<-kmeans(Zemoc, centers=2, iter.max = 50, nstart = 10)
> #ponovimo kar 10-krat (da se izognemo lokalnim minimumom)
> #končni voditelji na standardiziranih spremenljivkah
> #(te smo uporabili za razvrščanje)
> kMeansR10$centers
      03      04R      06      07R      010R
1  0.2807587  0.5873913  0.2360699  0.5311423  0.6126683
```

Tabela 1.2: Hierarhično razvrščanje - povprečja po skupinah

	1	2
O3	3.02	4.00
O4R	2.43	4.49
O6	3.89	4.62
O7R	2.65	4.14
O10R	2.45	4.12
O11R	2.31	3.75
O13R	3.30	3.79
O17R	2.97	4.22
O19R	3.54	4.64
O20R	2.76	4.52

```

2 -0.3449321 -0.7216522 -0.2900287 -0.6525462 -0.7527067
      O11R      O13R      O17R      O19R      O20R
1  0.3711498  0.2967399  0.5099416  0.4663205  0.5518454
2 -0.4559841 -0.3645662 -0.6264997 -0.5729081 -0.6779815
> #velikosti skupin
> kMeansR10$size
[1] 344 280
> #vrednost Wardove kriterijske funkcije
> kMeansR10$tot.withinss
[1] 4585.664
> #povprečja na nestandardiziranih spremenljivkah po skupinah
> aggregate(x=emoc, by=list(kMeansR10$cluster), FUN=mean)
  Group.1      O3      O4R      O6      O7R      O10R
1      1 3.854651 4.261628 4.470930 4.104651 4.098837
2      2 2.950000 2.175000 3.889286 2.310714 2.042857
      O11R      O13R      O17R      O19R      O20R
1 3.526163 3.947674 4.290698 4.613372 4.322674
2 2.207143 2.978571 2.567857 3.285714 2.546429

```

```

> #graf
> means<-aggregate(x=emoc,by=list(kMeansR10$cluster),FUN=mean)
> matplot(t(means[,-1]),type="o",xaxt="n",ylab="povprečje")
> axis(side=1,at=1:dim(means[,-1])[2],
      labels=colnames(means)[-1],las=2)

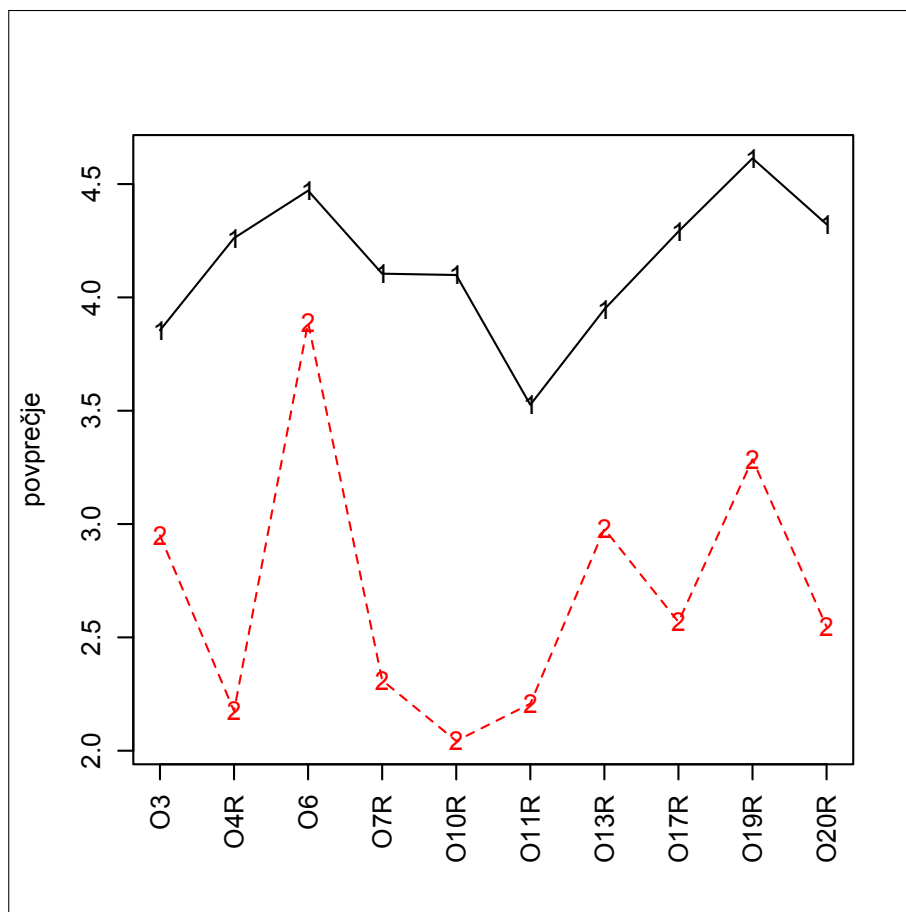
```

Primerjajmo še obe razbitji (dobljeni z različnima metodama). Vrednost kriterijske funkcije za razbitje, izračunano z Wardovo metodo, je 4913.53, za tisto z metodo voditeljev pa 4585.66. Razbitje, dobljeno z metodo voditeljev, je torej boljše. Kako

Tabela 1.3: Metoda voditeljev – povprečja po skupinah

	1	2
O3	3.85	2.95
O4R	4.26	2.17
O6	4.47	3.89
O7R	4.10	2.31
O10R	4.10	2.04
O11R	3.53	2.21
O13R	3.95	2.98
O17R	4.29	2.57
O19R	4.61	3.29
O20R	4.32	2.55

Slika 1.6: Graf povprečij po skupinah, dobljenih z metodo voditeljev



se razporeditvi prekrivata, je prikazano v tabeli 1.4. Prekrivanje je zelo veliko, saj se razbitji pri večini enot ujemata (ne ujemata se le pri manjšem številu enot).<sup>4</sup>

<sup>4</sup> Večina enot, ki je v prvi skupini po Wardovi metodi, je v drugi po metodi voditeljev, medtem ko je večina tistih, ki so z Wardovo metodo razporejene v drugo skupino, z metodo voditeljev

```

> #Primerjava kriterijskih funkcij
> wardKF(Zemoc,wardCluK2) #Wardova metoda
[1] 4913.525
> kMeansR10$tot.withinss #metoda voditeljev
[1] 4585.664
> #Primerjava končnih razvrstitev po enotah
> table("Wardova metoda"=wardCluK2,
      "Metoda voditeljev"=kMeansR10$cluster)
      Metoda voditeljev
Wardova metoda  1   2
                1  85 267
                2 259  13

```

Tabela 1.4: Primerjava končnih razvrstitev po enotah

	1	2
1	85	267
2	259	13

Razbitje, prikazano v prostoru prvih dveh glavnih komponent, je prikazano na sliki 1.7. Vidi se, da prva glavna komponenta zelo dobro loči med skupinama, dobljenima z metodo voditeljev.

```

> # metoda glavnih komponent
> PC<-princomp(Zemoc)
> plot(PC$scores[,1:2],ylab="2. komponenta",xlab="1. komponenta",
      ,pch=kMeansR10$cluster,col=kMeansR10$cluster)
> legend(x=mean(range(PC$scores[,1])), y=max(PC$scores[,2])+0.5,
      legend = c("1. skupina", "2. skupina"),
      title="Metoda voditeljev",xjust=0.5, yjust=0,xpd=TRUE,
      pch=kMeansR10$cluster,col=kMeansR10$cluster,horiz=TRUE)

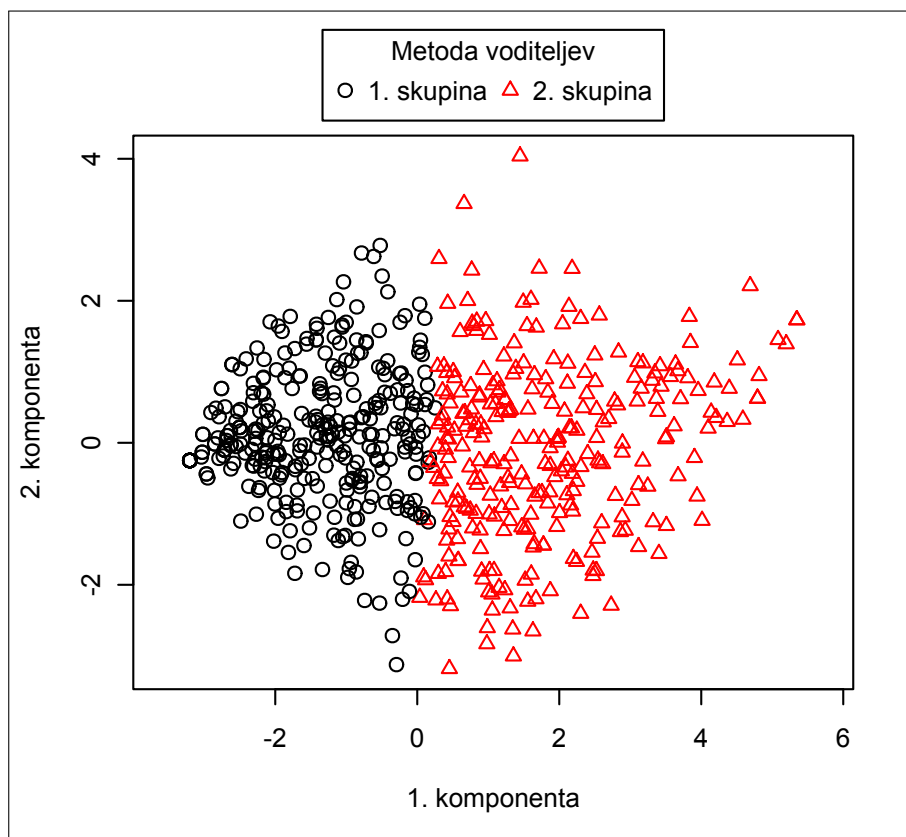
```

S tem zaključujemo uvodni primer. V nadaljevanju sledi pregled osnovnih informacij o delu s programskim paketom **R**.

---

razporejena v prvo skupino. Pri teh enotah se razvrstitvi ujemata, saj oznake skupin (prva, druga) pri razvrščanju v skupine niso pomembne.

Slika 1.7: Rešitev razvrščanja v prostoru glavnih komponent



## 1.2 Osnovne informacije

### 1.2.1 Osnovne računske operacije

Najprej spoznajmo osnovne računske operacije. V **R**-ju se zanje uporabljajo standardni simboli. Morda omenimo le, da je  $\hat{}$  (strešica) znak za potenco, `sqrt` pa funkcija kvadratnega korena.

```
> 2+2
[1] 4
> 6-2
[1] 4
> 3*5
[1] 15
> 2/5
[1] 0.4
> 5%2 #vrne ostanek pri deljenju
```

```
[1] 1
> 5^2 #potenca
[1] 25
> sqrt(25) #kvadratni koren preko funkcije
[1] 5
> 25^(1/2) #in drugače (bolj splošno)
[1] 5
> # zgoraj upoštevamo, da je koren le posebna različica
> # potence, in sicer oblike 1/(stopnja korena)
```

## 1.2.2 Spremenljivke

Vrednost shranimo tako, da jo priredimo spremenljivki (objektu). Za prirejanje uporabimo znak "`<-`" (uporabljamo lahko tudi znak "`=`", a se njegova uporaba odsvetuje). Spremenljivke/objekte nato uporabljamo namesto vrednosti, ki so shranjene v njih.

V imenih spremenljivk lahko nastopajo:

- črke (velike in male – **velikost črk je pomembna**),
- številke,
- znak "`_`" – do verzije 1.8.0 R-ja bil uporabljan kot znak za prirejanje, na kar je treba biti pozoren, če naletite na kakšno zelo staro kodo,
- znak "`.`" – predvsem pri funkcijah ima znak "`.`" v imenu funkcije tudi poseben pomen (uporablja se za prilagoditev funkcije za določen razred objekta), zato se uporaba "`.`" v drugih primerih odsvetuje (ni pa prepovedana).

Ime spremenljivke se mora nujno začeti s črko ali piko ("`.`"). Spremenljivke, ki se začenjajo s piko ("`.`"), so skrite, se pravi, da se ne izpišejo s funkcijo `ls()` (ta funkcija izpiše vse vidne objekte v okolju).

Primer uporabe spremenljivk:

```
> a <- 5 #priredimo vrednost spremenljivki
> a #izpišemo vrednost spremenljivke
[1] 5
> a + 4
[1] 9
> b <- a*2 #nekaj izračunamo in rezultat priredimo
> b
[1] 10
> (c <- a + b) #Kaj se zgodi zaradi oklepajev?
[1] 15
```

Zaradi oklepajev se najprej izvede izraz znotraj oklepaja, torej se rezultat seštevanja shrani v spremenljivko `c`. Nato pa se še "pokliče" oklepaj, kar izpiše vrednost spremenljivke `c`.

### 1.2.3 Uporaba funkcij in pomoči

Funkcija `ls` je prva **R**-jeva funkcija, ki smo jo spoznali. Za pomoč o katerikoli **R**-jevi funkciji napišemo v **R** eno od spodnjih možnosti:

```
?ime_funkcije
help(ime_funkcije)
```

Za iskanje po pomoči lahko uporabite

```
help.search("iskalni pogoj")
??"iskalni pogoj"
```

Dodatno pomoč pa najdete tudi v meniju **Help**.

Za iskanje funkcije, katere ime ustreza določenemu vzorcu, pa uporabite funkcijo `apropos`.

Vsaka spremenljivka je pravzaprav objekt (objekti so tudi funkcije). Objekte lahko brišemo s funkcijo `rm`.

Funkcijo kličemo tako, da navedemo njeno ime in potem v oklepajih `()` njene argumente, ločene z vejico. Nujno moramo navesti vse argumente, ki nimajo privzete vrednosti, ostale pa le, če nam privzeta vrednost ne ustreza. Tudi če ne navedemo nobenega argumenta, moramo nujno navesti oklepaje `()`, kamor v tem primeru ne napišemo nič. Za primer glejte podpodpoglavje [1.3.5 \(Matrika\)](#).

Pri navajanju argumentov imamo dve možnosti:

- Argumente navajamo v takem vrstnem redu, kot so navedeni v funkciji. Ta način lahko uporabimo samo pri argumentih, ki so na začetku oziroma pri katerih smo navedli tudi vse argumente pred njimi.
- Argumente navajamo v poljubnem vrstnem redu skupaj z njihovimi imeni kot dvojice `ime=vrednost`.

V vsakem primeru argumente ločimo z vejicami. Drugi način je dosti varnejši. Prvi se ponavadi uporablja le za prvi argument ali pri zelo pogosto uporabljenih funkcijah, čeprav tudi takrat ni priporočljiv.

```
> ls() #Kličemo funkcijo ls, ne da bi navedli kak argument
[1] "a"          "b"
[3] "c"          "colinEigen"
```

```

[5] "corTestDf"      "d"
[7] "d2"             "ego"
[9] "ego1"           "emoc"
[11] "frekTab"        "i"
[13] "insert"         "kMeansR10"
[15] "makeFactorLabels" "means"
[17] "PC"             "plotMeans"
[19] "printCorTestDf" "razsiriPodatke"
[21] "rekodiraneSprem" "spremEkst"
[23] "spremEmoc"      "spremZaRekod"
[25] "ss"             "ssAllVar"
[27] "tbl"            "ward"
[29] "wardCluK2"      "wardKF"
[31] "Zemoc"

> #Argumenti niso potrebni, ker je pri vseh argumentih
> #privzeta vrednost ustrezna
> #Z njo izpišemo vse vidne objekte - med njimi je tudi a
>
> a<-1 #kreiramo a
> rm(a) #izbrišemo a - argument funkcije je
> #Pozor: argument funkcije je objekt (ne njegovo ime)
> ls() #a-ja ni več med objekti
[1] "b"             "c"
[3] "colinEigen"    "corTestDf"
[5] "d"             "d2"
[7] "ego"           "ego1"
[9] "emoc"          "frekTab"
[11] "i"             "insert"
[13] "kMeansR10"     "makeFactorLabels"
[15] "means"         "PC"
[17] "plotMeans"     "printCorTestDf"
[19] "razsiriPodatke" "rekodiraneSprem"
[21] "spremEkst"     "spremEmoc"
[23] "spremZaRekod" "ss"
[25] "ssAllVar"      "tbl"
[27] "ward"          "wardCluK2"
[29] "wardKF"        "Zemoc"
> ?ls #pomoč za funkcijo ls
> help(ls) #enako
> ??"List Objects" #išče po pomoči
> ls(all.names = TRUE) #izpišemo vse objekte, tudi skrite
[1] ".cairo"        ".Random.seed"
[3] "b"             "c"

```

```

[5] "colinEigen"      "corTestDf"
[7] "d"               "d2"
[9] "ego"            "ego1"
[11] "emoc"           "frekTab"
[13] "i"              "insert"
[15] "kMeansR10"      "makeFactorLabels"
[17] "means"          "PC"
[19] "plotMeans"      "printCorTestDf"
[21] "razsiriPodatke" "rekodiraneSprem"
[23] "spremEkst"      "spremEmoc"
[25] "spremZaRekod"   "ss"
[27] "ssAllVar"       "tbl"
[29] "ward"           "wardCluK2"
[31] "wardKF"         "Zemoc"
> #torej tudi tiste, ki se začnejo s piko (".")
>
> rm(list=ls()) #izbrišemo vse vidne objekte
> #ta ukaz uporabljajte zelo pazljivo
>
> #ustvarimo nekaj spremenljivk
> a<-1;b<-2;aa<-3;aaaabb<-4;daa<-5
> #V R-ju lahko v eno vrstico napišemo tudi več ukazov,
> #ki jih ločimo s podpičji
> #ponavadi to sicer ni priporočljivo
>
> ls(pattern = "aa") #vsi objekti, ki imajo v imenu "aa"
[1] "aa"      "aaaabb" "daa"
> #uporabimo lahko sicer tudi regularne izraze
> #za več informacij vtipkajte ?regexp

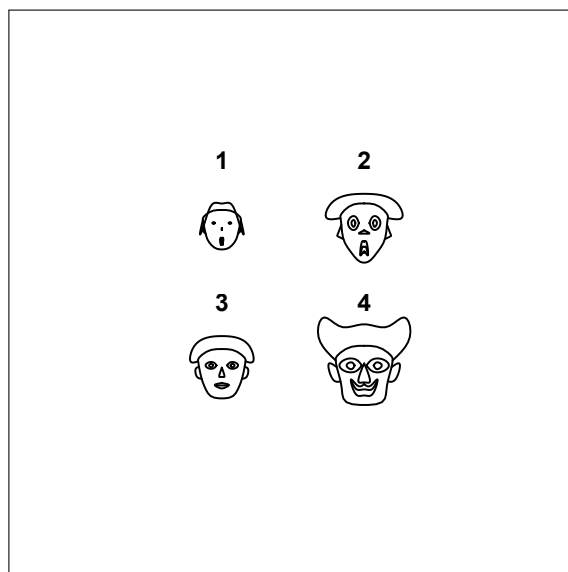
```

### 1.2.4 Paketki

Vse funkcije (ki jih nismo napisali sami) in tudi veliko podatkov je v **R**-ju shranjeno v paketkih. Paketke naložimo preko menija **Packages|Load package...** ali enostavneje s funkcijama *library* ali *require*. Edina razlika med njima je v tem, da prva ob neuspehu (če na primer paketek ni nameščen oziroma inštaliran) javi napako, druga pa vrednost *FALSE*. Da pa paketke lahko **naložimo**, morajo biti prej **nameščeni**.

Na voljo je veliko paketkov. 30 osnovnih paketkov je vključenih že v osnovno distribucijo **R**-ja (in jih imate tako po namestitvi **R**-ja že nameščene), veliko več (trenutno 8551) pa jih je na voljo na CRAN-u (Comprehensive R Archive Network).

Slika 1.8: Štirje obrazi



Paketke najlažje namestimo neposredno iz CRAN-a, in sicer preko menija **Packages|Install package(s)...** ali preko funkcije `install.packages`.

```
> install.packages("TeachingDemos")
package TeachingDemos successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\Ales\AppData\Local\Temp\RtmpcFnMNj\downloaded_packages
> #namestimo paketek s CRAN-a
> library(TeachingDemos) #naložimo paketek
> #funkcija "faces" ni na voljo, če prej ne naložimo paketka
> faces() #uporabimo funkcijo iz paketka - narišemo 4 obraze
```

Rezultat funkcije je na sliki [1.8](#).

## 1.2.5 Drugi osnovni podatki

Prostor, kamor v R-ju pišemo ukaze, se imenuje *ukazna vrstica* oziroma *konzola*. Vanjo lahko ukaze tipkamo, lepimo ali pošljemo iz urejevalnika besedil (če le-ta to omogoča).

Zelo priročno je tudi pomikanje po že uporabljenih ukazih (za ponavljanje ali manjše spremembe) s smernimi tipkami za gor  $\uparrow$  in dol  $\downarrow$ . Po posameznem ukazu pa se premikamo s smernimi tipkami za levo  $\leftarrow$  in desno  $\rightarrow$ .

**R** zapremo preko menija **File|Exit** ali s funkcijama *q* oziroma *quit* (funkciji sta identični). Ko ga ugasnemo, nas **R** tudi vpraša (če nismo tega "povedali" že z argumenti prej omenjenih funkcij), ali želimo shraniti delovno okolje (workspace) in zgodovino.

Stanje *delovnega okolja* (workspace) lahko kadarkoli shranimo s funkcijo *save.image* ali preko menija **File|Save Workspace...**, zgodovino ukazov pa s *savehistory* oziroma preko menija **File|Save History...** Vse datoteke se (če ne izberemo drugače) shranijo v delovno mapo (working directory). Delovno mapo lahko spremenimo preko funkcije *setwd* ali preko menija **File|Change dir...** Način spreminjanja mape preko menija lahko uporabimo tudi le za to, da najdemo trenutno delovno mapo, lahko pa to ugotovimo tudi s funkcijo *getwd*.

Shranjeno okolje lahko potem naložimo na več načinov. Če nismo spreminjali imen (privzeto ime je ".RData" za delovno okolje in ".RHistory" za zgodovino), potem se oboje samodejno naloži, če bodisi zaženemo **R** v isti mapi bodisi zaženemo **R** z dvoklikom na shranjeno okolje ".RData". Delovno okolje lahko naložimo tudi tako, da datoteko "prenesemo" v **R**-jevo okno, s pomočjo funkcije *load.image* ali preko menija **File|Load Workspace...** Podobno velja tudi za zgodovino.

Zgodovino lahko kadarkoli izpišemo tudi s funkcijo *history*.

**R** kot decimalno ločilo uporaba piko (.), na primer  $\frac{1}{2}$  zapišemo kot 0.5. Pri ukazih se tega ne da spreminjati (kar je nujno za konsistentno delovanje kateregakoli programskega jezika), pri izpisih pa to lahko nastavimo preko funkcije *options*. Če želimo kot decimalno ločilo nastaviti vejico (,), torej decimalno ločilo, ki ga uporabljamo v slovenščini, uporabimo ukaz *options(OutDec = ",")*.

V tem učbeniku uporabljam kot decimalno ločilo piko (.), kjub temu, da se zavedam, da je to v nasprotju s slovenskim pravopisom. Razloga za to sta dva, oba pa izhajata iz narave učbenika, torej učenje **R**-ja. Prvi je v tem, da menim, da bi uporaba enega decimalnega ločila (pike) pri ukazih in drugega (vejice) pri izpisih lahko zmedla bralca. Drugi je v tem, da želim, da lahko uporabniki z uporabo ukazov iz učbenika dobijo enak izpis, kot ga podajam v učbeniku. Zato raje (kjer ni potrebno) ne spreminjam začetnih nastavitvev. S pomočjo funkcije *options* je mogoče nastavljanje tudi veliko drugih nastavitvev, na primer širino izpisa (tekstovnega), način izpisa števil (število decimalnih mest, znanstven ali običajen zapis ...). Za pregled in opis nastavitvev, ki jih je mogoče spreminjati, uporabite *?options*.

## 1.3 Podatkovne strukture

### 1.3.1 Osnovni podatkovni tipi

**R** pozna naslednje osnovne podatkovne tipe:

**logical** logični vektor – vrednosti *TRUE* in *FALSE*,

**integer** cela števila,

**numeric** realna števila – to je privzeta vrednost za vsa števila v R-ju, tudi cela, na primer 2,

**complex** kompleksna števila,

**character** znaki (poljubno število) – vedno jih moramo pisati v narekovajih, saj se drugače upoštevajo kot imena spremenljivk ali drugih objektov (funkcij),

**raw** shranjuje surove bajte (ang. byte) – tega ne rabite.

Podatkovne tipe spreminjamo s funkcijami *as.nekaj*, kjer je "nekaj" ime tipa. S funkcijo *is.nekaj* preverimo, ali je nek podatek tipa "nekaj" (torej nekega določene tipa). Kakšnega tipa je nek podatek (objekt), pa preverimo s funkcijo *mode*.

Obstajajo pa še sledeče posebne vrednosti:

**NA** Not available – ni na voljo oziroma R-jeva koda za manjkajočo vrednost.

**NaN** Not a Number – ni število. Relevantno samo pri številskih tipih, uporablja se, kadar rezultat nekega izračuna ni definiran, na primer *0/0*.

**NULL** Absoluten nič v R-ju, prazen element.

Pri prvih dveh (*NA*, *NaN*) je rezultat izračuna, kjer nastopa katera od teh vrednosti, praktično vedno kar ta vrednost, tretji pa ima podobno vlogo kot prazen element.

```
> 4 #realno število
[1] 4
> is.numeric(4)
[1] TRUE
> is.integer(4) #ni celo število
[1] FALSE
> a <- as.integer(4) #smo naredili celo število
> is.integer(a)
[1] TRUE
> a <- "to je znakovna spremenljivka"
> a <- "4" #še vedno znakovna spremenljivka
> a #četudi je znak številka - ne moremo računati
[1] "4"
> as.numeric(a)+3 #po spremembi lahko računamo
[1] 7
> 0/0
[1] NaN
> 1 + NA
[1] NA
```

Pri realnih podatkih prav pridejo spodaj prikazane funkcije za zaokroževanje.

```

> floor(2.6) #navzdol
[1] 2
> ceiling(2.1) #navzgor
[1] 3
> round(2.1) #na najbližjo vrednost
[1] 2
> round(2.6)
[1] 3
> round(1.5) #pozor - uporablja standard IEC 60559
[1] 2
> #vrednosti na meji (5 na koncu) se zaokrožujejo na najbližje
> #sodo število
> round(2.5)
[1] 2
> round(pi,digits=3) #lahko nastavimo tudi število decimalk
[1] 3.142

```

### 1.3.2 Vektor

Vektor je najosnovnejša podatkovna struktura. Vsebuje lahko samo podatke enega tipa. Če želimo ustvariti vektor iz podatkov različnih tipov, jih **R** sam pretvori v tip, ki je najsplošnejši (če ns primer združujemo številke in znake v tip "character"). Vektor ustvarimo tako, da združimo (combine) vrednosti s pomočjo funkcije *c*.

**R** (oziroma njegov predhodnik S) je vektorski jezik, kar pomeni, da se, če ne izberemo le posameznega dela, operacije izvajajo na vseh elementih vektorja.

#### Opozorilo!

**R** dovoli tudi množenje dveh vektorjev različnih dolžin. V primeru, da dolžina daljšega ni mnogokratnik dolžine krajšega, izpiše opozorilo, sicer pa le izračuna rezultat.

Elemente vektorja izbiramo tako, da njihove indekse navedemo v oglatih oklepajih *[ ]* (glejte primer spodaj). Izbrane dele lahko tudi spreminjamo. Negativna vrednost v oglatih oklepajih pomeni, da izberemo vse razen tistega, kar sledi "-".

```

> a<-c(4,6,7,3,1,2) #ustvarimo vektor
> a #izpišemo vektor
[1] 4 6 7 3 1 2
> a + 3
[1] 7 9 10 6 4 5

```

```

> #vsem elementom vektorja a prištejemo 3 (a se seveda ne
> #spremeni)
>
> b<-c(1,2,3,4,5,6)
> a+b #elementi se seštejejo
[1] 5 8 10 7 6 8
> a*b #oziroma zmnožijo
[1] 4 12 21 12 5 12
> b<- c(1,2)
> a+b #b se 3x ponovi, da doseže dolžino a-ja
[1] 5 8 8 5 2 4
> #pri tem R ne izpiše kakršnegakoli opozorila
> a*b
[1] 4 12 7 6 1 4
> length(a) #izpišemo dolžino vektorja a
[1] 6
> length(b)
[1] 2
> length(a*b) #dolžina rezultata je enaka dolžini daljšega
[1] 6
> b<-c(1,2,3,4)
> a*b #b se ponovi 1,5-krat. Izpiše se opozorilo
[1] 4 12 21 12 1 4
> a[3] #izberemo 3. element vektorja a (štetje se začne z 1)
[1] 7
> c<-c(3,6,"d") #če združimo različne elemente
> c #dobimo najsplošnejši tip (znakovni)
[1] "3" "6" "d"
> c[2]<-bla bla #spremenimo 2. element v vektorju b
> c
[1] "3"      "bla bla" "d"
> c[-2] #izpišemo vse elemente razen drugega
[1] "3" "d"

```

Vektor/zaporedje celih števil od  $a$  do  $b$  tvorimo tako, da napišemo  $a:b$ , kjer  $a$  in  $b$  zamenjamo z ustreznima številoma oziroma kjer sta to dve spremenljivki z ustreznima vrednostma. To lahko s pridom uporabljamo tudi pri izbiranju dela vektorja, saj lahko izberemo tudi več kot en element vektorja, tako da kot indeks podamo vektor števil.

Namesto indeksov lahko uporabimo tudi enako (kot originalni) dolg logični vektor (ta ima samo vrednosti *TRUE* in *FALSE*). Izbrani so elementi, kjer je vrednost *TRUE*.

```

> 1:5 #vektor od 1 do 5
[1] 1 2 3 4 5
> a<- -10:10 #kreiramo vektor od -10 do 10 in ga shranimo v a
> a #izpišemo vektor
[1] -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3
[15] 4 5 6 7 8 9 10
> a[5:10] #izpišemo številke na mestih od 5 do 10 (vključno)
[1] -6 -5 -4 -3 -2 -1
> a[5:10][1:3]
[1] -6 -5 -4
> #izmed številke na mestih od 5 do 10 izberemo prve 3
>
> a>0 #dobimo logični vektor, ki ga lahko uporabimo za izbor
[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[10] FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
[19] TRUE TRUE TRUE
> a[a>0] #izberemo samo pozitivna števila
[1] 1 2 3 4 5 6 7 8 9 10
> a[c(1,11,21)] #izberemo številke na mestih 1, 11 in 21
[1] -10 0 10
> a[-c(1,11,21)]
[1] -9 -8 -7 -6 -5 -4 -3 -2 -1 1 2 3 4 5 6 7 8 9
> #izberemo vse razen tistih na mestih 1, 11 in 21
> a[-(1:5)] #izberemo vse razen tistih na mestih od 1 do 5
[1] -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10

```

### Opozorilo!

Če pri zadnjem izrazu spustimo oklepaj, to ni veljavno, ker imajo potem indeksi vrednosti od  $-1$  do  $5$ , kar pa ni dovoljeno. Pri indeksih ne smemo mešati negativnih in pozitivnih števil.

Splošnejša zaporedja dobimo s funkcijo `seq`. Pri pisanju zank je še posebej uporabna funkcija `seq_len`, ki ima samo en argument, in sicer dolžino zaporedja. Njena prednost je v tem, da če je dolžina  $0$ , dejansko ustvari vektor dolžine  $0$ .

```

> seq(from=1, to = 11, by= 2)
[1] 1 3 5 7 9 11
> seq(from=1, to = 12, by= 2)
[1] 1 3 5 7 9 11
> #enako, ker pri koraku 2 število 12 tako ali tako izpustimo
> seq(from=10, to = 0, by= -1) #v obratnem vrstnem redu

```

```

[1] 10 9 8 7 6 5 4 3 2 1 0
> n<-5
> 1:n
[1] 1 2 3 4 5
> seq_len(n) #enako kot zgoraj
[1] 1 2 3 4 5
> n<-0
> 1:n #morda ne ravno tisto, kar smo pričakovali
[1] 1 0
> seq_len(n) #vektor dolžine 0 - bistveno bolj varno pri zankah
integer(0)
> seq(length.out=n) #enako kot prej
integer(0)

```

Elementom vektorja (in tudi drugih struktur, ki jih bomo spoznali kasneje) lahko damo tudi imena. To lahko storimo, ko jih ustvarimo, ali kasneje s funkcijo `names`. S to funkcijo jih tudi priključimo. Imena lahko uporabimo namesto indeksov za dostopanje do posameznih elementov.

```

> a<-1:3 #ustvarimo vektor brez imen (za elemente)
> a #elementi nimajo imen
[1] 1 2 3
> names(a)<-c("a","b","c")
> a #členi imajo sedaj imena, ki se izpišejo nad njimi
a b c
1 2 3
> a["b"] #izberemo 2. element - b
b
2
> #če želimo izbrati več kot en element, moramo uporabiti
> #vektor imen
> a[c("a","c")]
a c
1 3
> names(a)<-NULL #imena izbrišemo
> b<-c("1a"=1,b=2,"3. element"=3) #imena podamo ob nastanku,
> #če imamo v imenih presledke ali če se začnejo s številko,
> #jih moramo dati v narekovaje, sicer ni nujno
> b
      1a          b 3. element
      1           2           3
> names(b)[2]<-"bbb" #spremenimo ime 2. členu
> b

```

1a	bbb 3. element	
1	2	3

### 1.3.3 Nominalne in ordinalne spremenljivke

Čeprav bi nominalne spremenljivke lahko predstavili z znakovno spremenljivko (*character*), je priporočljivo, da za njih uporabljamo tip *factor*. Tak pristop je varnejši (definiramo lahko kategorije) in učinkovitejši. Ustvarimo jih s funkcijo *factor* oziroma spremenimo iz znakovnih spremenljivk s funkcijo *as.factor*.

Zelo pomembni argumenti funkcije *factor* so:

**levels** Vrednosti, ki jih spremenljivka lahko zavzame. Vrednosti morajo torej biti take kot vrednosti originalne spremenljivke. Če je argument podan, se vse vrednosti spremenljivke, ki niso tu navedene, izločijo. Če je vrednost argumenta *ordered=TRUE*, argument poda tudi vrstni red vrednosti. Privzeta vrednost so vse vrednosti spremenljivke v takem vrstnem redu, kot se prvič pojavijo (kar je zelo pomembno za pravilno uporabo argumenta *labels*).

**labels** Imena za posamezne vrednosti. Prvo ime je ime vrednosti, ki je kot prva podana v *levels*, drugo ime za vrednost, ki je podana kot druga v *levels* ... Uporaba *labels* brez *levels* ni priporočljiva. Če pa že uporabljamo *labels* brez *levels*, se moramo zavedati, kakšna je privzeta vrednost za *levels* (glejte zgoraj).

**ordered** Ta pove, ali naj bo faktor "urejen", torej ordinalna spremenljivka.

#### Opozorilo!

Pravilen tip spremenljivke je pri **R**-ju zelo pomemben, ker se upošteva tudi pri statističnih modelih. Pri linearni regresiji na primer **R** iz spremenljivke tipa *factor* (nominalne) samodejno tvori umetne spremenljivke.

Nekaj primerov:

```
> factor(rep(1:3,times=3))
[1] 1 2 3 1 2 3 1 2 3
Levels: 1 2 3
> #nominalna spremenljivka z vrednostmi 1, 2 in 3
> factor(rep(1:3,times=3),levels=1:3, labels=c("a","b","c"))
[1] a b c a b c a b c
Levels: a b c
> #nominalna spremenljivka z vrednostmi "a", "b" in "c"
> factor(rep(1:3,times=3),levels=c("a","b","c"))
```

```

[1] <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>
Levels: a b c
> #vse vrednosti so NA - če je levels specificiran,
> #se kot veljavne vrednosti v originalni spremenljivki
> #upoštevajo samo vrednosti v levels
> factor(rep(1:3,times=3),labels=c("a","b","c"))
[1] a b c a b c a b c
Levels: a b c
> #sicer kot prej z "levels", a bolj nevarno
> #npr., če bi bila prva vrednost 3, bi "a" predstavljal 3
> factor(rep(1:3,times=3),labels=c("a","b","c"),levels=c(3,1,2))
[1] b c a b c a b c a
Levels: a b c
> #ker je podal levels v drugačnem vrstnem redu,
> #sedaj "a" predstavlja 3
>
> factor(rep(1:3,times=3),ordered=TRUE)
[1] 1 2 3 1 2 3 1 2 3
Levels: 1 < 2 < 3
> #ordinalna spremenljivka
> factor(rep(1:3,times=3),ordered=TRUE,levels=c(3,1,2))
[1] 1 2 3 1 2 3 1 2 3
Levels: 3 < 1 < 2
> #drugačen vrstni red kategorij
> factor(rep(1:3,times=3),ordered=TRUE,levels=c(3,1,2),
  labels=c("c","a","b"))
[1] a b c a b c a b c
Levels: c < a < b
> #drugačen vrstni red kategorij z imeni
> #imena smo nastavili tako, da "a" predstavlja 1 ...
> f1<-factor(rep(1:3,times=3),levels=1:3,labels=c("a","b","c"))
> factor(f1,ordered=TRUE,levels=c("c","a","b"))
[1] a b c a b c a b c
Levels: c < a < b
> #v 2 korakih, a morda bolj razumljivo

```

### 1.3.4 Seznam

Seznam (ang. oziroma v R-ju *list*) je posebna oblika vektorja, kjer so elementi lahko karkoli (katerikoli elementi, se pravi posamezna števila, vektorji, matrice, podatkovni okvirji, sezname, celo funkcije).

Ustvarimo jih s funkcijo `list` ali tako, da s funkcijo `c` skupaj zlepimo sezname. Do posameznih elementov dostopamo tako, da njihove indekse ali imena navedemo med **dvojne** oglate oklepaje – `[[ ]]`. Če navedemo imena ali indekse v enojnih oglatih oklepajih `[ ]`, potem je rezultat nov seznam z izbranimi elementi (tudi če izberemo samo enega).

Če imajo elementi imena, lahko do njih dostopamo tudi tako, da za imenom seznama napišemo znak za dolar `$`, ki mu sledi ime elementa.

Na seznamih sta zelo uporabni sestrski funkciji `lapply` in `sapply`. Obe funkciji poskušata neko funkcijo (ki jo podamo kot argument) uporabiti na vseh elementih seznama. Razlika je v tem, da poskuša `sapply` poenostaviti rezultat (v vektor ali matriko), `lapply` pa vedno vrne seznam.

```
> l<-list(a=c("ups", "abc"), b=1:5, c=list(aa="a", bb=5:7), d=list)
> #združili smo različne elemente - zadnji je funkcija list
>
> l[["a"]] #izberemo 1. element - vektor
[1] "ups" "abc"
> l[[1]]   #enako
[1] "ups" "abc"
> l["a"] #naredimo seznam, ki ima samo 1. element starega
$a
[1] "ups" "abc"
> l[1]    #enako
$a
[1] "ups" "abc"
> l[1:3]  #izberemo prve 3 elemente
$a
[1] "ups" "abc"

$b
[1] 1 2 3 4 5

$c
$c$aa
[1] "a"

$c$bb
[1] 5 6 7
> # l[[1:3]] #ne deluje - oziroma ima drugačno funkcijo
>
>
> l[["c"]][[2]][1]
```

```
[1] 5
> #izbrali smo 3. element (c) osnovnega seznama,
> #ki pa je tudi seznam
> #v tem seznamu smo izbrali 2. element, ki je vektor
> #v vektorju smo izbrali 1. element (številko 5)
> l[[3]][[2]][1] #enako
[1] 5
> l[[3]][[2]] #izberemo 2. element v 3. elementu
[1] 5 6 7
> l[[3:2]]      #tokrat deluje in je enako kot zgoraj
[1] 5 6 7
> #močno odsvetujem
>
> l$a #če imamo imena, lahko uporabimo tudi ta način
[1] "ups" "abc"
> l$c$a #deluje tudi v več stopnjah
[1] "a"
> x<-l$d(3,4,6) #če je element seznama funkcija, jo
>                #lahko tudi uporabimo
> x
[[1]]
[1] 3

[[2]]
[1] 4

[[3]]
[1] 6
> l #da se spomnimo, kaj je l
$a
[1] "ups" "abc"

$b
[1] 1 2 3 4 5

$c
$c$a
[1] "a"

$c$b
[1] 5 6 7
```



```
>
> A #ustvarjena matrika
      [,1] [,2] [,3] [,4]
[1,]    1    7   13   19
[2,]    2    8   14   20
[3,]    3    9   15   21
[4,]    4   10   16   22
[5,]    5   11   17   23
[6,]    6   12   18   24
> A[3,2] #izberemo 2. element v 3. vrstici
[1] 9
> #oziroma vse elemente, ki so v 3. vrstici in 2. stolpcu
>
> A[3,] #izberemo 3. vrstico
[1] 3 9 15 21
> A[,2] #izberemo 2. stolpec
[1] 7 8 9 10 11 12
> A[1:3,] #izberemo prve 3 vrstice
      [,1] [,2] [,3] [,4]
[1,]    1    7   13   19
[2,]    2    8   14   20
[3,]    3    9   15   21
> #izberemo elemente, ki so v 2. ali 3. vrstici in
> #v 1. ali 4. stolpcu
> A[c(2,4),c(1,4)]
      [,1] [,2]
[1,]    2   20
[2,]    4   22
> a<-1:4
> b<-2:5
> B<-cbind(a,b) #matrika z dvema stolpcema
> B
      a b
[1,] 1 2
[2,] 2 3
[3,] 3 4
[4,] 4 5
> C<-rbind(a,b) #matrika z dvema vrsticama
> C
      [,1] [,2] [,3] [,4]
a     1    2    3    4
b     2    3    4    5
```

**R** vsebuje tudi jezik za matrično računanje. Pri tem sta najpomembnejša operator za matrično množenje `%%` in funkcija `solve`, ki omogoča izračun inverzne matrike. Nekaj primerov.

```
> A<-matrix(data=1:24,nrow=6)
> #klasično se operacije pri matriki izvajajo tako kot pri
> #vektorjih, torej po elementih
> A+2
      [,1] [,2] [,3] [,4]
[1,]    3    9   15   21
[2,]    4   10   16   22
[3,]    5   11   17   23
[4,]    6   12   18   24
[5,]    7   13   19   25
[6,]    8   14   20   26
> A/2
      [,1] [,2] [,3] [,4]
[1,]  0.5  3.5  6.5  9.5
[2,]  1.0  4.0  7.0 10.0
[3,]  1.5  4.5  7.5 10.5
[4,]  2.0  5.0  8.0 11.0
[5,]  2.5  5.5  8.5 11.5
[6,]  3.0  6.0  9.0 12.0
> A*2
      [,1] [,2] [,3] [,4]
[1,]    2   14   26   38
[2,]    4   16   28   40
[3,]    6   18   30   42
[4,]    8   20   32   44
[5,]   10   22   34   46
[6,]   12   24   36   48
> A+A
      [,1] [,2] [,3] [,4]
[1,]    2   14   26   38
[2,]    4   16   28   40
[3,]    6   18   30   42
[4,]    8   20   32   44
[5,]   10   22   34   46
[6,]   12   24   36   48
> A*A #tudi če je operacija množenje
      [,1] [,2] [,3] [,4]
[1,]    1   49  169  361
[2,]    4   64  196  400
```

```
[3,] 9 81 225 441
[4,] 16 100 256 484
[5,] 25 121 289 529
[6,] 36 144 324 576
> #to ni matrično množenje
>
> A*(1:4)
      [,1] [,2] [,3] [,4]
[1,] 1 21 13 57
[2,] 4 32 28 80
[3,] 9 9 45 21
[4,] 16 20 64 44
[5,] 5 33 17 69
[6,] 12 48 36 96
> #tako kot pri računanju z vektorji, se krajši "reciklira"
> #torej se tolikokrat ponovi, da doseže dolžino daljšega
>
> t(A) #transponiramo matriko - zamenjamo vrstice in stolpce
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,] 1 2 3 4 5 6
[2,] 7 8 9 10 11 12
[3,] 13 14 15 16 17 18
[4,] 19 20 21 22 23 24
> dim(A) #ugotovimo dimenziji matrike
[1] 6 4
> dim(t(A)) #pri transponirani matriki sta ravno obrnjeni
[1] 4 6
> A %% t(A) #matrično množenje
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,] 580 620 660 700 740 780
[2,] 620 664 708 752 796 840
[3,] 660 708 756 804 852 900
[4,] 700 752 804 856 908 960
[5,] 740 796 852 908 964 1020
[6,] 780 840 900 960 1020 1080
> A %% (1:4) #matrično množenje z vektorjem
      [,1]
[1,] 130
[2,] 140
[3,] 150
[4,] 160
[5,] 170
[6,] 180
```

```

> #R samodejno poskrbi, da je vektor ustrezen
> #(stolpični ali vrstični)
> #bolje je sicer, če za to poskrbimo sami
> a<-matrix(1:4,ncol=1) #stolpični vektor
> A %*% a
      [,1]
[1,] 130
[2,] 140
[3,] 150
[4,] 160
[5,] 170
[6,] 180
> #izračunamo lahko tudi inverz matrike
> #seveda samo kvadratne in take z neničelno determinanto
> AA <- A %*% t(A) #naredimo kvadratno matriko
> det(AA) #izračunamo determinanto
[1] 5.494284e-59
> #determinanta je praktično nič, zato te matrike ne moremo
> #uporabiti
>
> B<-matrix(c(1,6,3,7,3,7,3,7,9),ncol=3)
> #naredimo kvadratno matriko, katere determinanta ni 0
> det(B) #izračunamo determinanto
[1] -154
> invB<-solve(B) #izračunamo inverz matrike B
> #shranili smo ga v invB
> invB #izpišemo
      [,1]      [,2]      [,3]
[1,] 0.1428571 2.727273e-01 -0.25974026
[2,] 0.2142857 4.205453e-18 -0.07142857
[3,] -0.2142857 -9.090909e-02 0.25324675
> B %*% invB #dokaz, da smo izračunali inverz
      [,1]      [,2] [,3]
[1,] 1.000000e+00 8.326673e-17 0
[2,] -3.053113e-16 1.000000e+00 0
[3,] -1.387779e-16 1.387779e-16 1
> #rezultat je enotska matrika
> #na diagonali so samo enice, drugod povsod ničle
> #nekatero vrednosti niso 0, ampak skoraj 0 zaradi
> #nenatančnosti izračuna -> nekaterih vrednosti v računalniku
> #ne moremo predstaviti natančno
> zapsmall(B%*%invB)

```

```

      [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> #funkcija zapsmall vrednosti, ki so skoraj 0, spremeni v 0

```

Podobno kot lahko na seznamih uporabljamo funkciji `lapply` in `sapply`, lahko na matrikah uporabimo funkcijo `apply`. Tej moramo poleg matrike in funkcije podati tudi dimenzijo (vrstice ali stolpce), na kateri želimo funkcijo uporabiti.

```

> A<-matrix(data=1:24,nrow=6)
> apply(X=A,MARGIN=1,FUN=sum) #vsota po vrsticah
[1] 40 44 48 52 56 60
> apply(X=A,MARGIN=2,FUN=sum) #vsota po stolpcih
[1] 21 57 93 129

```

### 1.3.6 Polje – Array

Nadgradnja matrike je **polje** (ang. **array**). Od matrike se razlikuje po tem, da ima poljubno število dimenzij. Za več pogledjte `?array`. Tudi na poljih je mogoče uporabiti funkcijo `apply`.

### 1.3.7 Podatkovni okvir – Data frame

Podatkovni okvir je podatkovna struktura, ki je običajno najprimernejša za shranjevanje podatkovij. Ima podobno vlogo in strukturo kot podatkovne datoteke pri ostalih statističnih programih, na primer SPSS-u. V vrsticah so enote, v stolpcih pa spremenljivke oziroma "vektorji".

Podatkovni okvirji imajo tako nekatere lastnosti matrik, pa tudi nekatere lastnosti seznamov. Po eni strani so podobni matrikam, vendar pa lahko pri podatkovnih okvirjih stolpci hranijo različne tipe podatkov. Še bolj so podobni seznamom, saj pravzaprav so sezname, kjer so vsi elementi vektorji enake dolžine. Se pravi so sezname, kjer je vsak element (v statističnem smislu) neka spremenljivka.

Tudi do elementov lahko dostopamo tako kot pri seznamih ali tako kot pri matrikah.

Podatkovni okvir ustvarimo s funkcijo `data.frame` ali tako, da vanj preoblikujemo matriko ali seznam (ki mora biti ustrezne oblike) s funkcijo `as.data.frame`.

```

> #ustvarimo nekaj vektorjev
> #generiramo vrednosti slučajno iz normalne porazdelitve

```

```
> x1<-rnorm(n=10,mean=176,sd=7)
> x2<-rnorm(n=10,mean=100,sd=15)
> #ustvarimo slučajen vektor z neštevilskimi elementi
> x3<-sample(c("a","b","c"),size=10,replace=TRUE)
> #izberemo vzorec s ponavljanjem velikosti 10
> podatki<-data.frame(x1,x2,x3)
> podatki
      x1      x2 x3
1 177.2191 116.93152 c
2 168.5902 100.92010 c
3 182.0382 120.62572 b
4 180.0462 102.95941 b
5 178.8490  99.00317 b
6 176.8851 102.02235 a
7 174.1618  88.16272 b
8 183.5689  84.34124 c
9 172.4907  98.02423 c
10 172.6332 116.36876 a
> names(podatki) #imena kot pri seznamu
[1] "x1" "x2" "x3"
> dim(podatki) #preverimo dimenzije kot pri matriki
[1] 10  3
> colnames(podatki) #preverimo imena stolpcev kot pri matriki
[1] "x1" "x2" "x3"
> dimnames(podatki) #kot pri matriki - imena stolpcev in vrstic
[[1]]
 [1] "1" "2" "3" "4" "5" "6" "7" "8" "9" "10"

[[2]]
[1] "x1" "x2" "x3"
> #izbiramo stolpce
> podatki[,"x1"] #kot pri matriki (stolpci so 2. dimenzija)
 [1] 177.2191 168.5902 182.0382 180.0462 178.8490 176.8851
 [7] 174.1618 183.5689 172.4907 172.6332
> podatki$x1 #kot pri seznamu
 [1] 177.2191 168.5902 182.0382 180.0462 178.8490 176.8851
 [7] 174.1618 183.5689 172.4907 172.6332
> podatki["x1"]
      x1
1 177.2191
2 168.5902
3 182.0382
4 180.0462
```

```
5 178.8490
6 176.8851
7 174.1618
8 183.5689
9 172.4907
10 172.6332
> #kot pri seznamu - to je še vedno podatkovni okvir
>
> podatki[c("x1", "x3")]
      x1 x3
1 177.2191 c
2 168.5902 c
3 182.0382 b
4 180.0462 b
5 178.8490 b
6 176.8851 a
7 174.1618 b
8 183.5689 c
9 172.4907 c
10 172.6332 a
> #kot pri seznamu - to je še vedno podatkovni okvir
> podatki[["x1"]] #kot pri seznamu - to je zdaj vektor
[1] 177.2191 168.5902 182.0382 180.0462 178.8490 176.8851
[7] 174.1618 183.5689 172.4907 172.6332
> #vektor dobimo (eno samo spremenljivko, ker smo uporabili
> #dvojne oglate oklepaje)
>
> #vrstice
> podatki[1:3,] #kot pri matriki (vrstice so 2. dimenzija)
      x1      x2 x3
1 177.2191 116.9315 c
2 168.5902 100.9201 c
3 182.0382 120.6257 b
> #elemente
> podatki[2, "x1"] #kot pri matriki
[1] 168.5902
> #tako lahko tudi več naenkrat
> podatki[2:4, c("x1", "x3")] #kot pri matriki
      x1 x3
2 168.5902 c
3 182.0382 b
4 180.0462 b
> podatki$x1[1:3] #kot pri seznamu - samo v enem stolpcu naenkrat
```

```
[1] 177.2191 168.5902 182.0382
> podatki[["x1"]][1:3] #enako na drug način
[1] 177.2191 168.5902 182.0382
```

### 1.3.8 Vaje

#### Vaja 1

Recimo, da imate sledeče vrednosti: 10, 23, 43, 43, 32, 12.

Izračunajte povprečje, varianco in standardni odklon. Za izračun vsote lahko uporabite funkcijo `sum`, naprednejših vgrajenih funkcij pa ne uporabljajte (za varianco, standardni odklon ...).

#### Vaja 2

Recimo, da imamo tri spremenljivke –  $y$ ,  $x_1$  in  $x_2$ , ki imajo sledeče vrednosti:

$y$ : 10, 23, 43, 43, 32, 12

$x_1$ : 1.5, 2.3, 5.4, 4.5, 4.2, 1.0

$x_2$ : 210, 183, 186, 164, 175, 200

Recimo, da je  $y$  odvisna spremenljivka ter da sta  $x_1$  in  $x_2$  neodvisni spremenljivki. S pomočjo matričnega računanja izračunajte vrednosti regresijskih koeficientov, napovedanih vrednosti in vrednosti rezidualov. Poskrbite, da bodo vsi trije rezultati vektorji (rezultat matričnega računanja je matrika).

Vse tri rezultate shranite v seznam.

*Namig:* Matrična formula za izračun regresijskih koeficientov je:

$$\beta = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'y$$

Matrika  $\mathbf{X}$  mora biti sestavljena iz stolpca enic in po enega stolpca za vsako neodvisno spremenljivko. Formula za izračun napovedi pa je:

$$y' = \mathbf{X}\beta$$

## 1.4 Funkcije in programiranje

Zaporedje ukazov, ki jih pogosto uporabljamo skupaj, je smiselno združiti skupaj v funkcijo. To omogoča, da jih zelo enostavno večkrat uporabimo na različnih podatkih.

### 1.4.1 Nekaj koristnih funkcij

Poleg statističnih funkcij, **R** vsebuje tudi veliko funkcij za manipulacijo s podatki. Navedimo le nekaj najpomembnejših.

**sort** Uredi enote (v številskem vektorju po velikosti).

**order** Vrne zaporedno številko vsake enote, če jih uredimo glede na določen vektor. Zelo uporabno za "sortiranje" podatkovnih okvirjev.

**na.omit** Iz podatkovnega okvirja izbriše vse enote (vrstice), kjer nastopajo manjkajoče vrednosti. To nam omogoča, da pri vseh spremenljivkah upoštevajo iste enote.

**attributes** Vrne lastnosti (ang. "attributes") nekega objekta.

**sample** Vrne slučajen vzorec izbrane velikosti iz izbranega vektorja ali seznama. Privzeto je vzorčenje brez ponavljanja, lahko pa tudi nastavimo vzorčenje s ponavljanjem. Če želimo, lahko tudi nastavimo verjetnosti za posamezne elemente.

**which** Vrne indekse (zaporedne številke), pri katerih je vrednost izraza **TRUE** (nek pogoj je izpolnjen).

Poleg funkcij so pomembni tudi operatorji, ki so sicer tudi sami neke vrste funkcija.

**Operatorji primerjav** **==** (je enako – **dva enačaja!!!**), **>=**, **<=**, **>**, **<**, **!=** (ni enako).

**!** Operator negacije. Uporablja se z logičnimi vrednostmi – **TRUE** spremeni v **FALSE** in obratno

**%in%** Operator vsebovanja. Pove, ali so elementi 1. vektorja vsebovani v 2. elementu.

```
> x1 <- podatki$x1
> x1 #prvotna razporeditev
[1] 177.2191 168.5902 182.0382 180.0462 178.8490 176.8851
[7] 174.1618 183.5689 172.4907 172.6332
> sort(x1) #vrednosti, urejene po velikosti naraščajoče
[1] 168.5902 172.4907 172.6332 174.1618 176.8851 177.2191
[7] 178.8490 180.0462 182.0382 183.5689
> sort(x1, decreasing = TRUE) #urejene po velikosti padajoče
[1] 183.5689 182.0382 180.0462 178.8490 177.2191 176.8851
[7] 174.1618 172.6332 172.4907 168.5902
> order(x1) #zaporedne št. v takem vrstnem redu,
[1] 2 9 10 7 6 1 5 4 3 8
> #da bi bile vrednosti urejene po velikosti
> x1[order(x1)] #enak rezultat kot sort(x1)
```

```

[1] 168.5902 172.4907 172.6332 174.1618 176.8851 177.2191
[7] 178.8490 180.0462 182.0382 183.5689
> podatki #originalni vrstni red
      x1      x2 x3
1 177.2191 116.93152 c
2 168.5902 100.92010 c
3 182.0382 120.62572 b
4 180.0462 102.95941 b
5 178.8490  99.00317 b
6 176.8851 102.02235 a
7 174.1618  88.16272 b
8 183.5689  84.34124 c
9 172.4907  98.02423 c
10 172.6332 116.36876 a
> podatki[order(podatki$x3),]
      x1      x2 x3
6 176.8851 102.02235 a
10 172.6332 116.36876 a
3 182.0382 120.62572 b
4 180.0462 102.95941 b
5 178.8490  99.00317 b
7 174.1618  88.16272 b
1 177.2191 116.93152 c
2 168.5902 100.92010 c
8 183.5689  84.34124 c
9 172.4907  98.02423 c
> #podatki, urejeni po velikosti x3 (a,b,c)
> podatki[order(podatki$x3,podatki$x1),]
      x1      x2 x3
10 172.6332 116.36876 a
6 176.8851 102.02235 a
7 174.1618  88.16272 b
5 178.8490  99.00317 b
4 180.0462 102.95941 b
3 182.0382 120.62572 b
2 168.5902 100.92010 c
9 172.4907  98.02423 c
1 177.2191 116.93152 c
8 183.5689  84.34124 c
> #podatki, urejeni najprej glede na x3
> #znotraj kategorij x3 pa glede na x1
>
> podatkiNA<-podatki #skopiramo podatke

```

```

> #dodajmo v podatke manjkajoče vrednosti NA
> podatkiNA[3,"x1"]<-NA
> podatkiNA[6,"x2"]<-NA
> podatkiNA[2,"x3"]<-NA
> podatkiNA[3,"x2"]<-NA
> podatkiNA #podatki z manjkajočimi vrednostmi
      x1      x2  x3
1 177.2191 116.93152  c
2 168.5902 100.92010 <NA>
3      NA      NA  b
4 180.0462 102.95941  b
5 178.8490  99.00317  b
6 176.8851      NA  a
7 174.1618  88.16272  b
8 183.5689  84.34124  c
9 172.4907  98.02423  c
10 172.6332 116.36876  a
> podatkiNAomit<-na.omit(podatkiNA)
> #odstranimo podatke z manjkajočimi enotami
> podatkiNAomit
      x1      x2 x3
1 177.2191 116.93152  c
4 180.0462 102.95941  b
5 178.8490  99.00317  b
7 174.1618  88.16272  b
8 183.5689  84.34124  c
9 172.4907  98.02423  c
10 172.6332 116.36876  a
> attributes(podatkiNAomit)$na.action
2 3 6
2 3 6
attr(,"class")
[1] "omit"
> #informacija o tem, katere vrstice smo odstranili
>
> sample(x=1:100,size=5)
[1] 94 55 41 83 23
> #slučajni vzorec 5 enot iz vektorja 1:100 brez ponavljanja
> sample(x=1:5,size=5) #permutacija (ker porabimo vse enote)
[1] 4 3 1 5 2
> abc<-c("a","b","c")
> sample(x=abc,size=10,replace=TRUE) #vzorec s ponavljanjem
[1] "c" "b" "c" "b" "a" "a" "b" "a" "b" "c"

```

```

> sample(x=abc,size=10,replace=TRUE, prob=c(0.6,0.2,0.2))
[1] "a" "a" "b" "c" "b" "a" "c" "a" "c" "c"
> #vzorec s ponavljanjem - različne verjetnosti izbora
>
> abc<-c("a","b","c")
> rep(abc,times=3) #vektor ponovimo 3-krat
[1] "a" "b" "c" "a" "b" "c" "a" "b" "c"
> rep(abc,each=3) #vsak element ponovimo 3-krat
[1] "a" "a" "a" "b" "b" "b" "c" "c" "c"
> rep(abc,each=3,times=2)
[1] "a" "a" "a" "b" "b" "b" "c" "c" "c" "a" "a" "a" "b" "b"
[15] "b" "c" "c" "c"
> #vsak element ponovimo 3-krat, vektor 2-krat
>
>
> x<-sample(x=abc,size=10,replace=TRUE, prob=c(0.6,0.2,0.2))
> x
[1] "b" "b" "c" "a" "a" "a" "c" "c" "a" "a"
> which(x=="a") #na katerih mestih so a-ji
[1] 4 5 6 9 10
> x[which(x=="a")]
[1] "a" "a" "a" "a" "a"
> x %in% c("b","c")
[1] TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE
[10] FALSE
> !(x=="a") #enako
[1] TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE FALSE
[10] FALSE

```

Poleg zgornjih je treba omeniti še funkcije za generiranje psevdoslučajnih števil. **R** vsebuje mnogo funkcij za generiranje slučajnih vrednosti iz različnih porazdelitev. Naj omenimo samo najpomembnejše:

`rnorm` normalna porazdelitev

`runif` enakomerna porazdelitev

`rbinom` binomska porazdelitev

`rbeta` beta porazdelitev (zelo fleksibilna)

`rexp` eksponentna porazdelitev

`rf` F porazdelitev

`rt` t porazdelitev

`r???` Še kar nekaj drugih porazdelitev. Ime funkcije se vedno začne z "r", sledi oznaka (ime ali okrajšava) za porazdelitev.

Vse zgoraj navedene funkcije kot prvi argument sprejmejo  $n$ , ki pove, koliko števil želimo generirati. Poleg tega kot dodatne argumente sprejmejo parametre porazdelitve. **Pozor:** Porazdelitvene funkcije so lahko definirane drugače, kot ste navajeni.

Vse zgoraj navedene funkcije imajo tudi "sestre", ki se namesto z "r" začnejo z:

d funkcija gostote

p (kumulativna) porazdelitvena funkcija

q kvantilna funkcija – vrne vrednost, ki pripada določnem kvantilnem rang

Pomembna je tudi funkcija `set.seed`, kjer lahko nastavimo seme (koristno, če bi še kdaj želeli generirati enake vrednosti).

Spodaj je nekaj primerov za generiranje slučajnih števil iz različnih porazdelitev.

```
> set.seed(1)
> rnorm(n=5)
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
> #5 slučajnih števil iz standardizirane normalne porazdelitve
> rnorm(n=5, mean=10, sd=2) #drugačno povprečje in sd
[1]  8.359063 10.974858 11.476649 11.151563  9.389223
> rexp(n=10, rate=0.5) #10 števil iz eksponentne porazdelitve
[1] 0.294092 2.781470 1.524060 2.475207 8.847868 2.109086
[7] 2.070488 3.752070 1.309493 0.673867
> rbeta(n=5, shape1=1, shape2=1)
[1] 0.2057601 0.2762891 0.1790537 0.2170672 0.4702804
> #kovanec vržemo 100-krat
> #predvidevamo, da je pošten (prob=0.5)
> #kolikokrat pade cifra
> #poskus ponovimo 10-krat
> rbinom(n=10, size=100, prob=0.5)
[1] 46 56 50 52 42 44 49 54 39 52
```

## 1.4.2 Definiranje funkcij ★

Funkcija je zaporedje ukazov, ki se izvrši na vhodnih parametrih (če jih funkcija ima). Funkcija lahko kaj vrne, ni pa nujno. Osnovna sintaksa funkcije je:

```
ime_funkcije <- function(argumenti){
  izraz1
  izraz2
  izraz3
  return(nek_objekt)
}
```

Izrazi so lahko katerikoli veljavni **R** izrazi. Ukaz `return` ni nujen. Če ukaza `return` ni, funkcija vrne rezultat zadnjega izračuna. Če želimo, da funkcija ne vrne ničesar (če jo uporabljamo zaradi "stranskih učinkov", kot je na primer risanje, pisanje v datoteko ...), potem moramo eksplicitno napisati `return(NULL)`. Funkcija se vedno zaključi pri ukazu `return` (če ta obstaja). Karkoli za njim se ne izvede.

Argumente navedemo tako, da navedemo njihova imena. Če jim želimo pripisati tudi privzete vrednosti (le-te se uporabijo, če ob klicu ne navedemo drugih), navedemo argumente kot:

```
ime1=privzeta_vrednost1, ime2=privzeta_vrednost2 itd.
```

Poseben argument je tudi argument "...". Če vsebuje funkcija ta argument, sprejme tudi katerikoli argument, ki ni eksplicitno naveden v definiciji funkcije, uporablja pa se predvsem za posredovanje argumentov funkcijam znotraj funkcije.

V funkciji seveda lahko uporabimo tudi druge funkcije in tudi *funkcijo, ki jo pišemo*. Slednje je lahko zelo močno orodje, a tudi zelo nevarno (nujna je uporaba pogojev).

```
> #funkcija, ki izračuna ploščino kroga
> plKroga<-function(r){
  #Funkcija ima samo en argument in sicer polmer kroga.
  #Zamiki niso potrebni, je pa koda tako preglednejša.
  pl<-pi*r^2 #pi je vrednost konstante pi.
  return(pl) #Ta vrstica pravzaprav ni nujna,
  #saj funkcija tudi sicer vrne pl,
  #ker je to rezultat zadnjega izračuna,
  #je pa to dobra praksa.
}
> plKroga(10) #pokličemo funkcijo
[1] 314.1593
> #rezultat lahko tudi shranimo
> plR10<-plKroga(10)
> plR10
[1] 314.1593
```

### 1.4.3 Programski tok ★

Pri kateremkoli programu je zelo pomembno usmerjanje programskega toka.

V ta namen so v **R**-ju na voljo sledeči ukazi:

`if` Pogojni stavek. Telo stavka se izvrši, če je pogoj izpolnjen oziroma če je vrednost pogoja `TRUE`. Ukaz ima obliko:

```
if(pogoj){izraz1} else {izraz2}
```

*pogoj* je nekaj, kar ima kot rezultat *TRUE* ali *FALSE*. Če je vrednost *pogoja* *TRUE*, se izvede *izraz1*, sicer pa *izraz2*. *Pozor*: R smatra 0 kot *FALSE*, vse ostale številke pa kot *TRUE*.

*else* del ni obvezen. Če ga ni, se v primeru, da je vrednost *pogoja* *FALSE*, ne izvede nič.

**while** Pogojno ponavljanje/zanka. Telo stavka se ponavlja, dokler je pogoj izpolnjen oziroma dokler je vrednost *pogoja* *TRUE*. Ukaz ima obliko:

```
while(pogoj){izraz}
```

**repeat** Neskončno ponavljanje oziroma neskončna zanka. Telo stavka se ponovi, dokler znotraj ne naleti na ukaz *break* (glejte naprej). Ukaz ima obliko:

```
repeat{izraz}
```

### Opozorilo!

Ukaz *repeat* je zelo nevaren. Nujno moramo v *izraz* zapisati ukaz *break*, sicer dobimo *neskončno zanko* (smrt vsakega programa).

**for** Ponavljanje/zanka preko zaporedja. Telo stavka se ponavlja, tolikokrat, da zanka "obdelava" vse elemente zaporedja. Ukaz ima obliko:

```
for(spremenljivka in zaporedje) {izraz}
```

*izraz* se ponavlja, dokler *spremenljivka* ne pride čez vse elemente *zaporedja*. Zanka se torej ponovi tolikokrat, kolikor je elementov zaporedja.

*spremenljivka* ima v vsaki ponovitvi drugačno vrednost, vrednost naslednjega elementa iz *zaporedja*.

*zaporedje* je ponavadi nek vektor (lahko tudi seznam), zelo pogosto kar rezultat funkcije *seq* oziroma izraza *a:b*.

**break** Prekinitev zanke. Ukaz se uporablja le znotraj zank in sproži takojšnjo prekinitev zanke.

**next** Naslednji. Ukaz se uporablja le znotraj zank in sproži začetek naslednje ponovitve (trenutna ponovitev se na mestu ukaza konča).

Uporaba ukazov *break* in *next* je smiselna le znotraj *if* stavka (ko se ukaza izvedeta le pod določenim pogojem), saj zanka sicer nima smisla.

```
> x<-1
> if(x==1){print("ok")}else{print("ups")}
[1] "ok"
> #pogoj je bil izpolnjen, zato se izpiše "ok"
>
> x<-2
> if(x==1){print("ok")}else{print("ups")}
[1] "ups"
```

```
> #pogoj ni bil izpolnjen, zato se izpiše "ups"
>
> for(i in 1:5){
    print(i)
  }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
> x<-1
> while(x<=5){
    print(x)
    x<-x+1
  }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
> x<-1
> repeat{
    print(x)
    x<-x+1
    if(x>5) break
  }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
> for(i in 1:5){
    if(i%%2==0) next
    print(i)
  }
[1] 1
[1] 3
[1] 5
```

## 1.4.4 Vaje

### Vaja 1

Za probleme iz vaj v prejšnjem podpoglavju (podpoglavje [Podatkovne strukture](#), podpoglavje [1.3.8](#) na strani [38](#)) napišite funkcije tako, da jih bo moč uporabiti na poljubnem številu enot. Prikažite uporabo.

### Vaja 2

Z uporabo zank napišite funkcijo za izračun vsote vseh elementov vektorja.

### Vaja 3

Napišite funkcijo, ki izračuna fakulteto poljubnega naravnega števila. Poskusite napisati funkcijo z zankami in z uporabo rekurzivne funkcije (funkcije v funkciji).

*Namig:* Fakulteto števila izračunamo po naslednji funkciji.

$$n! = n \cdot (n - 1) \cdot (n - 2) \cdots 2 \cdot 1 = \prod_{k=1}^n k = n \cdot (n - 1)!$$

**Pozor:** Fakulteta hitro postane zelo veliko število, zato funkcije preizkušajte na manjših številih (recimo do 10).

### Vaja 4

Napišite funkcijo za izračun mediane.

*Namig 1:* Za ureditev enot po vrsti lahko uporabite funkcijo `sort`.

*Namig 2:* Za ugotovitev, ali je število enot sodo ali liho, pride prav operator `%`, ki vam vrne ostanek pri deljenju.

## 1.5 Delo z datotekami

Nekaj dela z datotekami (shranjevanje in nalaganje delovnega okolja in zgodovine) smo že obdelali v podpoglavju [1.2.5](#). Tu pa si bomo pogledali predvsem funkcije za branje in pisanje podatkov.

### 1.5.1 Tekstovne datoteke

Tekstovne datoteke so datoteke, ki jih lahko beremo z urejevalniki besedila (na primer TextPad, Notepad, Notepad++ ...) in ne z urejevalniki dokumentov, kot so Microsoft (Office) Word ali OpenOffice.org Writer. Ponavadi so podatki urejeni tako, da so v stolpcih spremenljivke, v vrsticah pa enote. Stolpci so ločeni z nekim znakom, ki je lahko tabulator (`\t`), podpičje (`;`), vejica (`,`), presledek () ...

Za branje tekstovnih datotek se najpogosteje uporablja funkcija `read.table` in njene izpeljanke. Izpeljanke so identične originalni funkciji, le da imajo drugačne privzete vrednosti, ki so primerne za branje določenih formatov tekstovnih datotek. Predvsem se pogosto uporabljata funkciji (za ostale glejte pomoč za funkcijo `read.table`):

`read.csv2` Za branje datotek "csv". Na koncu imena je "2", ker je to verzija za datoteke "csv", ki za ločilo uporabljajo podpičje (`;`), vejico (`,`) pa za decimalno ločilo, kot je standard pri nas. Obstaja tudi verzija brez končnice "2", ki sledi ameriškemu standardu. Take datoteke znajo brati in pisati programi za delo s preglednicami, na primer Excel. Tak program je običajno tudi privzeti program za odpiranje takih datotek.

`read.delim2` Za branje datotek, ki so ločene z tabulatorjem (`\t`). Na koncu imena je "2", ker je to verzija za datoteke, ki za decimalno ločilo uporabljajo vejico (`,`), kot je standard pri nas. Obstaja tudi verzija brez "2", ki sledi ameriškemu standardu. Tudi take datoteke znajo brati in pisati programi za delo s preglednicami, na primer Excel.

Funkcija `read.table` in njene izvedenke kot glavni argument sprejmejo ime datoteke (s potjo, če je potrebno), vrnejo pa podatkovni okvir. Imajo še vrsto argumentov, s katerimi je mogoče določati, kako se obravnavajo znakovne spremenljivke, kodiranje ...

Sedaj bomo prebrali tekstovno datoteko "izborESSn30.txt". Njenih prvih nekaj vrstic je videti takole:

```
"spol" "izobrazba" "kraj" "placa"
"1" "zenski" 12 "kmetija" 170
"2" "moski" 12 "manjse mesto" 140
"3" "moski" 11 "manjse mesto" 130
"4" "zenski" 14 "predmestje" 150
```

V 1. vrstici so imena spremenljivk, v 1. stolpcu pa "imena" vrstic. Prva vrstica (imena spremenljivk) se začne kar s prvo spremenljivko (in ne z imenom stolpca za imena vrstic), kar funkcija `read.delim` (obe verziji, saj se razlikujeta le v decimalnem ločilu) tudi pričakuje. Stolpci so ločeni s tabulatorjem, čeprav to iz zgornjega zapisa ni vidno, v boljših urejevalnikih besedila pa je mogoče nastaviti, da so vidni.

Narekovaji okoli nenumeričnih spremenljivk sicer niso nujni (kadar so vrednosti ločene s tabulatorji), so pa priporočljivi.

```
> izbor<-read.delim(file="izborESSn30.txt")
> izbor[1:4,] #izpišemo samo zgornje vrstice
  spol izobrazba      kraj placa
1 zenski      12      kmetija  170
2 moski      12 manjse mesto  140
3 moski      11 manjse mesto  130
4 zenski      14  predmestje  150
> sapply(izbor,class) #pogledamo tipe spremenljivk
  spol izobrazba      kraj  placa
"factor" "integer" "factor" "integer"
```

Ko smo prebrali podatke, smo izpisali tipe spremenljivk. Nenumerične spremenljivke so v R-ju postale spremenljivke tipa *factor*. Če tega ne želimo, če želimo, da so tipa *character* lahko to spremenimo z argumentom *stringsAsFactors*, ki ga nastavimo na *FALSE*.

Za pisanje tekstovnih datotek se najpogosteje uporablja funkcija *write.table* in njene izpeljanke (*write.csv* in *write.csv2*).

Podatki, ki jih želimo zapisati s funkcijo *write.table*, morajo biti v obliki *matrike* ali *podatkovnega okvirja*.

Za bolj splošno branje tekstovnih datotek (v obliki tabel) se uporablja funkcija *scan*, za pisanje takih datotek pa funkcija *cat*. Za več informacij glejte njuno pomoč.

```
> x1<-rnorm(n=10,mean=176,sd=7)
> x2<-rnorm(n=10,mean=100,sd=15)
> #in še eno nenumerično
> x3<-sample(c("a","b","c"),size=10,replace=TRUE)
> #izberemo vzorec s ponavljanjem velikosti 10
> podatki<-data.frame(x1,x2,x3)
> write.table(x=podatki, file="podatki.txt" ,sep = "\t",
  dec = ",", row.names = FALSE)
> #sep = "\t" - ločilo stolpcev je tabulator
> #dec = "," - decimalno ločilo je vejica
> #row.names = FALSE - imena vrstic ne shranimo
> #tako datoteko lahko preberemo z read.delim2
```

## 1.5.2 Shranjevanje in branje objektov

Za shranjevanje **R**-jevih objektov je najprimernejša funkcija `save`. Je zelo podobna funkciji `save.image`, ki smo jo spoznali v podpodpoglavju 1.2.5. Razlika je le v tem, da funkcija `save` shrani le izbrane objekte, `save.image` pa vse objekte v delovnem okolju.

Objekte, ki jih shranimo s funkcijo `save`, lahko preberemo s funkcijo `load`, niso pa primerni za branje z drugimi programi ali z urejevalniki besedila. Običajno damo datotekam končnico `.RData`.

```
> save(podatki, file="podatki.Rdata")
> rm(podatki)
> ls() #objekta podatki ni več
 [1] "a"          "A"          "aa"
 [4] "AA"        "aaaabb"    "abc"
 [7] "b"          "B"          "c"
[10] "C"          "daa"       "f1"
[13] "i"          "invB"      "izbor"
[16] "l"          "n"         "plKroga"
[19] "plR10"     "podatkiNA" "podatkiNAomit"
[22] "x"          "x1"        "x2"
[25] "x3"
> load(file="podatki.Rdata")
> podatki #so spet na voljo
      x1      x2 x3
1 174.2265 90.81960 b
2 180.8787 105.11680 a
3 179.8966 83.05955 c
4 171.1787 121.49536 b
5 171.0475 129.70600 b
6 178.5521 94.49168 a
7 181.3797 84.33798 a
8 175.2136 108.54579 b
9 182.1678 97.97418 b
10 178.7867 136.02427 a
```

Drug način, ki pa ne deluje dobro pri čisto vseh objektih, je uporaba funkcije `dump` za pisanje in `source` za branje. Funkcija `dump` poskuša predstaviti objekt z besedilom in to besedilo zapisati v tekstovno datoteko. Deluje vedno, kadar je mogoče objekt predstaviti z besedilom. Ta način je posebej primeren za shranjevanje/nalaganje funkcij, funkcija `source` pa tudi za nalaganje kode, ki smo jo na primer napisali v urejevalniku besedila. Običajno damo datotekam končnico `.R`.

Funkcija `dump` je uporabna tudi, ko želimo videti predstavitev objekta z besedilom.

```

> kvadrat<-function(x)x^2
> ls()
 [1] "a"          "A"          "aa"
 [4] "AA"        "aaaabb"    "abc"
 [7] "b"          "B"          "c"
[10] "C"          "daa"       "f1"
[13] "i"          "invB"      "izbor"
[16] "kvadrat"   "l"         "n"
[19] "plKroga"   "plR10"     "podatki"
[22] "podatkiNA" "podatkiNAomit" "x"
[25] "x1"        "x2"        "x3"
> dump(list=c("podatki","kvadrat"), file="razno.R")
> rm(list=c("podatki","kvadrat"))
> ls() #objektov podatki in kvadrat ni več
 [1] "a"          "A"          "aa"
 [4] "AA"        "aaaabb"    "abc"
 [7] "b"          "B"          "c"
[10] "C"          "daa"       "f1"
[13] "i"          "invB"      "izbor"
[16] "l"          "n"         "plKroga"
[19] "plR10"     "podatkiNA" "podatkiNAomit"
[22] "x"         "x1"        "x2"
[25] "x3"
> source(file="razno.R")
> ls() #pa sta nazaj
 [1] "a"          "A"          "aa"
 [4] "AA"        "aaaabb"    "abc"
 [7] "b"          "B"          "c"
[10] "C"          "daa"       "f1"
[13] "i"          "invB"      "izbor"
[16] "kvadrat"   "l"         "n"
[19] "plKroga"   "plR10"     "podatki"
[22] "podatkiNA" "podatkiNAomit" "x"
[25] "x1"        "x2"        "x3"

```

### 1.5.3 Branje in pisanje datotek drugih programov

R vsebuje tudi paketek (foreign), ki omogoča branje in pisanje datotek drugih statističnih programov. Tako med drugim bere podatke iz formata sledečih programov:

- SPSS
- Stata

- SAS
- Weka
- Octave
- Minitab

Zaradi razširjenosti SPSS-a v družboslovju v tem učbeniku največkrat uporabljam funkcijo `read.spss`, ki omogoča branje SPSS-ovih datotek, obstajajo pa tudi podobne funkcije za ostale popularne statistične programe (glejte zgoraj).

```
> #naložimo paketek za branje podatkov iz "tujih" zapisov
> library(foreign)
> #preberemo podatke
> ego<-read.spss("egoomr00_f2.sav",to.data.frame=TRUE,
  use.value.labels = FALSE,use.missings=TRUE)
> ego[1:3,1:9] #izpišemo prve tri enote in prvih 9 spremenljivk
  EG0ID Q1A Q2A Q3A Q4A DQ1A DQ2A DQ3A DQ4A
1  1001  5  NA  5  NA  4  NA  4  NA
2  1003  5  6  6  6  5  6  6  6
3  1004  6  3  5  5  6  NA  5  5
```

Najpomembnejši argumenti funkcije `read.spss` so:

`to.data.frame` Če ga nastavimo na `TRUE`, bo rezultat funkcije podatkovni okvir (kar ponavadi želimo), sicer pa seznam (privzeta vrednost).

`use.value.labels` Ali naj **R** upošteva opise vrednosti ("value labels") iz SPSS-ove datoteke. Če ga nastavimo na `TRUE`, potem bodo vse spremenljivke, ki imajo nastavljene opise vrednosti, obravnavane kot faktorji, torej kot nominalne spremenljivke. Ne glede na vrednost argumenta se opisi vrednosti shranijo v "attribute" z imenom "value labels". Do njih lahko dostopamo preko ukaza `attr(x, "value.labels")`, kjer je `x` spremenljivka (npr: `ego$E_SPOL`), za katero želimo prebrati opise. Če smo uporabili `use.value.labels=FALSE`, je za spremembo spremenljivke v faktor uporabna funkcija `makeFactorLabels` iz datoteke "UcbenikR-funkcije.R".

`max.value.labels` Argument je uporaben le, kadar je `use.value.labels=TRUE`. V tem primeru lahko tudi nastavimo, koliko različnih vrednosti ima lahko spremenljivka (v podatkih), da se še uporabijo opisi – da se spremenljivka obravnava kot nominalna spremenljivka.

`use.missings` Ali naj se upoštevajo kode za manjkajoče vrednosti. Če je `TRUE`, se vse SPSS-ove "uporabniške manjkajoče vrednosti" pretvorijo v `NA`. **Opozorilo:** Ta pretvorba ne deluje vedno v redu, zato je dosti varneje, če uporabniške manjkajoče vrednosti pretvorimo v sistemske manjkajoče vrednost v SPSS-u.

V tem učbeniku beremo vse podatke iz SPSS-ovega formata. Razlog za izbor tega formata je predvsem v razširjenosti uporabe tega formata in tudi samega programa SPSS na Fakulteti za družbene vede Univerze v Ljubljani, pa tudi v obstoju odprtokodnega programa PSPP (<http://www.gnu.org/software/pspp/>), ki omogoča branje in pisanje podatkov v tem formatu. Seveda pa je moč podatke iz ADP prenesti tudi v drugih formatih in jih prebrati v R. Večina ukazov v primerih v tem učbeniku bo delovala neodvisno od izvirnega zapisa, vendar pa mora uporabnik v tem primeru sam poskrbeti za dostop do dolgih imen spremenljivk ter imen vrednosti spremenljivk. Funkcija `makeFactorLabels` denimo ne bo več delovala. Vendar pa njena uporaba v nekaterih primerih niti ni potrebna, na primer pri branju iz Statinega (dta) formata.

V nadaljevanju sledi še primer branja podatkov is Statinega formata (dta). Prebrali bomo enake podatke, kot so uporabljeni v uvodnem primeru oziroma zgornjem primeru za branje s funkcijo `read.spss`, le da smo te podatke iz ADP-ja prenesli v Statinem formatu. Tokrat bomo uporabili funkcijo `read.dta`, ki je za uporabo še preprostejša, saj običajno zelen rezultat dobimo že s privzetimi vrednostmi argumentov. Tako izpostavimo le argument `convert.factors`, ki nadzoruje, ali naj se številske spremenljivke z opisi vrednosti shranijo kot faktorji. Privzeta vrednost `TRUE` kot faktorje shrani tiste spremenljivke, ki imajo definirane opise za vse vrednosti. Pri vrednosti `FALSE` se nobena spremenljivka ne shrani kot faktor (kot pri `read.spss` pri vrednosti `use.value.labels=FALSE`), pri vrednosti `NA` pa se vse spremenljivke, ki imajo za vsaj kakšno vrednost definiran opis, shranijo kot faktorji (kot pri `read.spss` pri vrednosti `use.value.labels=TRUE`). Funkcija `read.dta` vrne podatkovni okvir. Dolga imena spremenljivk so shranjena v atributu `var.labels` vrnjenega podatkovnega okvirja, opisi vrednosti spremenljivk pa v atributu `label.table`.

```
> #naložimo paketek za branje podatkov iz "tujih" zapisov
> library(foreign) #ni potrebno, ker bi moral biti že naložen
> #preberemo podatke s privzetimi nastavitvami,
> #ki so običajno primerne
> egoStata<-read.dta("egoomr00_f2.dta")
> egoStata[1:3,1:9]
  EGOID          Q1A          Q2A
1  1001 precej zadovoljen-a      <NA>
2  1003 precej zadovoljen-a zelo zadovoljen-a
3  1004 zelo zadovoljen-a malo nezadovoljen-a
          Q3A          Q4A
1 precej zadovoljen-a      <NA>
2 zelo zadovoljen-a zelo zadovoljen-a
3 precej zadovoljen-a precej zadovoljen-a
          DQ1A          DQ2A          DQ3A
```

```

1 malo zadovoljen-a          <NA> malo zadovoljen-a
2 precej zadovoljen-a zelo zadovoljen-a zelo zadovoljen-a
3 zelo zadovoljen-a          <NA> precej zadovoljen-a
      DQ4A
1          <NA>
2 zelo zadovoljen-a
3 precej zadovoljen-a
> #izpišemo prve enote in prvih 9 spremenljivk
>
> #preberemo ponovno - tokrat brez pretvorbe v faktorje
> egoStata2<-read.dta("egoomr00_f2.dta", convert.factors=FALSE)
> egoStata2[1:3,1:9]
      EGOID Q1A Q2A Q3A Q4A DQ1A DQ2A DQ3A DQ4A
1  1001   5  NA   5  NA   4   NA   4   NA
2  1003   5   6   6   6   5   6   6   6
3  1004   6   3   5   5   6  NA   5   5
> #zopet izpišemo prve tri enote in prvih 9 spremenljivk
>
> attributes(egoStata2)$var.labels[1:3]
[1] "ID ega"
[2] "zadovoljstvo z materialno oporo-1.merjenje"
[3] "zadovoljstvo z informacijsko oporo-1.merjenje"
> attributes(egoStata2)$label.table[1:3]
$Q1A
      zelo nezadovoljen-a precej nezadovoljen-a
                        1                        2
      malo nezadovoljen-a malo zadovoljen-a
                        3                        4
      precej zadovoljen-a zelo zadovoljen-a
                        5                        6

$Q2A
      zelo nezadovoljen-a precej nezadovoljen-a
                        1                        2
      malo nezadovoljen-a malo zadovoljen-a
                        3                        4
      precej zadovoljen-a zelo zadovoljen-a
                        5                        6

$Q3A
      zelo nezadovoljen-a precej nezadovoljen-a
                        1                        2
      malo nezadovoljen-a malo zadovoljen-a

```

	3	4
precej zadovoljen-a		zelo zadovoljen-a
	5	6

Za pisanje datotek, ki jih želimo uvoziti v SPSS, bi sicer glede na opis funkcije lahko uporabili funkcijo `write.foreign` z argumentom `package = 'SPSS'`. Ta funkcija generira dve **tekstovni** datoteki, podatkovno in datoteko s sintakso. Vendar pa SPSS pri branju žal javi napako. Tudi če bi postopek deloval, je lažje zapisati datoteko v formatu programa Stata s funkcijo `write.dta`, saj lahko ta format uvozi tudi SPSS.

```
> x1<-rnorm(n=10,mean=176,sd=7)
> x2<-rnorm(n=10,mean=100,sd=15)
> x3<-sample(c("a","b","c"),size=10,replace=TRUE)
> podatki<-data.frame(x1,x2,x3)
> library(foreign) #ni potrebno, ker bi moral biti že naložen
> write.dta(dataframe=podatki, file="podatki.dta")
```

## 1.5.4 Vaje

### Vaja 1

Ustvarite podatkovno datoteko "csv" (recimo v Excel-u). V datoteko vpišite vsaj 10 enot in vsaj 4 spremenljivke (vsaj eno numerično in eno nominalno, opisno). Datoteko si oglejte tudi z urejevalnikom besedila. Datoteko preberite z R-jem.

### Vaja 2

Poljubne podatke shranite v tekstovno datoteko in jo potem pogledajte v urejevalniku besedila in programu za delo s preglednicami. Lahko izberete tudi podatke, ki so dostopni v R-ju preko funkcije `data` (napišite `data()` za seznam podatkovij).

### Vaja 3

Uvozite podatke iz poljubne SPSS-ove datoteke. Poskrbite, da boste dobili podatkovni okvir in da bodo uporabniške manjkajoče vrednosti opredeljene kot manjkajoče.

## Vaja 4

Podatke iz Vaje 2 (ali kakšne druge) izvozite tako, da jih boste lahko odprli z SPSS-om. Med podatki naj bo vsaj ena spremenljivka tipa *factor*. Podatke odprite z SPSS-om. So pri spremenljivki tipa *factor* (v **R**-ju) vidni opisi vrednosti v SPSS-u?

## 1.6 Risanje

**R** ima zelo dobro podprto grafiko oziroma risanje grafov. Za vpogled v to, kaj vse je mogoče narisati z **R**-jem, in tudi za ideje, kako kaj narediti, sta zelo koristni strani "R Graphical Manual" (<http://bm2.genes.nig.ac.jp/RGM2/> in "R graph gallery" <http://addictedtor.free.fr/graphiques/>.

### 1.6.1 Visokonivojske funkcije za risanje

Visokonivojske funkcije za risanje so tiste, ki ustvarijo nov graf. Kasneje bomo spoznali tudi nizkonivojske, ki le dodajo elemente na obstoječi graf. Ločnica med visoko- in nizko-nivojskimi funkcijami sicer ni ostra, saj se nekatere funkcije lahko uporabljajo v obeh vlogah.

Osnovna funkcija za risanje v **R**-ju je *plot*. Delovanje funkcije se razlikuje glede na tip objekta, ki ji ga posredujemo kot argument. Funkcija namreč poskuša pripraviti graf, ki je primeren vhodnemu argumentu. Tu ne bomo mogli pregledati vseh možnosti, zato navajam le nekaj primerov.

```
> #uporabimo podatke iz "izborESSn30.txt"
> izbor<-read.delim(file="izborESSn30.txt")
> sapply(izbor,class) #ponovimo, kakšne tipe podatkov imamo
      spol izobrazba      kraj      placa
"factor" "integer" "factor" "integer"
> plot(izbor$placa)
> #na y osi je plača, na x pa indeksi (zaporedne številke enot)
> plot(izbor$placa,type="o")
> plot(izbor$spol) #strukturni stolpci, ker je spol nominalka
> plot(placa~spol,data=izbor) #škatlasti grafikon (z zavihki)
> plot(spol~placa,data=izbor) #porazdelitev po razredih
> plot(kraj~spol,data=izbor)
> plot(placa~izobrazba,data=izbor)
> #pod data navedemo podatkovni okvir, kjer se nahajajo sprem.
> #razsevni grafikon - odvisnost plače od izobrazbe
> plot(placa~izobrazba,data=izbor)
```

```

> plot(izbor) #razsevni grafikon za vsak par spremenljivk
> lmFit<-lm(placa~izobrazba, data=izbor) #linearna regresija
> plot(lmFit)
> plot(placa~izobrazba,data=izbor,pch=as.numeric(spol),
      col=spol, cex=2, main = "Razsevni grafion",
      xlab="Izobrazba v letih", ylab = "Plača v 1000 sit")

```

Kot smo videli, funkcija `plot` vedno skuša narisati nekaj primernega, tudi takrat, ko smo kot argument uporabili rezultat linearne regresije. Ta funkcija in večina funkcij za risanje sprejme več argumentov, s katerimi lahko prilagajamo risanje (glejte predvsem daljši primer na strani 59). Nekaj najpogostejših:

`type` Tip grafikona. Relevantno pri razsevnih grafikonih – točke, črta, točke povezane s črto ...

`data` Iz katerega podatkovnega okvirja (ali seznama) naj uporabi podatke. Predvsem se uporablja, kadar je osnovni argument formula.

`main` Glavni naslov.

`xlab`, `ylab` Ime osi x in y.

`pch` Simbol, ki naj se uporablja za risanje točk. Lahko ima samo eno vrednost ali pa je vrednost vektor, ki za vsako točko poda simbol.

`col` Barva (ponavadi točke). Podobno kot zgoraj.

`cex` Velikost točke. Podobno kot zgoraj.

`xlim`, `ylim` Meje risanja na osi x in y (še nismo uporabili, bomo kasneje).

Obstaja še veliko dodatnih argumentov, s katerimi je mogoče po želji prilagoditi prikaz. Za pregled teh argumentov si pogledjte pomoč za funkciji `plot.default` in `par`.

Predvsem argumente slednje je treba pogosto nastaviti s klicem funkcije `par` pred klicem izbrane funkcije. Tak zelo uporaben argument je na primer argument `mfrow` oziroma `mfcol`, ki nam omogoča, da na eno sliko narišemo več grafov. Primer je na sliki 1.9 (koda sledi v naslednjem primeru).

Vseeno pa privzeti graf ni vedno tisti, ki ga želimo. Pogledjmo si še dodatne funkcije za risanje grafov:

`hist` Histogram.

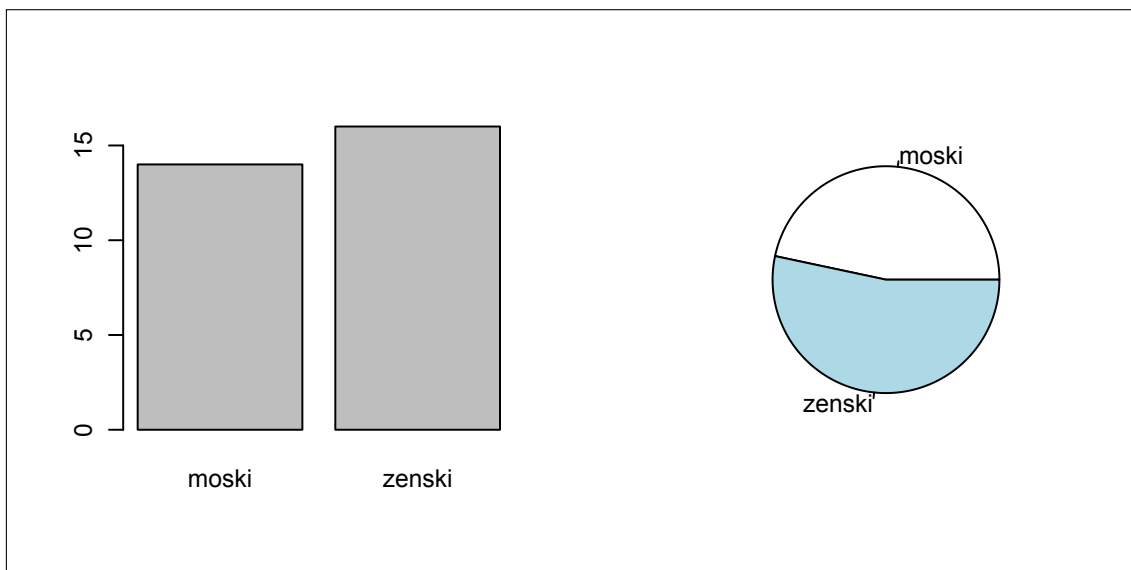
`barplot` Stolpci. **Pozor:** Za klasične strukturne stolpce mora biti argument tabela (le-to dobimo s funkcijo `table`).

`pie` Strukturni krog. Enako kot zgoraj.

`boxplot` Škatlasti grafikon (z zavihki).

`curve` Risanje krivulj.

Slika 1.9: Strukturna stolpca in krog na eni sliki



```

> curve(dnorm(x),xlim=c(-4,4))
> #standardna normalna porazdelitev
>
> hist(izbor$placa) #histogram - frekvence
> hist(izbor$placa, freq=FALSE) #frekvence - gostota
> boxplot(izbor$placa) #ena "skupina"
> boxplot(placa~spol, data=izbor) #dve "skupini"
> par(mfrow=c(2,1)) #spodnja 2 grafa želimo imeti na eni sliki
> barplot(table(izbor$spol))
> pie(table(izbor$spol))
> par(mfrow=c(1,1)) #povrnemo originalne nastavitve

```

## 1.6.2 Nizkonivojske funkcije za risanje

Nizkonivojske funkcije za risanje ne ustvarijo novega grafa, ampak le dodajo elemente na obstoječi graf. Nekaj najpomembnejših:

`points`, `lines` Dodata točke, črte ali točke, povezane s črto.

`text` Doda tekst.

`mtext` Doda tekst na obrobe/osi.

`axis` Doda osi.

`title` Doda naslov.

`legend` Doda legendo.

`expression` Uporabljam ga za vnašanje matematičnih formul ali simbolov na slike.<sup>5</sup>

`bquote` Uporablja se za vnašanje matematičnih formul, ki vključujejo izračunane količine, na slike (podobno kot `expression`).

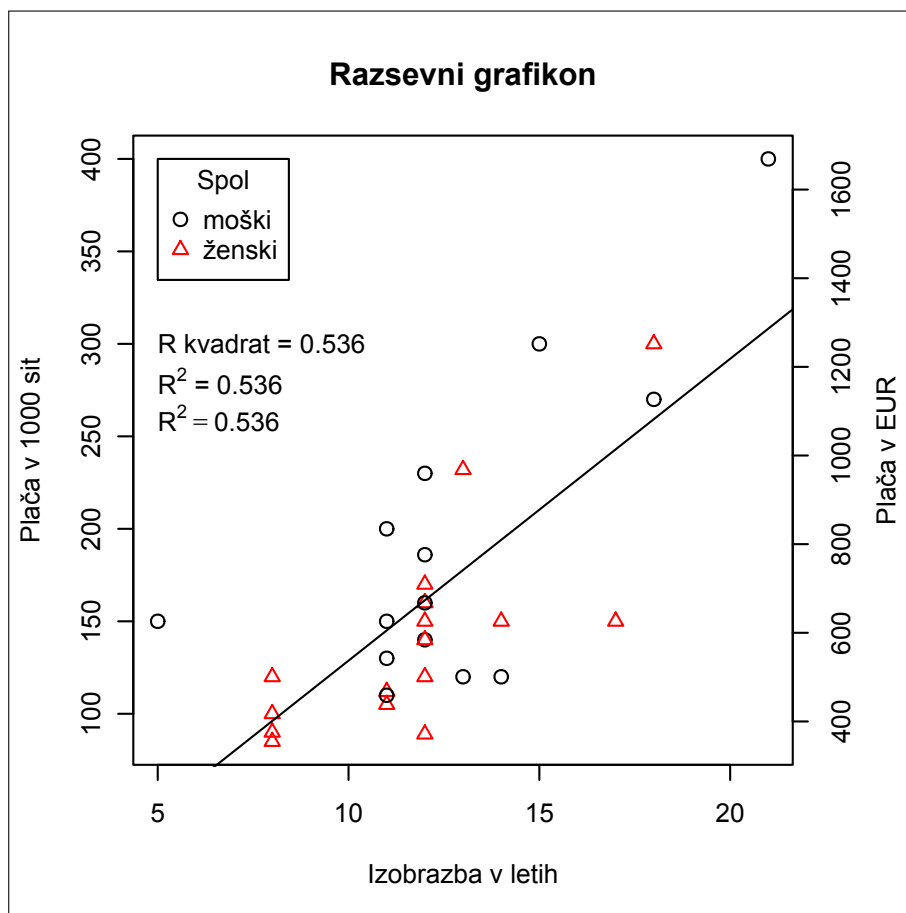
Posebej je treba razložiti uporabo `expression`. Včasih želimo na sliki izpisati tudi matematične formule ali znake. Le-te lahko uporabimo takorekoč povsod, kjer lahko pišemo besedilo (na primer naslovi, legende, oznake osi, oznake na oseh ...). V tem primeru namesto besedila določenemu argumentu (na primer `main`, `text`, `labels` ... kot vrednost podamo `expression`). Slabost tega načina pa je, da ne omogoča "mešanja" izračunih količin (recimo izračunano vrednost  $R^2$  v naslednjem primeru) in matematičnih oznak. Način pisanja za obe možnosti je opisan v pomoči pod geslom `plotmath` (za dostop do pomoči torej uporabimo `?plotmath`), je pa precej podoben L<sup>A</sup>T<sub>E</sub>X-u. Primer slike z veliko dodatnimi elementi je na sliki 1.10.

```
> lmFit<-lm(placa~izobrazba, data=izbor) #linearna regresija
> mar.old<-par("mar")
> par(mar=c(5,4,4,4)+0.1)
> plot(placa~izobrazba, data=izbor, pch=as.numeric(spol),
      col=spol, cex=1, main = "Razsevni grafikon",
      xlab="Izobrazba v letih", ylab = "Plača v 1000 sit")
> abline(lmFit)
> axis(side=4, at=(2:8)*2*100*239.64/1000,
      labels = (2:8)*2*100)
> mtext(text="Plača v EUR", side=4, line=2.5)
> par(mar=mar.old) #grafične parametre nastavimo nazaj
> #na staro vrednost
> legend(x=5, y = 400, legend = c("moški", "ženski"),
      title="Spol", pch=1:2, col=1:2, xjust=0, yjust=1)
> text(x=5, y = 300, adj=0, labels=
      paste("R kvadrat =", round(summary(lmFit)$r.squared, 3)))
> text(x=5, y = 280, adj=0,
      labels=expression(paste(R^2, " = ", 0.536)))
> text(x=5, y = 260, adj=0,
      labels=bquote(R^2 == .(round(summary(lmFit)$r.squared, 3))))
```

Poleg tega je mogoče veliko visokonivojskih funkcij uporabljati tudi za dodajanje na obstoječi graf (na primer `curve`, `hist` ...), tako da jim dodamo argument `add=TRUE`.

<sup>5</sup> Glavni rezultat funkcije je sicer, da ustvari R-jev izraz, ki pa se ne izvede (oziroma evalvira, izračuna).

Slika 1.10: Slika z veliko dodanimi elementi



```
> hist(izbor$placa, freq=FALSE) #frekvence - gostota
> curve(dnorm(x,mean=mean(izbor$placa), sd=sd(izbor$placa)),
  add=TRUE) #vrišemo normalno krivuljo
```

### 1.6.3 Shranjevanje slik

Slike, narisane z **R**-jem, je zelo enostavno shraniti v več formatih. To je možno storiti na več načinov:

1. Preko menija, ki se odpre s klikom desnega miškinega gumba kjerkoli na sliki. Preko tega menija je mogoče:
  - (a) Shraniti slike v datoteko formata Windows metafile (WMF/EMF) ali PostScript (PS/EPS).

- (b) Kopirati sliko v odložišče v obliki bitne slike ("bitmap") ali v formatu Windows metafile. To lahko potem s "prilepi" oziroma "paste" prilepite v večino programov (na primer *Word* ...).
2. Preko funkcije `savePlot` je mogoče shraniti sliko iz trenutno aktivnega grafičnega okna v enem izmed naslednjih formatov: Windows metafile, PNG, JPEG, BMP (Windows bitmap format), TIFF, PostScript in PDF. Tudi s pomočjo te funkcije je mogoče shraniti sliko v odložišče.
3. Tako da "rišemo" neposredno v datoteko oziroma na "napravo" (device), ki predstavlja datoteko. Postopek je sestavljen iz sledečih korakov:
  - (a) Odpremo izbrano napravo. Za seznam različnih naprav (tipov datotek) uporabite ukaz `?Devices`. Možnosti in argumenti, ki so dovoljeni pri klicu funkcij, ki odpirajo naprave, se razlikujejo glede na vrsto naprave/datoteke.
  - (b) Rišemo na enak način, kot če bi risali v odprto okno.
  - (c) Napravo zapremo s funkcijo `dev.off()`.

Ta (tretji) način omogoča največjo fleksibilnost pri določanju lastnosti končne slike (na primer kvalitete pri slikah jpg).

Recimo, da želimo narisati porazdelitve vseh spremenljivk v podatkih *izbor*, vsako na svoj graf. Te grafe želimo shraniti v datoteke tipa png. To lahko naredimo s spodnjo kodo.

```
> plot.numeric<-hist
> #S tem nastavimo histogram za privzeti prikaz numeričnih
> #spremenljivk. Tako početje je NEVARNO, ker spreminjamo
> #privzeto obanašanje funkcije plot.
> png(filename = "Slika%d.png",width = 1200, height = 1200,
      res=200) #namesto %d se bo v ime datoteke zapisala
>           #zaporedna številka slike
> #nekaj parametrov smo nastavili po svoje
> for(i in names(izbor)) plot(izbor[[i]],main=i,xlab=i)
> dev.off()
pdf
  2
> rm(plot.numeric) #"Odstranimo" nevarno nastavitvev
```

Pojasnila:

- V imenu slike smo uporabili "%d". To nam omogoča, da naredimo več slik, ne da bi za vsako posebej klicali funkcij `png` in `dev.off`. Png namreč omogoča le eno sliko na datoteko.

- Ker je png nevektorski format, širino in višino podamo v točkah (piksljih). Velikost slike določimo z argumentom `res`, kjer določimo število točk na inč. Pri vektorskih formatih (na primer pdf), širino in višino podamo v inčih.
- Na začetku smo uporabili ukaz `plot.numeric<-hist`. S tem smo nastavili histogram za privzeti prikaz numeričnih spremenljivk. Tako početje je **NEVARNO**, saj bi nam povzročalo probleme pri risanju razsevnih grafikonov. Zato smo na koncu tudi uporabili `rm(plot.numeric)`. Ker ta funkcija prej ni obstajala, smo s tem povrnili prejšnje stanje.

Narišimo še isto v datoteko pdf. Ker pdf podpira več strani, lahko vse 4 grafe narišemo v eno datoteko.

```
> plot.numeric<-hist
> pdf(file = "StiriSlike.pdf", width = 6, height = 6)
> #nekaj parametrov smo nastavili po svoje
> for(i in names(izbor)) plot(izbor[[i]],main=i,xlab=i)
> dev.off()
pdf
  2
> rm(plot.numeric)
```

*Opomba:* Pri formatih pdf, eps in ps lahko velikost "papirja" določamo neodvisno od velikosti slike z argumentom `papersize`. Privzeta vrednost je sicer, da je velikost papirja enaka velikosti slike.

## 1.6.4 Vaje

### Vaja 1

Narišite histogram za poljubno spremenljivko. Vrišite tudi normalno krivuljo. Glavni naslov in imena osi nastavite ročno (ne uporabite privzetih). Stolpci naj bodo modre barve.

*Namig:* Normalno krivuljo je veliko lažje vrisati, če pri histogramu rišete gostoto in ne frekvenc.

### Vaja 2

Na eno sliko narišite 4 histograme (v 2 vrstici s po 2 stolpcema) za poljubno spremenljivko. Na vsakem histogramu uporabite drugo število razredov.

### Vaja 3

Narišite razsevni grafikon za poljubni dve intervalni spremenljivki. Izberite vsaj eno izmed sledečih možnosti:

- Velikost točke naj se razlikuje glede na vrednost neke tretje razmernostne spremenljivke.
- Oblika točke naj se razlikuje glede na vrednost neke nominalne spremenljivke.

## 1.7 Priprava dokumentov z rezultati iz R-ja

R podpira več načinov za pripravo dokumentov z rezultati analiz. Najzmožljivejši temeljijo na konceptu *ponovljive znanosti* (ang. "reproducible research"). Pri pripravi tega učbenika je uporabljen sistem Sweave (Leisch 2002) (<http://www.stat.uni-muenchen.de/~leisch/Sweave/>), ki je del osnovne R-jeve distribucije. Sistem omogoča, da v enem dokumentu združimo tekst in R-jevo kodo ter rezultate iz R-ja (izpis, tabele, grafi ...). Osnovni sistem je bil razvit za L<sup>A</sup>T<sub>E</sub>X, preko paketa *R2HTML* pa omogoča tudi uporabo HTML-ja. Zelo podoben sistem podpira tudi paketek *knitr* (<http://yihui.name/knitr/>), ki ima dodatno prednost, da lahko za urejanje besedila uporablja tudi druge sisteme (ne samo L<sup>A</sup>T<sub>E</sub>X), na primer HTML in Markdown. Predvsem Markdown je zelo enostaven za uporabo, omogoča pa tudi izdelavo končnega dokumenta v različnih formatih (Word, PDF ali HTML).

Sweave sistem je mogoče uporabiti tudi preko dokumentov formata "Open Document Format" oziroma "ODF", ki jih na primer uporabljata *OpenOffice.org* in *Libre Office*. To omogoča paketek *odfWeave* (<http://cran.r-project.org/web/packages/odfWeave/index.html>). Prednost tega je, da ne zahteva znanja L<sup>A</sup>T<sub>E</sub>X-a ali HTML-ja.

Za uporabnike *Microsoft (Office) Word*a pa sta najbolj zanimiva paketeka *R2wd* in *rtf*. Prednost prvega je, da omogoča vstavljanje neposredno v odprt Wordov dokument, drugega pa, da ne potrebuje dodatnih programov (npr. Word) in nima z njimi povezanih omejitev, sej neposredno ustvari RTF (rich text format) dokument. Tako je mogoče neposredno ustvariti poročilo, ali pa vanj le izvoziti tabele, ki jih potem lahko zelo enostavno prenesemo v Word.

Poglejmo najprej paketek *R2wd*, ki deluje le v povezavi z 32-bitnim *Wordom*. V primeru, da je namen dodajati rezultate iz R-ja v *Word* in ne avtomatsko generiranje poročil, je verjetno najuporabnejša funkcija *wdTable*. Najbolj problematično pri izvozu rezultatov v Word je namreč dodajanje tabel, saj je osnovni R-jev izpis tekstovni, kar v poročilih ni videti lepo. Ravno funkcija *wdTable* pa omogoča vstavljanje lepo oblikovanih tabel v *Word*, žal pa je precej počasna. Preden pa lahko karkoli pišemo v dokument, ga moramo najprej odpreti s funkcijo *wdGet*.

S spodnjo kodo bomo v *Word* nalepili tri tabele.

```

> library(R2wd)
> wdGet() #odpremo nov ali obstoječi (kot argument podamo ime
>       #in po potrebi pot do njega) dokument
> #vmes moramo počakati, da se dokument odpre
> #najbolj varno je, da prej nimamo odprtega nobenega dokumenta
>
> wdTable(izbor) #izvozimo tabelo (podatke) v dokument
> #tabela se pojavi na mestu, kjer je kazalnik (oz. kurzor)
>
> wdTable(table(Kraj=izbor$kraj,Spol=izbor$spol))
> #kontingenčna tabela
>
> regKoeff<-summary(lm(placa~izobrazba+spol,data=izbor))$coef
> regKoeff<-format(as.data.frame(regKoeff),digits=c(4,4,4,1),
>                 scien=FALSE) #da bo lepši format
> wdTable(format(regKoeff))
> #izvozimo tabelo regresijskih koeficientov
>
> X<-matrix(1:15,ncol=3)
> #wdTable(X)
> #tole bi javilo napako
> #tabela ali matrika mora imeti definirana imena stolpcev in
> #vrstic
> colnames(X)<-c("a","b","c")
> rownames(X)<-1:5
> wdTable(X)
> ?"R2wd-package"
> #več pomoči o paketku

```

Kot lahko vidite na podlagi pomoči za paketek (`?R2wd-package`), paketek omogoča generiranje celotnih *Word*ovih dokumentov, av tem učbeniku je predstavljen le najbolj problematičen del. Za vstavljanje slik sicer lahko uporabimo funkcijo `wd-Plot`, lahko pa tudi sliko iz R-ja enostavno skopiramo v odložišče ali shranimo v datoteko (glejte podpodoglavje 1.6.3) in od tam pilepimo v *Word*.

Poglejmo sedaj še uporabo paketka `rtf`. Tudi tu je najbolj ključna funkcija za izvoz tabel funkcija `addTable`, predstavljena pa je še uporaba funkcij `addHeader` za ustvarjanje naslovov in `addText` za vstavljanje navadnega besedila. Pred oblikovanjem dokumenta ga moramo najprej ustvariti s funkcijo `RTF` (ki kot argument sprejme ime datoteke) in na koncu zapreti s funkcijo `done`.

```
> library(rtf)
> rtfDoc<-RTF("test.rtf") #kreiramo novo datoteko in shranimo
>                               #dobljeni objekt
>
> #dodamo naslov in podnaslov
> addHeader(rtfDoc,title="Test",subtitle="Osnovni podatki")
> addTable(rtfDoc,izbor)
> addText(rtfDoc,"Tole so osnovni podatki.\n") #dodamo tekst
> addHeader(rtfDoc,title=NULL,subtitle="Kontingenčna tabela")
> addTable(rtfDoc,table(Kraj=izbor$kraj,Spol=izbor$spol))
> #kontingenčna tabela
>
>
> regKoeff<-summary(lm(placa~izobrazba+spol,data=izbor))$coef
> regKoeff<-format(as.data.frame(regKoeff),digits=c(4,4,4,1),
  scien=FALSE) #da bo lepši format
> addHeader(rtfDoc,title=NULL,subtitle="Regresijski koeficienti")
> addTable(rtfDoc,format(regKoeff))
> #izvozimo tabelo regresijskih koeficientov
>
>
> X<-matrix(1:15,ncol=3)
> addHeader(rtfDoc,title=NULL,
  subtitle="Matrika brez imen stolpcev in vrstic")
> #dodajo se privzeta imena stolpcev
> addTable(rtfDoc,X)
> addText(rtfDoc,"Dodana so bila privzeta imena stolpcev.\n")
> #dodamo tekst
>
> colnames(X)<-c("a","b","c")
> rownames(X)<-1:5
> addHeader(rtfDoc,title=NULL,
  subtitle="Matrika z imeni stolpcev in vrstic")
> addTable(rtfDoc,X, row.names=TRUE)
> #če želimo izpisati imena vrstic
> done(rtfDoc) #zapremo datoteko
> ?"rtf-package"
> #več pomoči o paketku
```

## 1.8 Rešitve vaj

### 1.8.1 Podatkovne strukture

#### Vaja 1

```

> x<-c(10, 23, 43, 43, 32, 12)
> povX<-sum(x)/length(x)
> povX
[1] 27.16667
> varX<-sum((x - povX)^2)/length(x)
> varX
[1] 177.8056
> sdX<-varX^(1/2) #ali sqrt(varX)
> sdX
[1] 13.33437
> res<-c(povprecje=povX,varianca=varX,sd=sdX)
> res
povprecje  varianca      sd
27.16667 177.80556 13.33437

```

#### Vaja 2

```

> y<-c(10, 23, 43, 43, 32, 12)
> x1<-c(1.5, 2.3, 5.4, 4.5, 4.2, 1.0)
> x2<-c(210, 183, 186, 164, 175, 200)
> X<-cbind(1,x1,x2)
> b<-as.vector(solve(t(X) %*% X) %*% t(X) %*% y)
> b
[1] 52.819111 6.240474 -0.243166
> names(b)<-colnames(X)
> b
           x1      x2
52.819111 6.240474 -0.243166
> yNap <- as.vector(X %*% b)
> e <- y - yNap
> cbind(y=y,napoved=yNap,rezidual=e)
      y  napoved  rezidual
[1,] 10 11.11496 -1.1149560
[2,] 23 22.67282 0.3271824
[3,] 43 41.28879 1.7112125

```

```

[4,] 43 41.02201 1.9779861
[5,] 32 36.47505 -4.4750455
[6,] 12 10.42638 1.5736205
> res<-list(b=b, napovedi=yNap ,rezidual=e)
> res
$b
      x1      x2
52.819111 6.240474 -0.243166

$napovedi
[1] 11.11496 22.67282 41.28879 41.02201 36.47505 10.42638

$rezidual
[1] -1.1149560 0.3271824 1.7112125 1.9779861 -4.4750455
[6] 1.5736205

```

## 1.8.2 Funkcije in programiranje

### Vaja 1

Rešitev Vaje 1 iz podpodpoglavja [1.3.8](#) (podpoglavje [Podatkovne strukture](#)) v obliki funkcije:

```

> #definicija funkcije
> opisStat<-function(x){
  povX<-sum(x)/length(x)
  varX<-sum((x - povX)^2)/length(x)
  sdX<-varX^(1/2) #ali sqrt(varX)
  res<-c(povprecje=povX,varianca=varX,sd=sdX)
  return(res)
}
> x<-c(10, 23, 43, 43, 32, 12)
> opisStat(x)
povprecje  varianca      sd
27.16667 177.80556 13.33437

```

Rešitev Vaje 2 iz podpodpoglavja [1.3.8](#) (podpoglavje [Podatkovne strukture](#)) v obliki funkcije:

```

> regresija<-function(y,X){
  X<-cbind(1,X)

```

```

        b<-as.vector(solve(t(X) %*% X) %*% t(X) %*% y)
        names(b)<-colnames(X)
        yNap <- as.vector(X %*% b)
        e <- y - yNap
        res<-list(b=b, napovedi=yNap ,rezidual=e)
        return(res)
    }
> y<-c(10, 23, 43, 43, 32, 12)
> x1<-c(1.5, 2.3, 5.4, 4.5, 4.2, 1.0)
> x2<-c(210, 183, 186, 164, 175, 200)
> X<-cbind(x1,x2)
> regresija(y,X)
$b
              x1          x2
52.819111  6.240474 -0.243166

$napovedi
[1] 11.11496 22.67282 41.28879 41.02201 36.47505 10.42638

$rezidual
[1] -1.1149560  0.3271824  1.7112125  1.9779861 -4.4750455
[6]  1.5736205

```

## Vaja 2

```

> vsota<-function(x){
    vsota<-0
    for(i in x){
        vsota <- vsota + i
    }
    return(vsota)
}
> x<-c(10, 23, 43, 43, 32, 12)
> vsota(x)
[1] 163
> sum(x) #enak rezultat
[1] 163

```

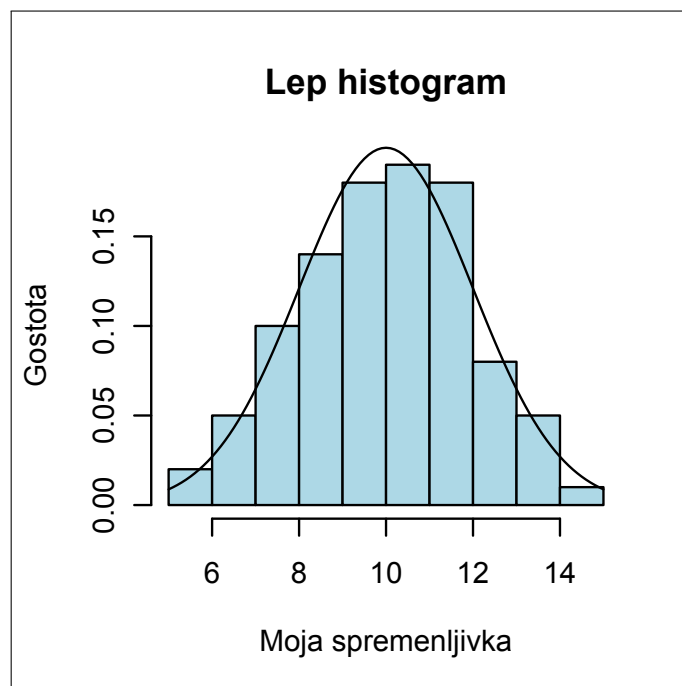
## Vaja 3

```
> #funkcija s for zanko
> fakFor<-function(n){
  res<-1
  for(i in seq_len(n)){
    res<-res*i
  }
  return(res)
}
> #funkcija z while zanko
> fakWhile<-function(n){
  res<-1
  while(n>1){
    res<-res*n
    n<-n-1
  }
  return(res)
}
> #funkcija z rekurzivno funkcijo
> fakRek<-function(n){
  if(n<=1){
    return(1)
  }else{
    return(n*fakRek(n-1))
  }
}
> #vse funkcije dajo seveda enak rezultat
> fakFor(10)
[1] 3628800
> fakWhile(10)
[1] 3628800
> fakRek(10)
[1] 3628800
> factorial(10) #tudi R-jeva vgrajena funkcija
[1] 3628800
```

#### Vaja 4

```
> med<-function(x){
  x<-sort(x)
  n<-length(x)

  if(n %% 2 ==0){
```

Slika 1.11: Rešitev vaje 1 ([Risanje](#))

```

        return((x[n/2]+x[n/2+1])/2)
      }else{
        return(x[ceiling(n/2)])
      }
    }
> x<-c(10, 23, 43, 43, 32, 12)
> med(x)
[1] 27.5

```

### 1.8.3 Risanje

#### Vaja 1

```

> set.seed(2010)
> x<-rnorm(100,mean=10,sd=2)
> hist(x,col="lightblue",main="Lep histogram",
      xlab="Moja spremenljivka",freq=FALSE,ylab="Gostota")
> curve(dnorm(x,mean=10,sd=2),add=TRUE,xpd=NA)

```

Rezultat je prikazan na sliki [1.11](#).

## Vaja 2

```
> set.seed(2010)
> x<-rnorm(1000,mean=10,sd=2)
> par(mfrow=c(2,2))
> hist(x,main="približno 5 razredov",freq=FALSE,
      ylab="Gostota", br=5)
> hist(x,main="približno 10 razredov",freq=FALSE,
      ylab="Gostota",br=10)
> hist(x,main="približno 30 razredov",freq=FALSE,
      ylab="Gostota",br=30)
> hist(x,main="približno 50 razredov",freq=FALSE,
      ylab="Gostota",br=50)
> par(mfrow=c(1,1))
```

Rezultat je prikazan na sliki 1.12.

## Vaja 3

```
> set.seed(2010)
> x<- rnorm(100,mean=0,sd=2)
> y<- x^2 +rnorm(100,sd=3)
> z<-abs(abs(y*x)^(1/4)+rnorm(100,sd=0.5))
> k<-as.numeric(cut(x+rnorm(100,sd=1),breaks=5))
> plot(y~x,pch=k,cex=z)
```

Rezultat je prikazan na sliki 1.13.

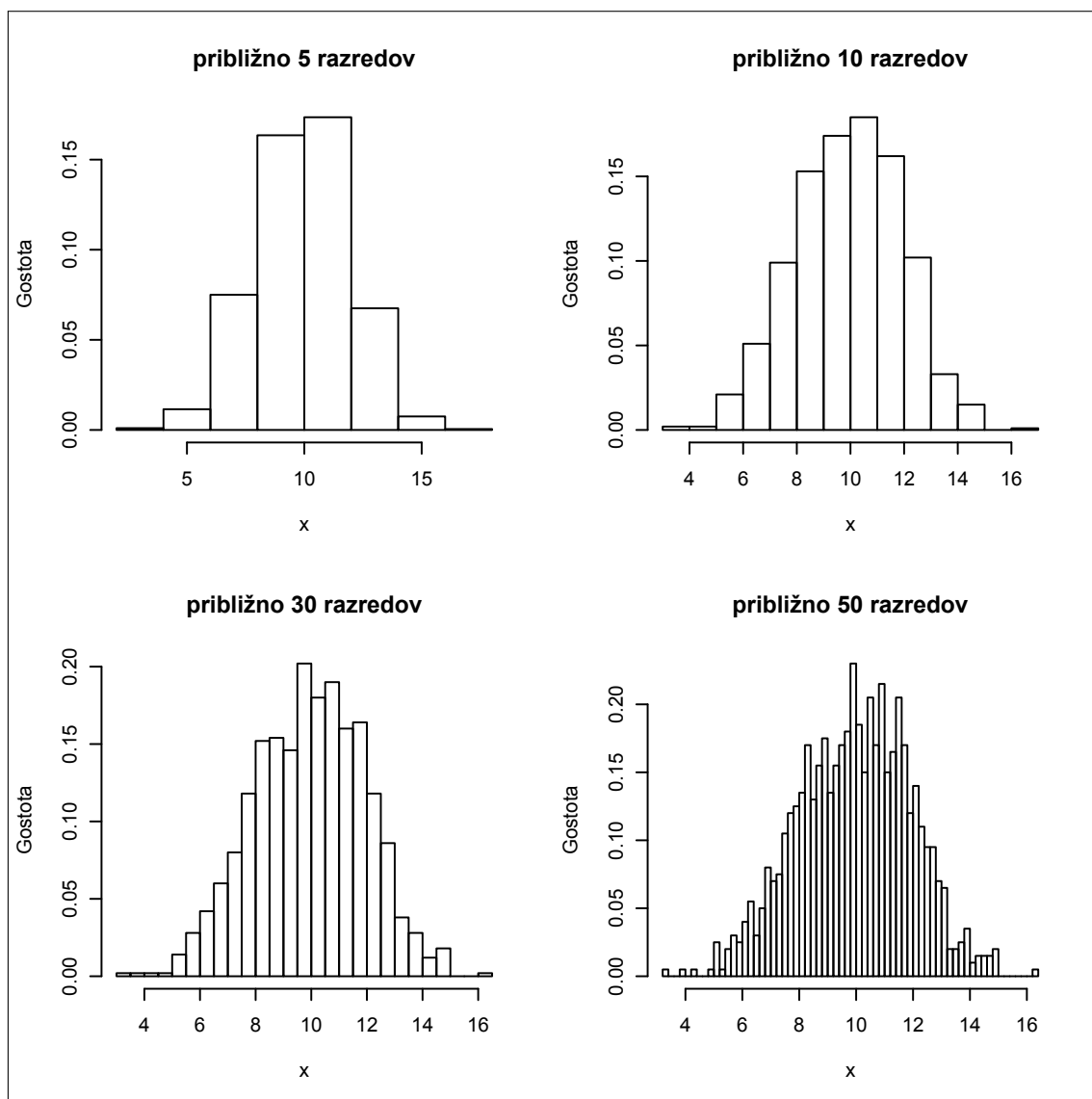
## 1.9 Viri za poglobljanje znanja

### 1.9.1 Spletni viri

R je odprtokodni program, zato je veliko virov na voljo tudi na spletu. Tukaj naštevamo nekaj najpomembnejših:

**R-project** Glavna spletna stran za R, ki vsebuje veliko uporabnih vsebin, predvsem pa omogoča tudi prenos programskega paketa R URL: <http://www.r-project.org/>

**R-project – "uradni" priročniki** Uradni priročniki za R - precej tehnično in osnovno. URL: <http://cran.r-project.org/manuals.html>

Slika 1.12: Rešitev vaje 2 ([Risanje](#))

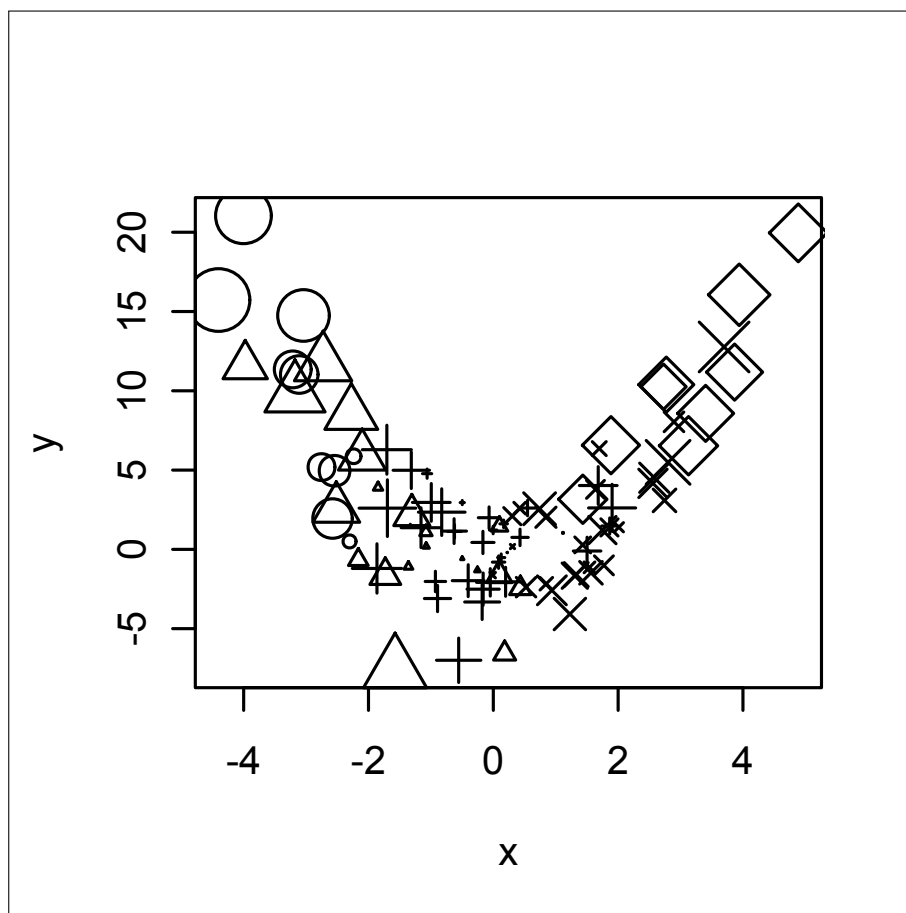
**R-project – ostali priročniki** Poleg uradnih priročnikov je na voljo tudi mnogo drugih priročnikov in podobnega gradiva, ki so ga prispevali uporabniki **R**-ja. URL: <http://cran.r-project.org/other-docs.html>

**Rseek** Spletni iskalnik po vsebinah, povezanih z **R**-jem, vključno s seznamami za elektronsko pošto za pomoč uporabnikom. URL: <http://www.rseek.org/>

**Quick-R** Odličen hitri vodič po **R**-ju za uporabnike drugih statističnih paketov (SPSS, SAS, Stata ...), se pravi za tiste, ki statistiko že znajo. URL: <http://www.statmethods.net/>

**R Graph Gallery** Galerija grafov, narejenih z **R**-jem, vključno z uporabljenimi kodo. URL: <http://addictedtor.free.fr/graphiques/>

Slika 1.13: Rešitev vaje 3 (Risanje)



**RStudio** Integrirano razvojno okolje za **R**. Združuje urejevalnik besedila z označevanjem in samodokončanjem **R**-jeve kode, konzolo, pomoč, okno za grafične prikaze in še veliko več. URL: <http://rstudio.org/>

**Spletna stran prof. dr. Andreja Blejca o R-ju** Vsebuje veliko koristnih vsebin povezanih z **R**-jem, vključno s priročnikom *Introduction to R*. URL: <http://ablejec.nib.si/R/>

## 1.9.2 Knjižni viri

### Splošne knjige o R-ju

Knjige v tem poglavju so splošni priročniki za programski paket **R**. Pokrivajo vsa ali večino podpoglavij tega poglavja, večinoma pa tudi področja iz ostalih dveh poglavjih.

- Zuur, Alain F., Elena N. Ieno in Erik H.W.G. Meesters. 2009. *A Beginner's Guide to R*. New York: Springer.<sup>6</sup>  
Zelo nezahtevna, samo osnove **R**-ja (skoraj) brez statistike.
- Muenchen, Robert A. 2011. *R for SAS and SPSS users*. New York: Springer.<sup>7</sup>  
Še posebej primerna za tiste, ki že poznajo SPSS ali SAS.
- Verzani, John. 2005. *Using R for introductory statistics*. Boca Raton: Chapman & Hall/CRC.<sup>8</sup>
- Dalgaard, Peter. 2002. *Introductory statistics with R*. New York: Springer.<sup>9</sup>

### Knjige o programiranju

- Matloff, Norman. 2011. *The Art of R Programming: A Tour of Statistical Software Design*. San Francisco: No Starch Press.  
Manj zahtevna.
- Venables, William N. in Brian D. Ripley. 2000. *S programming*. New York: Springer.  
Bolj zahtevna.

### Knjige o risanju

Praktično vse splošne knjige o **R**-ju vsebujejo tudi poglavje o risanju, ki ponavadi zadostuje za osnovne grafe. Spodaj naštetih knjigi pa bolj poglobljeno predstavljata različne sisteme za risanje v **R**-ju (prva) ali še posebej podrobno določen način risanja (druga).

- Murrell, Paul. 2011. *R graphics*. Boca Raton: CRC Press.  
Različni sistemi za risanje.
- Wickham, Hadley. 2009. *ggplot2: Elegant Graphics for Data Analysis*. Dordrecht; New York: Springer. URL <http://public.eblib.com/EBLPublic/PublicView.do?ptiID=511468>.<sup>10</sup>  
Sistem za risanje, ki temelji na Wilsonovi (2005) grafični slovnici.

<sup>6</sup> Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>7</sup> Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>8</sup> Starejša in manj obsežna različica knjige je dostopna tudi na <http://www.math.csi.cuny.edu/Statistics/R/simpleR/printable/simpleR.pdf>

<sup>9</sup> Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>10</sup> Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

## Priprava dokumentov/poročil

- Xie, Yihui. 2013. *Dynamic report generation with r and knitr*. Boca Raton: Chapman & Hall Crc.
- Gandrud, Christopher. 2013. *Reproducible research with R and RStudio*. Boca Raton: Chapman & Hall/CRC.

## 1.10 Vprašanja za ponavljanje

1. **R** je vektorski jezik. Kaj to pomeni?
2. Katera podatkovna struktura v **R**-ju je najprimernejša za shranjevanje podatkovij – podatkov o več spremenljivkah na več enotah?
3. Kako v **R**-ju prikličemo pomoč za neko funkcijo in kako iščemo po pomoči?
4. Kateri izmed sledečih znakov so dovoljeni v imenih objektov v **R**-ju: ? , . ! | \_ - \ + ?
5. Na kakšen način v **R**-ju predstavimo nominalne spremenljivke?
6. Ali **R** omogoča matrično računanje?
7. Kako **R** razširimo z dodatnimi funkcijami?
8. Katere vrste zank poznamo v **R**-ju in v čem se razlikujejo?
9. Ali lahko z **R**-jem beremo podatkovne datoteke iz ostalih popularnih statističnih paketov (SPSS, Stata ...)? Če da, kateri paketek potrebujemo za to?
10. Katere funkcije lahko uporabimo za branje in pisanje tekstovnih podatkovnih datotek?
11. Od česa je odvisno, kakšne vrste graf nariše funkcija *plot*?
12. Kakšna je razlika med visokonivojskimi in nizkonivojskimi funkcijami za risanje?
13. Kaj pomeni notacija  $y \sim x$  in zakaj se uporablja v funkcijah za risanje?
14. Kakšen graf nariše funkcija *plot*, če kot argument podamo nominalno spremenljivko, in kakšen, če kot argumenta podamo dve intervalni spremenljivki?
15. Na kakšne načine lahko shranjujemo slike?
16. Kako za neko funkcijo poiščemo njene argumente?
17. V katerih formatih (vrstah datotek) lahko izdelujemo poročila z **R**-jem?



## 2. poglavje

# Univariatna in bivariatna statistika

Drugo poglavje obravnava kar veliko univariatnih in bivariatnih statističnih metod, tako s področja osnovne opisne statistike kot tudi inferenčne statistike. Na začetku poglavja so najprej predstavljeni podatki, ki jih uporabljamo v tem poglavju, skupaj z navodili za njihovo pridobitev iz Arhiva družboslovnih podatkov in branje z **R**-jem. Sledi predstavitev osnovnih opisnih statistik, predvsem mer srednjih vrednosti in variabilnosti ter mer za obliko porazdelitve (asimetrija, sploščenost). Od tu naprej učbenik obravnava tudi metode inferenčne statistike, najprej pri pregledu metod za preverjanje domnev o srednjih vrednostih (o eni srednji vrednosti in o razliki med srednjimi vrednostmi za neodvisne in odvisne vzorce) in računanje ustreznih intervalov zaupanja, kjer poleg parametričnih metod obravnavamo tudi neparametrične. To je tudi skoraj edini del učbenika, kjer tudi teoretično predstavimo obravnavane metode. Sledi preverjanje domnev o deležih ter računanje ustreznih intervalov zaupanja. Na koncu poglavja obravnavamo še metode za merjenje o povezanosti dveh spremenljivk glede na merske lestvice spremenljivk ter za preverjanje domnev o povezanosti.

### 2.1 Uporabljeni podatki

Za prikaz predstavljenih metod bomo uporabili podatke iz Evropske družboslovne raziskave (<http://www.europeansocialsurvey.org/>) za Slovenijo za leto 2004. Uporabljena datoteka je bila pridobljena iz Arhiva družboslovnih podatkov (Toš in drugi 2004). Opis raziskave in povezave do datoteke so dostopni na tem spletnem naslovu: <http://www.adp.fdv.uni-lj.si/opisi/sjm042/>.

Iz Arhiva družboslovnih podatkov sem prenesel podatke v SPSS-formatu (".sav"). Najprej preberemo podatke iz SPSS-ove datoteke ter uvozimo funkcije iz datoteke "UcbenikR-funkcije.R".

```

> #naložimo podatke
> library(foreign)
> data<-read.spss(file="sjm042_f1.sav",
  to.data.frame = TRUE, use.value.labels = TRUE,
  max.value.labels=5, use.missings=TRUE)
> #naložimo tudi dodatne funkcije
> source("UcbenikR-funkcije.R")

```

Nato preverimo število enot in spremenljivk ter katere spremenljivke so v podatkih z uporabo funkcije `names`. Dolga imena spremenljivk iz SPSS-a (labels) so shranjena v atributu `variable.labels`.

```

> dim(data) #število enot in spremenljivk
> names(data)
> attributes(data)$variable.labels

```

Zaradi dolžine izpisa ga tu ne navajamo.

## 2.2 Osnovne statistike

**R** že v osnovni različici (brez dodatnih paketkov) vsebuje funkcije za praktično vse osnovne statistike. Navedimo nekaj najosnovnejših:

`mean` aritmetična sredina,

`median` mediana,

`sd` in `var` standardni odklon in varianca (vzorčna –  $(n - 1)$  v imenovalcu),

`min` in `max` minimum in maksimum,

`range` razpon (vrne minimum in maksimum kot vektor),

`quantile` kvantili.

Vse zgoraj naštetih funkcije kot argument sprejmejo vektor (spremenljivko). V privzeti obliki ne dovoljujejo manjkajočih vrednosti (*NA*), saj v tem primeru tudi vrnejo *NA*. Vendar pa je mogoče nastaviti, da manjkajoče vrednosti ignorirajo s parametrom `na.rm=TRUE`.

Recimo, da nas še posebej zanima spremenljivka G91 – Bruto plača. Tako bomo na njej izračunali osnovne statistike.

```

> #G91 - bruto plača
> #izračunajmo osnovne statistike zanjo
> sum(!is.na(data$G91))

```

```

[1] 325
> #preštejemo število enot z veljavnimi vrednostmi pri G91
> #imamo jih samo 325
>
> hist(data$G91)#narišemo histogram
> #pri vseh sledečih funkcijah dodamo argument "na.rm=TRUE"
> #brez njega bi zaradi manjkajočih vrednosti dobili
> #rezultat NA
> range(data$G91, na.rm=TRUE) #preverimo razpon
[1] 53 760
> mean(data$G91, na.rm=TRUE) #aritmetična sredina
[1] 225.7108
> sd(data$G91, na.rm=TRUE) #standardni odklon
[1] 120.953
> median(data$G91, na.rm=TRUE) #mediana
[1] 200
> quantile(data$G91, probs=c(0.25,0.5,0.75), na.rm=TRUE)
25% 50% 75%
140 200 280
> #izračunamo kvartile

```

Zelo koristna je tudi funkcija *summary*, ki izračuna povzetek, ki je primeren glede na tip objekta/spremenljivke. Funkcija se lahko uporabi na različnih podatkovnih strukturah (tudi na primer na celotnem podatkovnem okvirju).

Funkcija *summary* sicer naredi kar dober povzetek, vendar pa pri številskih spremenljivkah pogrešam vsaj izračun standardnega odklona. Lahko bi sicer naredili svojo funkcijo, lahko pa uporabimo funkcijo *describe* iz paketa *psych*, ki poleg tega izračuna tudi mere asimetrije in sploščenosti. Tudi funkcijo *describe* lahko uporabimo le na eni spremenljivki ali na celotnem podatkovnem okvirju. Je pa ta funkcija primerna le za intervalne in razmernostne spremenljivke. Izračun naredi sicer tudi za ostale, a ni smiselno in zato jih v rezultatih označi z "\*".

V paketu *psych* najdemo tudi funkciji za koeficient asimetrije *skew* in sploščenosti *kurtosi* in še veliko drugih uporabnih funkcij (ki se uporabljajo predvsem v psihologiji in družboslovju).

Poglejmo torej najprej uporabo *summary* na eni sami spremenljivki.

```

> #"povzetek" za številsko spremenljivko
> summary(data$G91)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NAs
  53.0  140.0   200.0   225.7  280.0   760.0   1117
> #in še za nominalno - spol
> summary(data$O1F2)

```

```

moski zenski   NAs
  648   762   32
> #summary upošteva tip podatka
>
> #uporabili bomo tudi spremenljivko F6,
> #ki jo moramo prej spremeniti "nazaj" v faktor
> data$F6<-makeFactorLabels(data$F6)
> summary(data$F6)
      nedokoncana OS          dokoncana OS
      69                    368
      2-3 letna poklicna          gimnazija
      347                    434
      2-letna visja visoka sola ali fakulteta
      72                    138
      magisterij, doktorat          NAs
      11                    3

```

Funkcijo *summary* lahko uporabimo tudi na več spremenljivkah ali celo na celotnem podatkovju (podatkovnem okvirju), a jo bomo tu zaradi varčevanja s prostorom uporabili le na nekaj izbranih spremenljivkah.

```

> izbraneSprem<-c("01F3","01F2","F5","F6","F7","G91","G92")
> #izbor shranimo, ker ga bomo uporabljali večkrat
> #da se lažje spomnimo, kaj merijo
> attributes(data)$variable.labels[izbraneSprem]
      01F3
      "leto rojstva"
      01F2
      "spol"
      F5
      "Kje živite?"
      F6
      "izobrazba"
      F7
      "leta šolanja"
      G91
      "Kolikšna je običajno vaša bruto plača?"
      G92
      "Kolikšna je običajno vaša neto plača?"
> summary(data[izbraneSprem])
      01F3          01F2          F5
Min.   :1909   moski :648   veliko mesto:132
1st Qu.:1944   zenski:762   predmestje  :204

```

```

Median :1960   NAs  : 32   manjse mesto:326
Mean    :1959                               vas          :626
3rd Qu.:1975                               kmetija      :148
Max.    :1989                               NAs          : 6
NAs     :16

                                F6           F7
gimnazija           :434   Min.      : 1.00
dokoncana OS        :368   1st Qu.: 8.00
2-3 letna poklicna :347   Median  :11.00
visoka sola ali fakulteta:138 Mean     :11.27
2-letna visja      : 72   3rd Qu.:13.00
(Other)            : 80   Max.    :23.00
NAs                : 3   NAs     :11

      G91           G92
Min.   : 53.0   Min.   : 7.0
1st Qu.:140.0   1st Qu.: 72.5
Median :200.0   Median :120.0
Mean   :225.7   Mean   :124.8
3rd Qu.:280.0   3rd Qu.:170.0
Max.   :760.0   Max.   :600.0
NAs    :1117    NAs    :1046

```

Je s temi podatki kaj narobe?

Minimum za spremenljivko G92 je "7". To mora biti napaka, saj nihče ne more imeti bruto plače samo 7000 sit. Verjetno gre za napako pri vnosu. "7" je pogosta koda za manjkajočo vrednost, pri tej spremenljivki pa bi se morala uporabljati koda "7777777". Preglejmo, ali je takih vrednosti več. Recimo tako, da izpišemo frekvenčno tabelo za to spremenljivko. Nato popravimo podatke tako, da vrednosti, ki niso mogoče (še posebej, če predvidevamo, da gre za kode manjkajočih vrednosti), spremenimo v *NA*. Nato ponovno izpišemo povzetek in se prepričamo, da je sedaj vse v redu.

```

> table(data$G92)
 7  8 40 60 70 71 73 75 78 80 82 85 86 87 88
76 13  1  4  4  1  1  5  2  8  2  8  1  1  1
89 90 92 95 100 104 105 107 108 110 112 114 115 117 120
 1 11  1  5 19  1  2  1  1 10  1  1  4  2 19
122 125 130 132 135 137 138 140 145 150 153 154 155 160 165
 2  3 13  1  1  1  1 12  1 26  1  1  1 12  2
170 180 185 186 190 191 200 205 210 215 220 230 232 240 250
 13  8  2  1  4  1 19  1  5  2  6  6  1  2 11
260 270 280 300 320 336 360 370 380 400 480 600
 2  6  2 10  1  1  1  1  1  2  1  1

```

```
> data$G92[data$G92 %in% c(7,8)]<-NA
> #vrednosti 7 in 8 spremenimo v NA
> summary(data$G92)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NAs
  40.0  104.5   150.0   158.9  200.0   600.0   1135
```

### Opozorilo!

Kot kaže zgornji primer, moramo vedno pred analizo opraviti osnovni pregled podatkov, da ugotovimo, ali so v njih prisotne kakšne napake!

Za računanje opisnih statistik za številske spremenljivke pa je, kot rečeno, bolj primerna funkcija *describe* iz paketa *psych*. Uporabimo še to.

```
> #če paketek psych ni nameščen, ga je treba namestiti z
> #install.packages("psych")
> library(psych)
> describe(data$G91)
 vars   n   mean      sd median trimmed   mad min max
1     1 325 225.71 120.95   200  208.85 103.78  53 760
 range skew kurtosis   se
1   707 1.46    2.52 6.71
> #lahko jo izračunamo tudi za vse podatke, a tu jo zaradi
> #varčevanja s prostorom le za nekaj izbranih spremenljivk
> describe(data[izbraneSprem])
      vars   n   mean      sd median trimmed   mad min
01F3     1 1426 1958.64 18.96   1960 1959.17 23.72 1909
01F2*    2 1410    1.54  0.50     2    1.55  0.00    1
F5*      3 1436    3.32  1.12     4    3.39  1.48    1
F6*      4 1439    3.37  1.34     3    3.26  1.48    1
F7       5 1431   11.27  3.41    11   11.20  2.97    1
G91      6  325  225.71 120.95   200  208.85 103.78  53
G92      7  307  158.93  74.13   150  149.75  74.13  40
 max range  skew kurtosis   se
01F3 1989    80 -0.20   -0.97 0.50
01F2*  2     1 -0.16   -1.98 0.01
F5*    5     4 -0.60   -0.48 0.03
F6*    7     6  0.46   -0.30 0.04
F7    23    22  0.13    0.26 0.09
G91   760   707  1.46    2.52 6.71
G92   600   560  1.65    4.81 4.23
> skew(data$G91)
[1] 1.455042
```

```
> kurtosi(data$G91)
[1] 2.516902
```

Spremenljivka "Bruto plača" (G91) ima aritmetično sredino 226 tisoč sit (enote so 1000 sit), standardni odklon pa 121 tisoč sit. Njena porazdelitev je precej asimetrična v desno in koničasta, kar je pri porazdelitvi plače sicer pričakovano.

Funkciji *summary* in *describe* smo že lahko neposredno uporabili na več spremenljivkah naenkrat. Pri funkcijah, ki se uporabljajo na vektorjih, pa upoštevamo, da lahko neko funkcijo uporabimo tudi na več spremenljivkah s pomočjo funkcij *apply*, *lapply* in *sapply*.

```
> #naredimo nov izbor s samo vsaj intervalnimi spremenljivkami
> #torej s samo intervalnimi in razmernostnimi spremenljivkami
> izbraneIntSprem<-c("O1F3", "F7", "G91", "G92")
> #izračunajmo aritmetično sredino
> apply(data[izbraneIntSprem], 2, mean, na.rm=TRUE)
      O1F3      F7      G91      G92
1958.63534  11.26625  225.71077  158.93485
> sapply(data[izbraneIntSprem], mean, na.rm=TRUE) #enako
      O1F3      F7      G91      G92
1958.63534  11.26625  225.71077  158.93485
> lapply(data[izbraneIntSprem], mean, na.rm=TRUE) #drugačen izpis
$O1F3
[1] 1958.635

$F7
[1] 11.26625

$G91
[1] 225.7108

$G92
[1] 158.9349
> #standardni odklon
> sapply(data[izbraneIntSprem], sd, na.rm=TRUE)
      O1F3      F7      G91      G92
18.961464  3.412601 120.952965  74.128283
> #razpon - funkcija lahko vrne tudi več vrednosti
> sapply(data[izbraneIntSprem], range, na.rm=TRUE)
      O1F3 F7 G91 G92
[1,] 1909  1  53  40
[2,] 1989 23 760 600
```

```

> #za večjo prilagodljivost lahko napišemo svojo funkcijo
> mojPovzetek<-function(x){c(
  "arit. sredina" = mean(x,na.rm=TRUE),
  "std. odklon" = sd(x,na.rm=TRUE),
  "št. veljavnih vrednosti" = sum(!is.na(x)))}
> sapply(data[izbraneIntSprem],mojPovzetek)
              01F3              F7              G91
arit. sredina      1958.63534      11.266247 225.7108
std. odklon        18.96146        3.412601 120.9530
št. veljavnih vrednosti 1426.00000 1431.000000 325.0000
              G92
arit. sredina      158.93485
std. odklon        74.12828
št. veljavnih vrednosti 307.00000

```

Če želimo statistike izračunati po skupinah, določenih z neko drugo spremenljivko, so koristne splošne funkcije, kot je denimo *aggregate*, ki omogočajo uporabo funkcij na skupinah. Natančna uporaba je razvidna iz primera (in pomoči). Funkcija kot enega od argumentov sprejme funkcijo, ki jo uporabi na vsaki skupini.

V paketu *psych* pa obstaja tudi funkcija *describeBy*, ki to omogoča neposredno.

```

> data$01F2<-factor(data$01F2) #odstranimo "prazne" kategorije
> #glede na eno spremenljivko
> aggregate(x=data[c("G91","G92")], by = list(data$01F2),
  FUN=mean, na.rm=TRUE) #rezultat je ena številka
Group.1      G91      G92
1  moski 237.5896 168.9868
2  zenski 209.9103 147.2517
> aggregate(x=data[c("G91","G92")], by = list(data$01F2),
  FUN=mojPovzetek) #več številka
Group.1 G91.arit. sredina G91.std. odklon
1  moski      237.5896      126.3618
2  zenski      209.9103      112.7086
G91.št. veljavnih vrednosti G92.arit. sredina
1              173.0000              168.98675
2              145.0000              147.25170
G92.std. odklon G92.št. veljavnih vrednosti
1      81.53142              151.00000
2      64.30224              147.00000
> #funkcija aggregate dela le na R 2.11 in novejših
> #na starejših verzijah R-ja javi napako
>

```

```

> #glede na dve spremenljivki
> aggregate(x=data[c("G91", "G92")],
  by = list(data$O1F2, data$F5),
  FUN=mean, na.rm=TRUE) #rezultat je ena številka
  Group.1      Group.2      G91      G92
1  moski veliko mesto 252.0000 188.7059
2  zenski veliko mesto 252.3333 183.0000
3  moski  predmestje 231.1250 159.3810
4  zenski  predmestje 209.2381 147.2000
5  moski  manjse mesto 288.6923 189.4186
6  zenski  manjse mesto 210.1143 161.2353
7  moski      vas 227.0870 157.4615
8  zenski      vas 202.8060 133.5000
9  moski  kmetija 188.8889 146.0556
10 zenski  kmetija 188.0000 104.2857
> aggregate(x=data[c("G91", "G92")],
  by = list(data$O1F2, data$F5),
  FUN=mojPovzetek) #več številok
  Group.1      Group.2 G91.arit. sredina G91.std. odklon
1  moski veliko mesto      252.00000      127.94650
2  zenski veliko mesto      252.33333      143.55172
3  moski  predmestje      231.12500      95.43917
4  zenski  predmestje      209.23810      99.89640
5  moski  manjse mesto      288.69231      147.34656
6  zenski  manjse mesto      210.11429      92.63679
7  moski      vas      227.08696      125.35992
8  zenski      vas      202.80597      120.44064
9  moski  kmetija      188.88889      98.64713
10 zenski  kmetija      188.00000      98.54441
  G91.št. veljavnih vrednosti G92.arit. sredina
1      14.00000      188.70588
2      15.00000      183.00000
3      24.00000      159.38095
4      21.00000      147.20000
5      39.00000      189.41860
6      35.00000      161.23529
7      69.00000      157.46154
8      67.00000      133.50000
9      27.00000      146.05556
10     7.00000      104.28571
  G92.std. odklon G92.št. veljavnih vrednosti
1      80.84148      17.00000
2      83.44260      19.00000

```

3	53.39052	21.00000
4	56.79202	25.00000
5	79.64362	43.00000
6	61.77675	34.00000
7	94.27670	52.00000
8	59.01313	62.00000
9	65.50929	18.00000
10	34.13070	7.00000

Za funkcijo `describeBy` je koda spodaj, izpis pa je zaradi varčevanja s prostorom izpuščen (seveda lahko kodo preizkusite sami).

```
> #z describeBy
> #glede na eno spremenljivko
> describeBy(x=data[c("G91", "G92")],
  group = data$01F2)
> #glede na dve spremenljivki
> describeBy(x=data[c("G91", "G92")],
  group = list(data$01F2, data$F5))
```

## 2.3 Preverjanje domnev o srednjih vrednostih in pripadajoči intervali zaupanja

V tem podpodpoglavju so obravnavane le metode za preverjanje domnev o srednji vrednosti in o razliki dveh srednjih vrednosti (na odvisnih in neodvisnih vzorcih). Metode za preverjanje domnev o več srednjih vrednostih so obravnavane v podpoglavju 3.2 ([Analiza variance \(ANOVA\)](#)). Poleg tega bomo pri parametričnih pristopih (t-testih) omenili tudi pripadajoči interval zaupanja.

Poleg parametričnega tukaj obravnavamo tudi neparametrični pristop. Nekaj besed o razliki med obema:

**parametrični testi** So "močnejši"<sup>11</sup>, a imajo običajno strožje predpostavke.

**neparametrični testi** Imajo manj predpostavk, a so "šibkejši". Če predpostavke parametričnih testov (približno) držijo, izberemo parametrične teste, sicer neparametrične (če njihove predpostavke držijo).

Neparametrične teste pogosteje uporabljamo na manjših vzorcih, saj ima tam kršenje predpostavk parametričnih testov hujše posledice.

<sup>11</sup>Močnejši test je tisti, ki pri izbrani stopnji tveganja zazna (statistično značilne) razlike pri manjši razliki med vzorčno statistiko in vrednostjo iz ničelne domneve.

**Opozorilo!**

Neparametrični testi niso samo "robustnejša verzija" parametričnih, ampak preverjajo **drugačne domneve**. Za primere, za katere tu ne podamo testa ali intervala zaupanja, je ponavadi priporočljivo uporabiti metode ponovnega vzorčenja.

V ta namen so uporabne predvsem sledeče funkcije:

`t.test` Za vse vrste t-testov.

`var.test` Test enakosti varianc. Za preverjanje predpostavk t-testa.

`fligner.test` Še en test enakosti varianc. Manj občutljiv na odstopanje od normalnosti. Za preverjanje predpostavk t-testa.

`leveneTest` Še en test enakosti varianc iz paketa `car`. Tudi ta je manj občutljiv na odstopanje od normalnosti. Z argumentom `center=mean` (kar ni privzeta možnost) je to test, ki ga uporablja SPSS. Za preverjanje predpostavk t-testa.

`wilcox.test` Wilxonovi testi za en ali dva vzorca. Test za dva neodvisna vzorca je znan tudi kot Mann-Whitneyjev test ali Wilxon-Mann-Whitneyjev test.

`binom.test` Uporaben za izvedbo testa predznaka (ang. sign test), ki se uporablja kot test mediane.

Preverjanje domnev uporabimo, če imamo oblikovano ničelno domnevo, ki jo želimo (običajno) zavrniti. Postopek temelji na tem, da preverimo, kako verjetno bi bilo, da bi na vzorcu dane velikosti dobili tako "ekstremno" vrednost testne statistike, kot smo jo dobili na našem vzorcu.

Intervale zaupanja podamo, kadar želimo podati oceno nekega parametra. Ker oceno računamo na podlagi vzorca, točkovna ocena (skoraj) zagotovo ni pravilna. Tako želimo izračunati nek interval, za katerega lahko z danim tveganjem (gotovostjo) trdimo, da vsebuje populacijski parameter. Če se na primer vzorčne ocene  $g$  parametra  $\gamma$  porazdeljujejo normalno s standardnim odklonom  $SE(g)$ , potem lahko interval zaupanja pri tveganju  $\alpha$  izračunamo kot:

$$P(g - z_{\alpha/2}SE(g) \leq \gamma \leq g + z_{\alpha/2}SE(g)) = 1 - \alpha$$

Bolj splošno (tudi če se vzorčne ocene ne porazdeljujejo normalno) vedno velja:

$$P(F^{-1}(p = \alpha/2, g, \dots) \leq \gamma \leq F^{-1}(p = 1 - \alpha/2, g, \dots)) = 1 - \alpha$$

$F^{-1}$  je kvantilna funkcija za porazdelitev vzorčnih ocen  $g$  oziroma inverzna funkcija porazdelitveni funkciji  $F$ . Tu velja omeniti, da **R** vsebuje kvantilne funkcije za večino v statistiki uporabljenih porazdelitev. Naj omenim samo najpomembnejše:

`qnorm` Normalna porazdelitev

`qt` t porazdelitev

`qf` F porazdelitev

`qbinom` Binomska porazdelitev

`qchisq`  $\chi^2$  porazdelitev

`qunif` Enakomerna porazdelitev

`qexp` Eksponentna porazdelitev

`q???` Še kar nekaj drugih porazdelitev. Ime funkcije se vedno začne s "q", sledi oznaka (ime ali okrajšava) za porazdelitev.

Izračun intervala zaupanja bomo podrobneje prikazali samo za aritmetično sredino, čeprav bi lahko na zelo podoben način izračunali tudi intervala za razliko dveh aritmetičnih sredin na neodvisnih in odvisnih vzorcih (razlika je predvsem pri izračunu standardne napake vzorčne ocene).

### 2.3.1 Preverjanje domneve o srednji vrednosti in pripadajoči interval zaupanja

Kaj je to *srednja vrednost*, je odvisno od testa. Pri parametričnih testih je to praviloma aritmetična sredina, drugje pa ni nujno. To velja tudi pri problemih in testih, ki jih obravnavamo v sledečih podpodglavjih.

T-test za en vzorec verjetno že dobro poznate, tako si pogledjmo malce podrobneje le neparametrična testa za preverjanje domneve o srednji vrednosti:

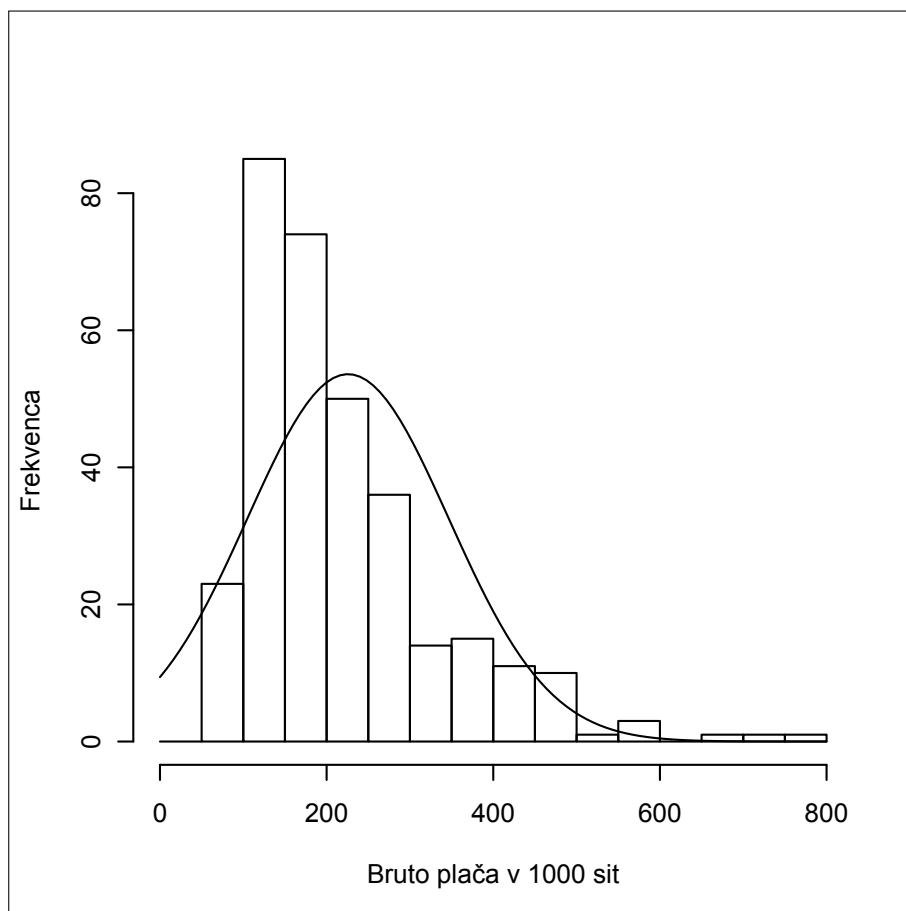
**Wilcoxonov test** oziroma Wilcoxonov test označenih rangov

- **Predpostavlja vsaj intervalno mersko lestvico.**
- Načeloma predpostavlja simetrično porazdelitev  $\Rightarrow$  potem se lahko uporablja kot test mediane.
- **V splošnem to torej ni test mediane in vsekakor ne "robusten" test o vrednosti aritmetične sredine.**
- Pogosto navajajo tudi, da testira, ali je porazdelitev simetrična glede na testno vrednost.
- Tehnično gledano preverja, ali je vsota rangov negativnih odklonov večja od pozitivnih.
- Uporablja se predvsem na majhnih vzorcih, ko je porazdelitev simetrična, a ne normalna.

**Test predznaka** Pogosto se uporablja tudi ime *test mediane*

- **Predpostavlja vsaj ordinalno mersko lestvico.** Zanima nas samo predznak odklona od testne vrednosti, torej katera vrednost je večja.

Slika 2.1: Porazdelitev bruto plače



- **Testira domnevo o vrednosti mediane** oziroma natančneje domnevo, da je število pozitivnih in negativnih odklonov enako.

Najprej preverimo domnevo o srednji vrednosti. V splošnem naj bo domneva:  $H_0$ : Srednja vrednost bruto plače je enaka 220 tisoč sit.

Poleg tega izračunajmo tudi 90-% interval zaupanja za povprečno plačo.

Porazdelitev bruto plače je prikazana na sliki 2.1.

```
> #najprej si oglejmo porazdelitev
> h<-hist(data$G91,br=(0:16)*50,xlab="Bruto plača v 1000 sit",
  main="",ylab="Frekvenca")
> curve(dnorm(x,mean=mean(data$G91,na.rm=TRUE),
  sd=sd(data$G91,na.rm=TRUE))*diff(h$breaks)[1]*
  sum(!is.na(data$G91)), add=TRUE)
> describe(data$G91)
```

### 90 2.3. Preverjanje domnev o srednjih vrednostih in pripadajoči intervali zaupanja

```
vars   n   mean    sd median trimmed   mad min max
1     1 325 225.71 120.95   200  208.85 103.78  53 760
range skew kurtosis  se
1     707 1.46     2.52 6.71
```

Iz slike 2.1 in opisnih statistik je razvidno, da porazdelitev ni niti približno normalna. Sedaj preverimo, ali lahko zavrնemo domnevo, da je povprečna plača enaka 220.000 sit, prav tako pa izračunajmo tudi 90-% interval zaupanja za povprečno bruto plačo.

```
> t.test(data$G91,mu=220,conf.level = 0.90)
      One Sample t-test

data:  data$G91
t = 0.85118, df = 324, p-value = 0.3953
alternative hypothesis: true mean is not equal to 220
90 percent confidence interval:
 214.6434 236.7782
sample estimates:
mean of x
 225.7108
```

Niti pri 10-% tveganju ne moremo trditi, da je povprečna bruto plača različna od 220 tisoč sit. Z 10-% tveganjem lahko trdimo, da je povprečna bruto plača med 214.64 in 236.78 tisoč sit.

```
> #interval zaupanja bi lahko izračunali tudi takole
> n<- sum(!is.na(data$G91)) #število veljavnih vrednosti
> alfa <- 0.1
> mean(data$G91,na.rm=TRUE) +
  qt(c(alfa/2,1-alfa/2),df=n-1)*sd(data$G91,na.rm=TRUE)/sqrt(n)
[1] 214.6434 236.7782
```

Wilcoxonov test se lahko uporablja za preverjanje domneve, ali je porazdelitev bruto plače simetrična okoli 220.000 sit. Vendar pa ta domneva nima ravno smisla, saj smo videli, da je porazdelitev izrazito nesimetrična. Tehnično gledano preverja domnevo, da je vsota rangov negativnih odklonov enaka vsoti rangov pozitivnih odklonov (pri računanju rangov se predznak ne upošteva).

```
> wilcox.test(data$G91,mu=220)
      Wilcoxon signed rank test with continuity correction

data:  data$G91
```

```
V = 21965, p-value = 0.1245
alternative hypothesis: true location is not equal to 220
```

Tudi z Wilcoxonovim testom ničelne domneve pri 10-% tveganju ne moremo zavrniti. Opravimo sedaj še test predznaka oziroma test mediane. Preverimo torej, ali je negativnih odklonov več kot pozitivnih (pri tem odklonov z vrednostjo 0 ne upoštevamo).

```
> nPoz<-sum(data$G91>220, na.rm = TRUE)
> #manjkajoče vrednosti ignoriramo
> nNeg<-sum(data$G91<220, na.rm = TRUE)
> binom.test(x=nPoz,n=nPoz+nNeg,p=0.5)
      Exact binomial test

data:  nPoz and nPoz + nNeg
number of successes = 124, number of trials = 312,
p-value = 0.0003462
alternative hypothesis: true probability of success is not equal
to 0.5
95 percent confidence interval:
 0.3427360 0.4541027
sample estimates:
probability of success
      0.3974359
> median(data$G92,na.rm=TRUE)
[1] 150
```

Ničelno domnevo, da je mediana plače enaka 220, lahko zavrnemo pri 0.03-% tveganju, kar niti ni presenetljivo, saj je vrednost mediane 200. `binom.test` lahko kličemo tudi takole:

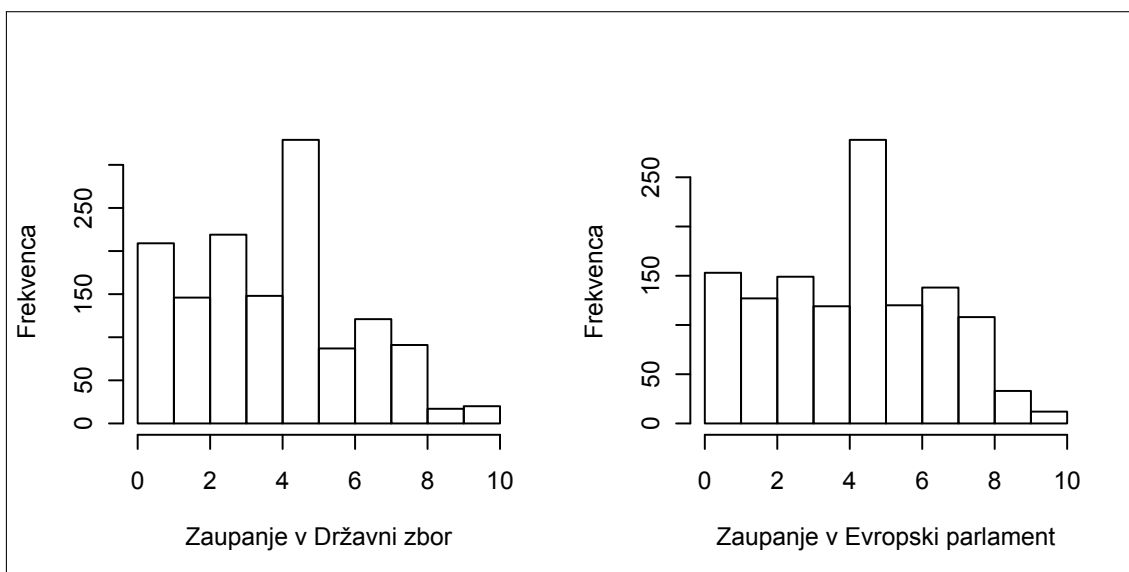
```
> binom.test(x=c(nPoz,nNeg),p=0.5)
```

### 2.3.2 Preverjanje domneve o razliki med srednjima vrednostma na odvisnih vzorcih in pripadajoči interval zaupanja

Vsi obravnavani testi za odvisne vzorce so enakovredni testom za en vzorec (o vrednosti srednje vrednosti), ki se izvedejo na razliki dveh spremenljivk. To velja upoštevati tudi pri interpretaciji rezultatov. Pri *t-testu* to sicer ni problematično, saj velja, da je aritmetična sredina razlike enaka razliki aritmetičnih sredin. Enako pa ne velja za mediano – mediana razlike **ni enaka** razliki median.

## 92 2.3. Preverjanje domnev o srednjih vrednostih in pripadajoči intervali zaupanja

Slika 2.2: Porazdelitev zaupanja v Državni zbor in Evropski parlament



### Opozorilo!

**Test predznaka** torej preverja domnevo o vrednosti mediane razlike in ne o vrednosti razlike median.

Primer domneve je lahko:  $H_0$ : Zaupanje v državni zbor je v "povprečju" enako zaupanju v evropski parlament.

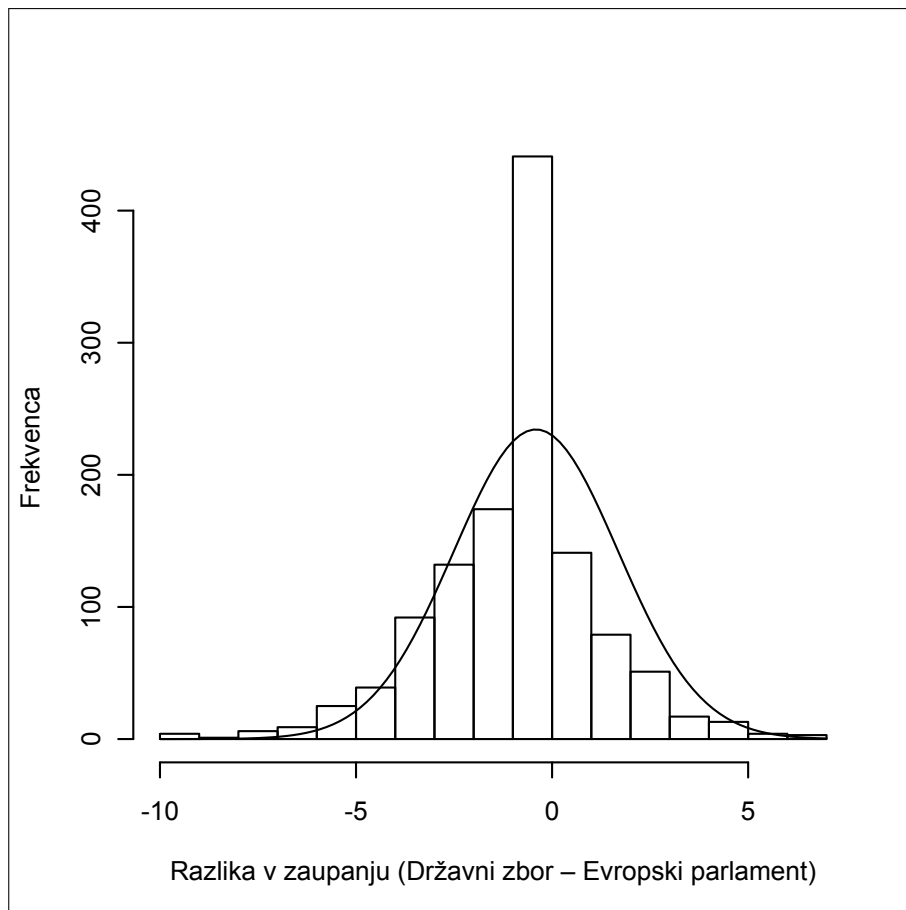
Poleg tega izračunajmo tudi 90-% interval zaupanja za razliko v zaupanju v oba parlamenta.

Poglejmo si najprej porazdelitve obeh spremenljivk (slika 2.2). Kot vemo, je ključna porazdelitev razlike obeh spremenljivk<sup>12</sup> (slika 2.3), saj so testi za odvisne vzorce enakovredni testom za en vzorec na razliki spremenljivk.

```
> #Pregledamo porazdelitve in opisne statistike
> #originalnih spremenljivk
> par(mfrow=c(1,2))
> hist(data$B4,xlab="Zaupanje v Državni zbor",main="",
      ylab="Frekvenca")
> hist(data$B9,xlab="Zaupanje v Evropski parlament",main="",
      ylab="Frekvenca")
> par(mfrow=c(1,1))
> describe(data[c("B4","B9")])
```

<sup>12</sup>Razliko izračunamo kot prva spremenljivka – druga spremenljivka.

Slika 2.3: Porazdelitev razlike med zaupanjem v Državni zbor in zaupanjem v Evropski parlament



```

vars      n mean  sd median trimmed mad min max range
B4       1 1387 4.13 2.38      4   4.10 1.48  0 10  10
B9       2 1247 4.53 2.41      5   4.57 2.97  0 10  10
  skew kurtosis  se
B4  0.11   -0.54 0.06
B9 -0.09   -0.70 0.07
> #razlike
> h<-hist(data$B4-data$B9,main="",ylab="Frekvenca",br=-10:7,
  xlab="Razlika v zaupanju (Državni zbor - Evropski parlament)")
> curve(dnorm(x,mean=mean(data$B4-data$B9,na.rm=TRUE),
  sd=sd(data$B4-data$B9,na.rm=TRUE))*diff(h$breaks)[1]*
  sum(!is.na(data$B4-data$B9)), add=TRUE)
> describe(data$B4-data$B9)
vars      n mean  sd median trimmed mad min max range
1         1 1231 -0.41 2.1      0  -0.37 1.48 -10  7  17

```

	skew	kurtosis	se
1	-0.31	1.82	0.06

Na slikah 2.2 in 2.3 ter zgornjem izpisu vidimo, da je porazdelitev obeh spremenljivk podobna, porazdelitev razlike pa sicer približno normalna z rahlo asimetrijo v levo in znatno koničavostjo.

Sedaj preverimo še (ničelno) domnevo, da je povprečno zaupanje v oba parlamenta enako, ter izračunajmo 90-% interval zaupanja.

```
> t.test(x=data$B4,y=data$B9, paired=TRUE, conf.level = 0.90)
      Paired t-test

data:  data$B4 and data$B9
t = -6.883, df = 1230, p-value = 9.317e-12
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 -0.5093517 -0.3127442
sample estimates:
mean of the differences
      -0.4110479
```

Ugotovimo lahko, da je razlika med obema povprečjema statistično značilna pri zanemarljivem tveganju. Z 10-% tveganjem lahko trdimo, da smo Slovenci leta 2004 v povprečju med  $-0.51$  in  $-0.31$  točke bolj zaupali Evropskemu parlamentu kot državnemu zboru.

Sedaj opravimo še neparametrične teste. Pri Wilcoxonovem testu je formalna domneva, da je vsota rangov negativnih vrednosti enaka vsoti rangov pozitivnih vrednosti (pri računanju rangov se predznak ne upošteva). Pri testu predznaka/mediane preverimo, ali je negativnih odklonov več kot pozitivnih (odklonov z vrednostjo 0 ne upoštevamo).

```
> wilcox.test(x=data$B4,y=data$B9, paired=TRUE)
      Wilcoxon signed rank test with continuity correction

data:  data$B4 and data$B9
V = 112990, p-value = 9.524e-12
alternative hypothesis: true location shift is not equal to 0
> nPoz<-sum((data$B4-data$B9)>0, na.rm = TRUE)
> #manjkajoče vrednosti ignoriramo
> nNeg<-sum((data$B4-data$B9)<0, na.rm = TRUE)
> binom.test(x=nPoz,n=nPoz+nNeg,p=0.5)
```

```

Exact binomial test

data:  nPoz and nPoz + nNeg
number of successes = 308, number of trials = 790,
p-value = 6.433e-10
alternative hypothesis: true probability of success is not equal
to 0.5
95 percent confidence interval:
 0.3556961 0.4248759
sample estimates:
probability of success
      0.3898734

```

Na podlagi obeh testov lahko ugotovimo, da je razlika med povprečjema statistično značilno različna od 0 pri zanemarljivem tveganju oziroma da srednji vrednosti nista enaki.

### 2.3.3 Preverjanje domneve o razliki med srednjima vrednostma na neodvisnih vzorcih in pripadajoči interval zaupanja

Edini neparametrični test, ki ga bomo obravnavali tu, je Mann-Whitneyjev test (oziroma Wilcoxonov test za neodvisna vzorca ali Wilcoxon-Mann-Whitneyjev test):

- **Predpostavlja vsaj ordinalno mersko lestvico.**
- **Testira, ali sta povprečna ranga enaka** oziroma ali je verjetnost, da ima enota iz 1. skupine večjo vrednost kot enota iz 2. skupine, enaka 0.5.
- Če sta obliki porazdelitev v obeh skupinah podobni, se lahko uporablja za test enakosti srednjih vrednosti.

Primer domneve je lahko:  $H_0$ : "Srednja vrednost" za bruto plačo je enaka pri moških in ženskah.

Poleg tega izračunajmo tudi 90-% interval zaupanja za razliko v bruto plači moških in žensk.

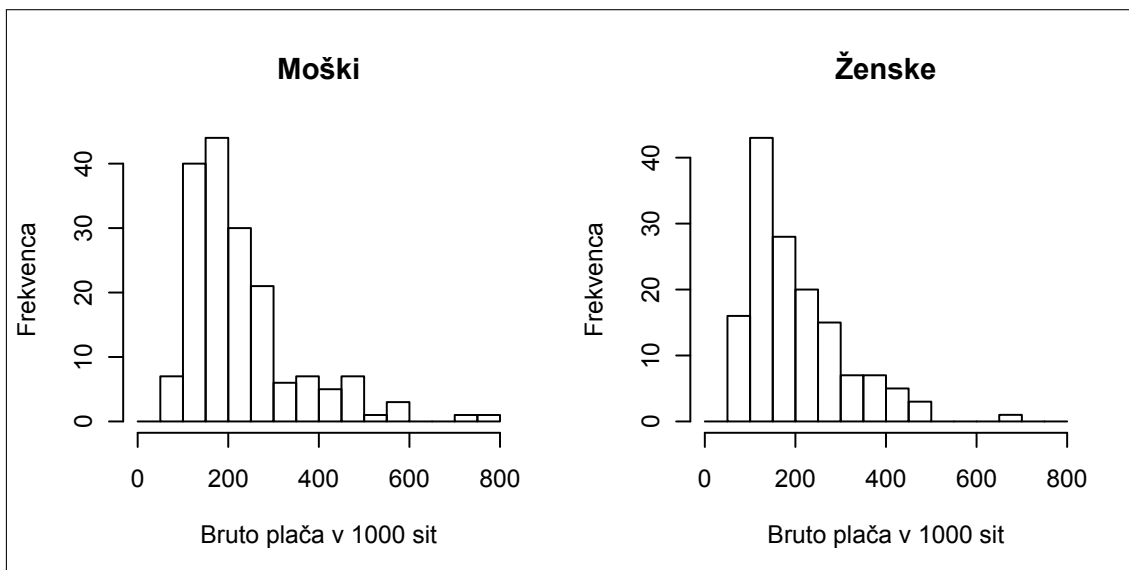
Poglejmo si najprej opisne statistike in porazdelitvi spremenljivke na obeh vzorcih (slika 2.4).

```

> #opisne statistike
> describeBy(x=data$G91,group=data$O1F2,mat=TRUE)
  item group1 vars  n    mean    sd median  trimmed
11    1  moski   1 173 237.5896 126.3618   200 218.4460
12    2  zenski   1 145 209.9103 112.7086   184 195.1026
      mad min max range    skew kurtosis    se

```

Slika 2.4: Porazdelitev bruto plače pri moških in ženskah



```

11 88.9560 61 760 699 1.573864 2.777364 9.607112
12 97.8516 53 700 647 1.263913 1.745150 9.359942
> #poglejmo porazdelitev
> par(mfrow=c(1,2))
> hist(data$G91[data$O1F2=="moski"],br=(0:16)*50,
      xlab="Bruto plača v 1000 sit",main="Moški",ylab="Frekvenca")
> hist(data$G91[data$O1F2=="zenski"],br=(0:16)*50,
      xlab="Bruto plača v 1000 sit",main="Ženske",ylab="Frekvenca")
> par(mfrow=c(1,1))

```

Predpostavka klasične različice t-testa za neodvisne vzorce je tudi enakost varianc. Zato izvedemo tudi test enakosti varianc, a se nanj ni dobro preveč opirati. Pri velikih vzorcih je namreč precej občutljiv in zazna razlike v variancah veliko prej, kot le-te začnejo vplivati na veljavnost klasičnega t-testa.

```

> var.test(G91~O1F2,data=data)
      F test to compare two variances

data:  G91 by O1F2
F = 1.2569, num df = 172, denom df = 144, p-value =
0.1561
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.9159805 1.7172157
sample estimates:

```

```

ratio of variances
      1.256949
> fligner.test(G91~O1F2,data=data)
      Fligner-Killeen test of homogeneity of variances

data:  G91 by O1F2
Fligner-Killeen:med chi-squared = 0.13014, df = 1,
p-value = 0.7183
> #neobčutljiv na odstopanja od normalnosti
> library(car)
> leveneTest(G91~O1F2,data=data)
Levenes Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  1  0.0538 0.8167
      316
> #neobčutljiv na odstopanja od normalnosti
> leveneTest(G91~O1F2,data=data, center=mean)
Levenes Test for Homogeneity of Variance (center = mean)
      Df F value Pr(>F)
group  1  0.369  0.544
      316
> #različica, kot jo uporablja SPSS
> #malce manj robustna

```

Glede na to, da so vsi testi enakosti varianc pokazali, da domneve o enakih variancah ne moremo zavrnila, lahko (brez zadržkov, kar se tiče enakosti varianc) izvedemo t-test s predpostavko enakih varianc. Poleg tega bomo izračunali tudi 90-% interval zaupanja za razliko aritmetičnih sredin.

```

> t.test(G91~O1F2,data=data,var.equal=TRUE, conf.level = 0.90)
      Two Sample t-test

data:  G91 by O1F2
t = 2.043, df = 316, p-value = 0.04188
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
  5.328537 50.029964
sample estimates:
mean in group moski mean in group zenski
      237.5896           209.9103

```

Pri 5-% tveganju so razlike statistično značilne, torej lahko s takim tveganje trdimo, da imajo moški v povprečju višjo plačo tudi na populaciji. Z 10-% tveganjem lahko trdimo, da imajo moški v povprečju od 5.33 do 50.03 tisoč sit višjo plačo kot ženske.

Če ne bi želeli predpostaviti enakih varianc, bi bila koda, kot sledi. Testa ne bomo opravili, ker v tem primeru ni potrebe.

```
> t.test(G91~01F2,data=data,var.equal=FALSE, conf.level = 0.90)
> #ali
> t.test(G91~01F2,data=data, conf.level = 0.90)
> #argumenta "var.equal=FALSE" ni treba navajati,
> #ker je to privzeta vrednost
```

Sedaj pa opravimo še ustrezen neparametričen test, torej Mann-Whitneyjev test.

```
> wilcox.test(G91~01F2,data=data)
      Wilcoxon rank sum test with continuity correction

data:  G91 by 01F2
W = 14496, p-value = 0.0167
alternative hypothesis: true location shift is not equal to 0
```

Tudi tu ugotovimo, da so razlike med srednjimi vrednostmi statistično značilne pri 5-% tveganju oz. natančneje v tem primeru pri 1,7-% tveganju.

## 2.4 Preverjanje domnev o deležih in pripadajoči intervali zaupanja

Domneve o deležih lahko preverjamo na več načinov. V primeru velikih vzorcev običajno predpostavljamo, da se deleži ali razlike med deleži porazdeljujejo približno normalno. Alternativni način za preverjanje domnev o deležih je s pomočjo kontingenčnih tabel in  $\chi^2$  porazdelitve, vendar je tudi ta le približen. Za posamezne probleme bomo obravnavali tudi natančne teste, ki pa se od problema do problema razlikujejo.

V praksi se pogosto uporablja tudi pristop, pri katerem se deleže obravnava kot povprečja spremenljivk, ki imajo vrednosti 0 in 1. V primeru dovolj velikih vzorcev in neekstremnih deležev (ki niso blizu 0 ali 1) je ta pristop primeren.

### 2.4.1 Preverjanje domneve o vrednosti deleža in pripadajoči interval zaupanja

Nekih volitev se jih je udeležilo 68.7 % anketirancev. Preverili bomo domnevo, da se je volitev udeležilo 70 % volilnih upravičencev. Ker je vzorec velik in delež ni ekstremen, uporaba približkov ni problematična.

```

> #pogledamo frekvenčno tabelo odgovorov (več o tem kasneje)
> tbl<-table(data$B11)
> tbl
  da ne
940 432
> #izračunamo deleže
> prop.table(tbl)
      da      ne
0.6851312 0.3148688
> #število tistih, ki so odgovorili z da ali ne
> n<-sum(tbl)
> n
[1] 1372
> #delež tistih, ki so glasovali
> p<-tbl[1]/n
> p
      da
0.6851312
> #preverjanje domneve s pomočjo klasičnega z-testa
> #H0: pi=piH=0.70
> piH<-0.70
> #izračun standardne napake
> seP<-sqrt(p*(1-p)/n)
> #izračun z statistike
> z<-(p-piH)/seP
> #izračunamo p-vrednost (dvostranska)
> 2*pnorm(-abs(z))
      da
0.2357121

```

Domnevo bi lahko zavrnilo šele pri 23.6-% tveganju in je zato ne zavrnilo. Torej ne moremo trditi, da se volitev ni udeležilo 70 % volilnih upravičencev (ampak več ali manj).

Namesto normalne bi bilo sicer boljše uporabiti  $t$ -porazdelitev (ker smo standardno napako ocenili na vzorcu), a pri tako velikih vzorcih v obeh primerih dobimo enak rezultat.

```
> 2*pt(-abs(z),df=n-1)
      da
0.2359176
```

Izračunajmo sedaj še klasična intervala zaupanja (z uporabo normalne in  $t$  porazdelitve).

```
> alfa <- 0.1
> (intZ<-p + qnorm(c(alfa/2,1-alfa/2))*seP)
[1] 0.6645058 0.7057566
> (intT<-p + qt(c(alfa/2,1-alfa/2),df=n-1)*seP)
[1] 0.6644919 0.7057705
```

S 10-% tveganjem torej lahko trdimo, da se je volitev udeležilo med 0.66 % (0.66 % pri  $t$ -porazdelitvi) in 0.71 % volilnih upravičencev.

Sedaj bomo preverili domnevo še s  $\chi^2$ -testom in natančnim testom (preko binomske porazdelitve). Test preko binomske porazdelitve je pravzaprav najnatančnejši. Obe funkciji zraven izračunata tudi interval zaupanja.

```
> #preko hi-kvadrat testa
> prop.test(x=tbl[1],n=n,p=piH, conf.level=1-alfa)
      1-sample proportions test with continuity correction

data:  tbl[1] out of n, null probability piH
X-squared = 1.3745, df = 1, p-value = 0.241
alternative hypothesis: true p is not equal to 0.7
90 percent confidence interval:
 0.6637884 0.7057324
sample estimates:
      p
0.6851312
> #dobimo tudi interval zaupanja
>
> #še natančen test
> binom.test(x=tbl[1],n=n,p=piH, conf.level=1-alfa)
      Exact binomial test

data:  tbl[1] and n
number of successes = 940, number of trials = 1372,
p-value = 0.2273
alternative hypothesis: true probability of success is not equal
to 0.7
```

```

90 percent confidence interval:
 0.6638601 0.7058132
sample estimates:
probability of success
 0.6851312

```

Rezultatov zaradi podobnosti s prejšnjimi ne bom interpretiral.

Za konec naredimo še test preko povprečij, torej t-test. Tu upoštevamo, da je povprečje spremenljivke, ki ima vrednosti le 0 in 1, pravzaprav delež enic. Ta test je zaradi popolne kršitve predpostavke o normalnosti manj primeren, čeprav približek pri velikih vzorcih ni slab.

```

> x<-data$B11
> x<-(x=="da")*1
> t.test(x,mu=piH, conf.level=1-alfa)
      One Sample t-test

data:  x
t = -1.1853, df = 1371, p-value = 0.2361
alternative hypothesis: true mean is not equal to 0.7
90 percent confidence interval:
 0.6644843 0.7057780
sample estimates:
mean of x
0.6851312

```

Kot vidimo, so rezultati v tem primeru (zaradi velikega vzorca) res skoraj identični.

## 2.4.2 Preverjanje domnev o razliki med deležema na neodvisnih vzorcih in pripadajoči intervali zaupanja

Tudi tu lahko uporabimo podobne pristope kot pri prejšnjem primeru. Preverimo domnevo, da je delež volivcev enak pri ženskah in moških. Najprej izračunamo kontingenčno tabelo (več o tem v podpodpoglavju 2.5.2).

```

> #izračunamo kontingenčno tabelo
> tbl<-table(data$O1F2,data$B11)
> tbl
      da ne
moski 427 188
zenski 492 235

```

```

> #izračunamo deleže
> prop.table(tbl,margin=1)
           da      ne
moski  0.6943089 0.3056911
zenski 0.6767538 0.3232462
> #število enot
> n<-apply(tbl,1,sum)
> n
moski zenski
   615   727
> #delež tistih, ki so glasovali
> p<-tbl[,1]/n
> p
moski   zenski
0.6943089 0.6767538

```

Najprej zopet s pomočjo klasičnega z-testa preverimo domnevo.

```

> #H0: piM=piZ
> #izračun standardne napake
> pSkup<-sum(tbl[,1])/sum(n)
> seP<-sqrt(sum(pSkup*(1-pSkup)/n))
> #izračun z statistike
> z<-(p[1]-p[2])/seP
> #izračunamo p-vrednost (dvostranska)
> 2*pnorm(-abs(z))
moski
0.4903852

```

Domnevo o enakosti deležev bi lahko zavrnili šele pri 49-% tveganju in je zato ne zavrnemo. Tako ne moremo trditi, da se delež volivcev na populaciji med spoloma razlikuje. Sledijo še ostali testi, torej  $\chi^2$ -test in test preko povprečij. Pri obeh dobimo tudi interval zaupanja za razliko deležev.

```

> #preko hi-kvadrat testa
> prop.test(x=tbl, conf.level=1-alfa)
2-sample test for equality of proportions with
continuity correction

data:  tbl
X-squared = 0.39783, df = 1, p-value = 0.5282
alternative hypothesis: two.sided
90 percent confidence interval:

```

```

-0.02575264  0.06086296
sample estimates:
  prop 1    prop 2
0.6943089 0.6767538
> #dobimo tudi interval zaupanja
>
> #preko povprečij
> x<-data$B11
> x<-(x=="da")*1
> spol<-data$O1F2
> t.test(x~spol, var.equal=TRUE, conf.level=1-alfa)
      Two Sample t-test

data:  x by spol
t = 0.6893, df = 1340, p-value = 0.4908
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 -0.02436485  0.05947517
sample estimates:
 mean in group moski mean in group zenski
      0.6943089          0.6767538

```

Tu ugotovimo, da ne moremo trditi, da se delež volivcev na populaciji med spoloma razlikuje. Kot rečeno, dobimo tudi intervala zaupanja. Če upoštevamo prvega, ki je zanesljivejši, lahko pri 10-% tveganju trdimo, da je razlika v deležu volivcev med moškimi in ženskami med  $-0.02575$  in  $0.06086$ .

## 2.5 Frekvenčne in kontingenčne tabele

### 2.5.1 Frekvenčne tabele

Frekvenčne in kontingenčne tabele v **R**-ju dobimo z ukazom `table`, ki izračuna samo frekvence. Za dodajanje vsote ("Skupaj") lahko potem uporabimo funkcijo `addmargins`, za izračun odstotkov pa `prop.table`.

Funkcija `table` s privzetimi vrednostmi ne upošteva manjkajočih vrednosti. Če želimo imeti v tabeli tudi manjkajoče vrednosti (`NA`), moramo pri klicu funkcije `table` nastaviti `exclude=NULL`.

Malce bogatejšo frekvenčno tabelo je moč dobiti tudi s funkcijo `frekTab`, ki se nahaja v datoteki "UcbenikR-funkcije.R".

Za risanje frekvenčnih tabel sta primerni predvsem funkciji `barplot` in `pie`.

Začnimo s frekvenčno tabelo.

```

> tbl<-table(data$F5)
> tbl
veliko mesto  predmestje manjse mesto      vas
           132           204           326      626
  kmetija
           148
> addmargins(tbl) #a skupaj
veliko mesto  predmestje manjse mesto      vas
           132           204           326      626
  kmetija
           148           Sum
           148           1436
> addmargins(prop.table(tbl)) # % + skupaj
veliko mesto  predmestje manjse mesto      vas
 0.09192201  0.14206128  0.22701950  0.43593315
  kmetija
 0.10306407  1.00000000
> barplot(tbl)
> pie(tbl)

```

Z argumentom `exclude=NULL` zahtevamo, naj se izpišejo tudi manjkajoče vrednosti (oz. natančneje, naj se iz tabele ničesar ne izpusti).

```

> #z NA
> table(data$F5,exclude=NULL)
veliko mesto  predmestje manjse mesto      vas
           132           204           326      626
  kmetija
           148           <NA>
           148           6

```

Še primer malce bogatejše frekvenčne tabele s funkcijo `frekTab` (iz datoteke "UcbenikR-funkcije.R").

```

> #s funkcijo frekTab
> frekTab(data$F5)
           Frekvenca Kum. frek.           % Kumulativni %
veliko mesto           132           132  9.192201           9.192201
predmestje           204           336 14.206128           23.398329
manjse mesto           326           662 22.701950           46.100279
vas           626           1288 43.593315           89.693593
kmetija           148           1436 10.306407           100.000000

```

## 2.5.2 Kontingenčne tabele

Za kontingenčne tabele lahko uporabimo iste funkcije kot za frekvenčne tabele. Če želimo na primer na podlagi dveh spremenljivk oblikovati osnovno kontingenčno tabelo, uporabimo funkcijo `table` in ji kot argumente podamo spremenljivke, na podlagi katerih želimo oblikovati tabelo (najprej spremenljivka v vrsticah, nato v stolpcih). Če pri podajanju spremenljivk (argumentov) podamo tudi imena argumentov (npr. `Kraj=data$F5` spodaj), postanejo imena argumentov imena posameznih dimenzij tabele.

Najprej izračunajmo kontingenčno tabelo za kraj bivanja in spol skupaj z vsotami. Vrednosti v tabeli tudi pretvorimo v deleže in zopet dodamo vsote.

```
> tbl2D<-table(Kraj=data$F5,Spol=data$O1F2)
> tbl2D
```

Kraj	Spol	
	moski	zenski
veliko mesto	56	72
predmestje	94	110
manjse mesto	151	160
vas	264	351
kmetija	78	69

```
> #skupaj z vsotami
> addmargins(tbl2D)
```

Kraj	Spol		Sum
	moski	zenski	
veliko mesto	56	72	128
predmestje	94	110	204
manjse mesto	151	160	311
vas	264	351	615
kmetija	78	69	147
Sum	643	762	1405

```
> #preračun v skupne deleže
> prop.table(tbl2D)
```

Kraj	Spol	
	moski	zenski
veliko mesto	0.03985765	0.05124555
predmestje	0.06690391	0.07829181
manjse mesto	0.10747331	0.11387900
vas	0.18790036	0.24982206
kmetija	0.05551601	0.04911032

```
> ptbl<-prop.table(tbl2D)
> addmargins(tbl2D)
```

```

      Spol
Kraj      moski zenski Sum
veliko mesto    56    72 128
predmestje     94   110 204
manjse mesto   151   160 311
vas            264   351 615
kmetija        78    69 147
Sum           643   762 1405
> #preračun v deleže po stolpcih
> prop.table(tbl2D,margin=2)
      Spol
Kraj      moski    zenski
veliko mesto 0.08709176 0.09448819
predmestje   0.14618974 0.14435696
manjse mesto 0.23483670 0.20997375
vas          0.41057543 0.46062992
kmetija      0.12130638 0.09055118
> addmargins(tbl2D,margin=2)
      Spol
Kraj      moski zenski Sum
veliko mesto    56    72 128
predmestje     94   110 204
manjse mesto   151   160 311
vas            264   351 615
kmetija        78    69 147
> ptbl<-addmargins(prop.table(addmargins(tbl2D,margin=2),
margin=2),margin=1)
> ptbl
      Spol
Kraj      moski    zenski    Sum
veliko mesto 0.08709176 0.09448819 0.09110320
predmestje   0.14618974 0.14435696 0.14519573
manjse mesto 0.23483670 0.20997375 0.22135231
vas          0.41057543 0.46062992 0.43772242
kmetija      0.12130638 0.09055118 0.10462633
Sum          1.00000000 1.00000000 1.00000000
> #v % na 2 decimalki
> round(ptbl*100,2)
      Spol
Kraj      moski zenski    Sum
veliko mesto  8.71  9.45  9.11
predmestje   14.62 14.44 14.52
manjse mesto 23.48 21.00 22.14

```

vas	41.06	46.06	43.77
kmetija	12.13	9.06	10.46
Sum	100.00	100.00	100.00

Za dvodimenzionalne tabele pa lahko za bogatejši izpis (tipa SPSS) uporabimo tudi funkcijo *CrossTable* iz paketa *gmodels*.

```
> #enostavneje s paketkom gmodels
> # install.packages("gmodels")
> library(gmodels)
> CrossTable(data$F5,data$O1F2)
```

Cell Contents

```
|-----|
|                N |
| Chi-square contribution |
|      N / Row Total |
|      N / Col Total |
|      N / Table Total |
|-----|
```

Total Observations in Table: 1405

	data\$O1F2		
data\$F5	moski	zenski	Row Total
veliko mesto	56	72	128
	0.114	0.096	
	0.438	0.562	0.091
	0.087	0.094	
	0.040	0.051	
predmestje	94	110	204
	0.004	0.004	
	0.461	0.539	0.145
	0.146	0.144	
	0.067	0.078	
manjše mesto	151	160	311
	0.528	0.446	
	0.486	0.514	0.221
	0.235	0.210	

	0.107	0.114	
vas	264	351	615
	1.083	0.914	
	0.429	0.571	0.438
	0.411	0.461	
	0.188	0.250	
kmetija	78	69	147
	1.710	1.443	
	0.531	0.469	0.105
	0.121	0.091	
	0.056	0.049	
Column Total	643	762	1405
	0.458	0.542	

```
> # želimo samo % po stolpcih + hi-kvadrat test - SPSS format
> CrossTable(data$F5,data$01F2, prop.r=FALSE, prop.c=TRUE,
  prop.t=FALSE, prop.chisq=FALSE, chisq = TRUE, format=c("SPSS"))
Cell Contents
```

Count
Column Percent

Total Observations in Table: 1405

	data\$01F2		Row Total
data\$F5	moski	zenski	
veliko mesto	56	72	128
	8.709%	9.449%	
predmestje	94	110	204
	14.619%	14.436%	
manjse mesto	151	160	311
	23.484%	20.997%	
vas	264	351	615
	41.058%	46.063%	

kmetija	78	69	147
	12.131%	9.055%	
-----	-----	-----	-----
Column Total	643	762	1405
	45.765%	54.235%	
-----	-----	-----	-----

Statistics for All Table Factors

Pearsons Chi-squared test

-----  
 Chi^2 = 6.340168      d.f. = 4      p = 0.1751438

Minimum expected frequency: 58.57936

Frekvenčne tabele z več kot dvema dimenzijama lahko ustvarimo na enak način kot dvodimenzionalne tabele s funkcijo `table`, a je izpis pogosto zelo nepregleden. Za preglednejši izpis lahko uporabimo funkcijo `fTable` ali tabelo pred izpisom pretvorimo v podatkovni okvir z `as.data.frame`. Kljub temu je izpis zelo dolg in ga zato izpuščam.

```
> #3 in več dimenzionalne tabele
> tbl3D<-table(data$O1F2,data$F5,data$F6)
> tbl3D
> #izpis postane zelo nepregleden
>
> fTable(tbl3D)
> as.data.frame(tbl3D)
```

### 2.5.3 Povezanost spremenljivk

S pomočjo kontingenčnih tabel in na njih osnovanih mer lahko merimo in preverjamo povezanost med nominalnimi spremenljivkami.

Za preverjanje domneve o neodvisnosti med dvema spremenljivkama lahko uporabimo  $\chi^2$ -test. Na voljo je v funkciji `chisq.test`, ki kot argument sprejme vektor, matriko ali dvodimenzionalno kontingenčno tabelo. Kot smo že videli, pa  $\chi^2$ -test vrne (če tako izberemo) tudi funkcija `CrossTable` iz paketa `gmodels`.

Funkcija `chisq.test` s privzetimi vrednostmi argumentov pri 2x2 tabelah uporabi tudi Yatesov popravek, omogoča pa tudi izračun (bolj) natančnih  $p$  vrednosti s pomočjo simulacij (preko permutacijskega testa).

Statistiki si glede uporabe Yatesovega popravka niso enotni.  $\chi^2$ -test ni natančen test, ampak le približek. Uporaba Yatesovega popravka lahko v nekaterih primerih ta približek izboljša, lahko pa naredi  $\chi^2$ -test tudi preveč konzervativen.

Za 2x2 tabele je na voljo tudi Fisherjev natančni test preko funkcije `fisher.test`.

Naredimo najprej običajni  $\chi^2$ -test za povezanost tipa kraja bivanja in spola. Pričakujemo, da ničelne domneve ne bomo mogli zavriniti.

```
> tblF501F2<-table(data$F5,data$01F2)
> tblF501F2
           moski zenski
veliko mesto    56    72
predmestje     94   110
manjse mesto   151   160
vas            264   351
kmetija        78    69
> chisq.test(tblF501F2)
      Pearson's Chi-squared test

data:  tblF501F2
X-squared = 6.3402, df = 4, p-value = 0.1751
```

Domnevo bi lahko zavrnilo šele pri 17.51-% tveganju in je zato ne zavrnamo.

Poglejmo še primer tabele 2x2, kjer bomo preverjali domnevo o povezanosti (ničelna domneva seveda predpostavlja neodvisnost) spremenljivk spol in "članstvo v politični stranki". Ker gre za tabelo 2x2 lahko uporabimo Yatesov popravek. Izračunali bomo obe možnosti (s popravkom in brez njega).

```
> attributes(data)$variable.labels["B21"]
      B21
"Ali ste član kakšne politične stranke?"
> data$B21<-factor(data$B21)
> (tblGndrB<-table(data$B21,data$01F2)) #2x2 tabela
           moski zenski
da        29    21
ne       618   738
```

```

> #hi-kvadrat test z Yatesovim popravkom (privzeta možnost)
> chisq.test(tblGndrB)
      Pearson's Chi-squared test with Yates continuity
      correction

data:  tblGndrB
X-squared = 2.5174, df = 1, p-value = 0.1126
> #hi-kvadrat test brez Yatesovega popravka
> chisq.test(tblGndrB, correct = FALSE)
      Pearson's Chi-squared test

data:  tblGndrB
X-squared = 2.9967, df = 1, p-value = 0.08343

```

Če uporabimo privzeto možnost, če torej uporabimo Yatesov popravek, lahko tudi tu ugotovimo, da niti pri 10-% tveganju ne moremo trditi, da sta spremenljivki spol in "članstvo v politični stranki" povezani.

Če pa tega popravka ne bi uporabili, bi pri 10-% tveganju že lahko trdili, da sta spremenljivki povezani.

Ker je  $\chi^2$ -test pravzaprav le približek, tukaj pa imamo celo dve različici, pravilnost izračuna preverimo preko simulacij in Fisherjevega natančnega testa.

```

> chisq.test(tblGndrB, sim=TRUE) #p-vrednost preko simulacij
      Pearson's Chi-squared test with simulated p-value
      (based on 2000 replicates)

data:  tblGndrB
X-squared = 2.9967, df = NA, p-value = 0.1134
> fisher.test(tblGndrB)
      Fishers Exact Test for Count Data

data:  tblGndrB
p-value = 0.1112
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.8979363 3.0744330
sample estimates:
odds ratio
 1.648488

```

Oba izračuna potrdita pravilnost izračuna  $\chi^2$  z Yatesovim popravkom. V primeru, da bi izbrali  $\chi^2$ -test brez Yatesovega popravka, bi dobili precej manjšo stopnjo tveganja (približno 8-%), ki pa bi bila glede na rezultate simulacij in Fisherjevega

natančnega testa napačna. Uporaba popravka je torej v tem primeru primerna, ni pa vedno tako, saj lahko popravek naredi test preveč konzervativen.

Povezanost spremenljivk merimo s kontingenčnimi koeficienti. Ti so med drugim na voljo v funkciji `assocstats` v paketu `vcd`. Razlaga vrnjenih koeficientov (glede na Ferligoj 1994, 166):

**Phi-Coefficien** Koren Pearsonovega koeficienta – pravzaprav gre v primeru binarnih spremenljivk za Pearsonov korelacijski koeficient in ima tudi enako interpretacijo.

**Contingency Coeff.** Kontingenčni koeficient (brez popravka).

**Cramer's V** Kramerjev koeficient.

```
> # install.packages("vcd")
> library(vcd)
> tblF501F2<-table(data$F5,data$O1F2)
> assocstats(tblF501F2)
              X^2 df P(> X^2)
Likelihood Ratio 6.3309  4  0.17576
Pearson          6.3402  4  0.17514

Phi-Coefficient   : NA
Contingency Coeff.: 0.067
Cramers V        : 0.067
```

Vsi koeficienti imajo vrednost 0.067 kar kaže, da spremenljivki pravzaprav nista povezani (oz. sta zelo zelo šibko povezani).

## 2.6 Korelacija

Povezanost med ordinalnimi spremenljivkami in intervalnimi ali ordinalnimi spremenljivkami merimo s pomočjo korelacijskih koeficientov.

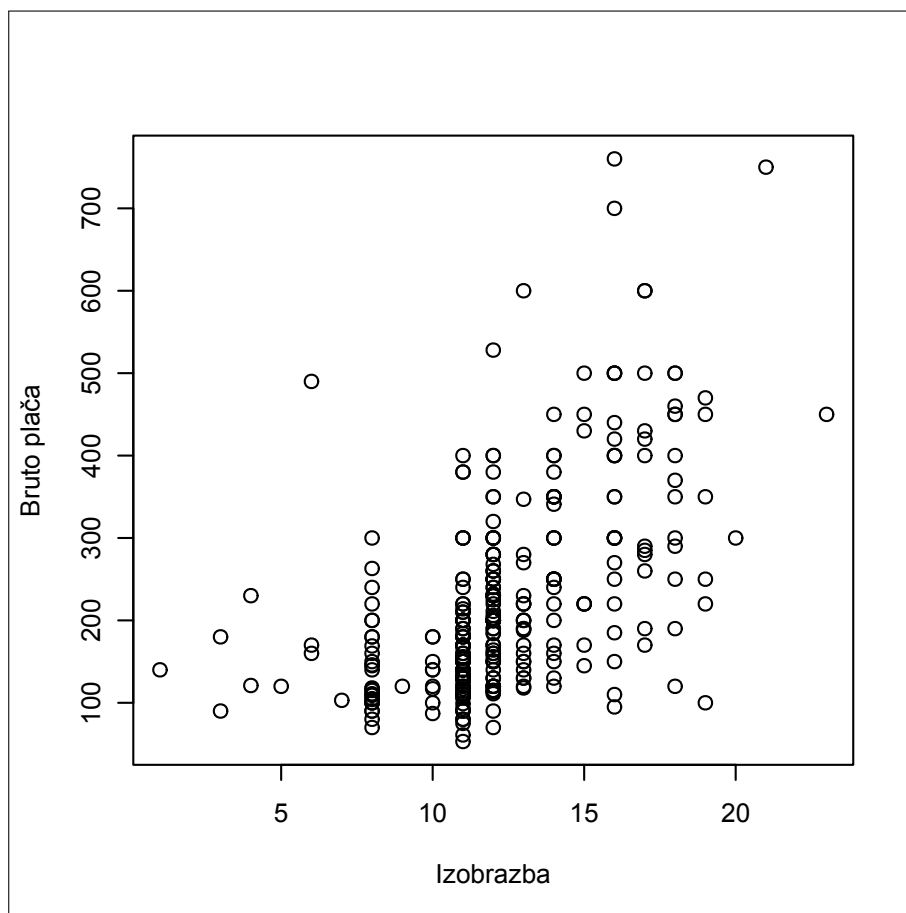
V **R**-ju lahko korelacijsko matriko izračunamo s pomočjo funkcije `cor`, domneve o korelaciji pa lahko preverjamo s funkcijo `cor.test`. Prva (`cor`) izračuna korelacijo med vsemi spremenljivkami v podatkovnem okvirju ali matriki, druga (`cor.test`) izračuna in testira le eno korelacijo naenkrat. Obe funkciji poznata naslednje korelacijske koeficiente:

**pearson** Pearsonov koeficient (linearne) korelacije (privzeta možnost),

**spearman** Spearmanov koeficient korelacije,

**kendall** Kendallov ( $\tau$ ) koeficient korelacije/konkordance.

Slika 2.5: Razsevni grafikon – izobrazba in bruto plača



Od teh treh je morda le Kendallov manj znan. Tako kot Spearmanov koeficient korelacije je tudi Kendallov uporaben na ordinalnih spremenljivkah, njegova vrednost pa se izračuna kot:

$$\frac{(\text{število parov, ki se ujema}) - (\text{število parov, ki se ne ujema})}{\text{število vseh možnih parov}}$$

Pari, ki se ujema, so pari, kjer je vrstni red enot (v paru) glede na obe spremenljivki enak, tisti, ki se ne ujema, pa tisti, kjer imata enoti drugačen vrstni red pri spremenljivkah. Število vseh parov je  $\frac{n(n-1)}{2}$ .

Za primer vzemimo korelacijo med številom let šolanja (F7) in bruto plačo (G91). Odnos je prikazan tudi na sliki 2.5. Očitno je zveza med spremenljivkama pozitivna, a nelinearna. Izračunajmo in testirajmo vse tri koeficiente.

```
> #narišemo razsevni grafikon
> plot(G91~F7,data=data,xlab="Izobrazba",ylab="Bruto plača")
```

```

> #izračunamo Pearsonov koeficient korelacije
> cor(x=data$F7,y=data$G91,use="complete.obs",method="pearson")
[1] 0.550864
> #in preverimo domnevo (ki tudi izračuna korelacijo)
> cor.test(x=data$F7,y=data$G91, method="pearson")
      Pearsons product-moment correlation

data:  data$F7 and data$G91
t = 11.862, df = 323, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4702540 0.6223585
sample estimates:
      cor
0.550864
> #Spearmanov
> cor.test(x=data$F7,y=data$G91, method="spearman")
      Spearmans rank correlation rho

data:  data$F7 and data$G91
S = 2392800, p-value < 2.2e-16
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.5817806
> #Kendallov
> cor.test(x=data$F7,y=data$G91, method="kendall")
      Kendalls rank correlation tau

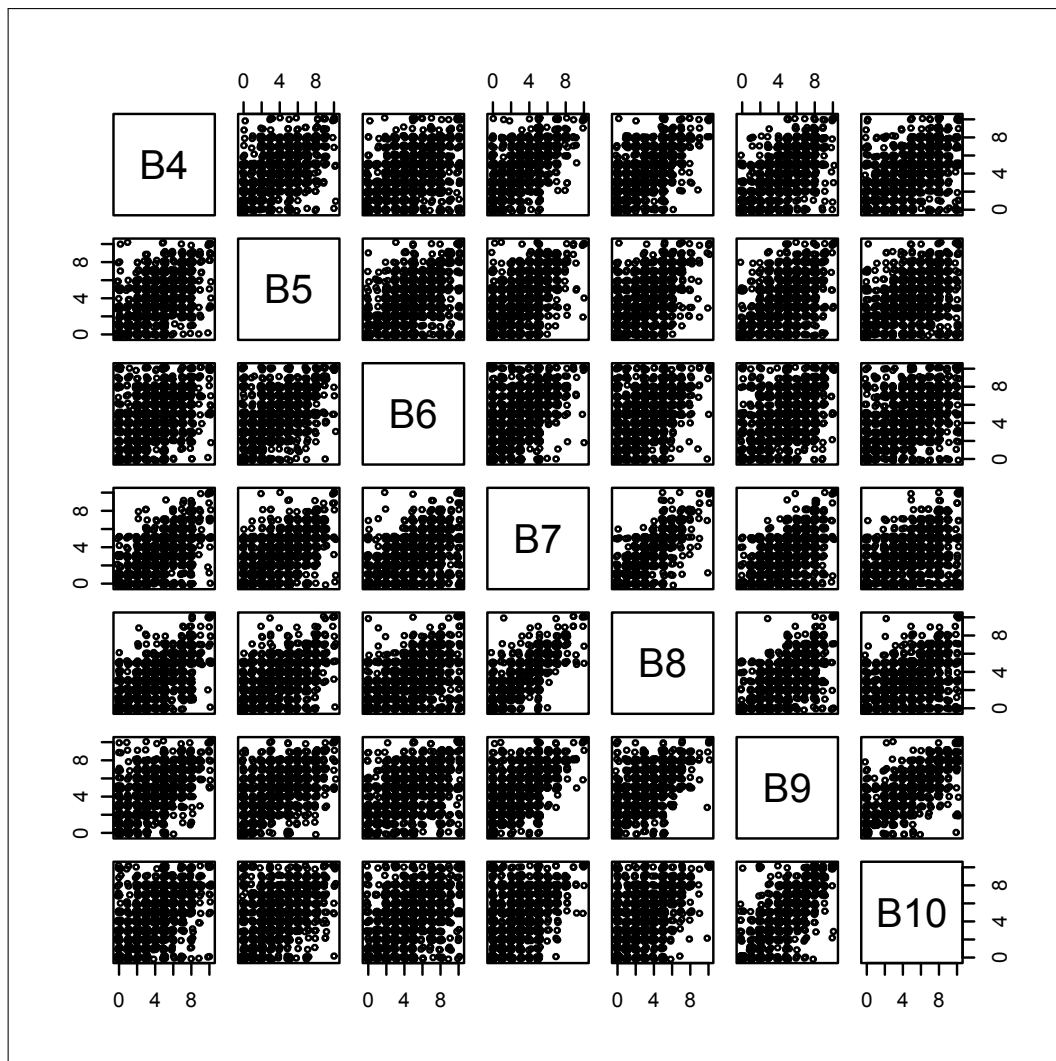
data:  data$F7 and data$G91
z = 11.146, p-value < 2.2e-16
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
0.4474194

```

Vsi trije koeficienti kažejo na srednje močno povezanost, a vseeno je vrednost Kendallovega koeficienta bistveno nižja, vrednost Spearmanovega pa najvišja.

Kot smo že omenili, funkcija `cor` izračuna korelacijo med vsemi spremenljivkami v podatkovnem okvirju, funkcija `cor.test` pa izračuna in testira le eno korelacijo naenkrat. V ta namen je v datoteki "UcbenikR-funkcije.R" funkcija `corTestDf`, ki omogoča, da funkcijo `cor.test` uporabimo na več spremenljivkah. Lepši izpis se dobi z uporabo funkcije `printCorTestDf`.

Slika 2.6: Razsevni grafikon med vsemi spremenljivkami, ki merijo zaupanje v institucije



Poglejmo si torej korelacije in statistične značilnosti na primeru sklopa spremenljivk B4-B10 (zaupanje v institucije). Izračunali bomo Pearsonove in Kendallove koeficiente. Hiter pregled povezanosti med njimi je na sliki 2.6 (točke so zaradi boljše preglednosti "zatresene").

```
> #izbor shranimo
> izbor<-c("B4", "B5", "B6", "B7", "B8", "B9", "B10")
> #izpišemo imena
> attributes(data)$variable.labels[izbor]
           B4                B5
"državnemu zboru"  "pravnemu sistemu"
```

```

          B6          B7
      "policiji"      "politikom"
          B8          B9
"političnim strankam" "Evropskemu parlamentu"
          B10
      "Združenim narodom"
> #narišemo grafe
> pairs(apply(data[izbor],2,jitter),cex=0.3)
> #izračunamo Pearsonove korelacije
> cor(data[izbor],use="pairwise.complete.obs",method="pearson")
          B4          B5          B6          B7          B8
B4  1.0000000 0.5946718 0.4616891 0.6415600 0.6311402
B5  0.5946718 1.0000000 0.5633582 0.5902897 0.5644650
B6  0.4616891 0.5633582 1.0000000 0.4800425 0.4414586
B7  0.6415600 0.5902897 0.4800425 1.0000000 0.8322464
B8  0.6311402 0.5644650 0.4414586 0.8322464 1.0000000
B9  0.6179115 0.5648233 0.4733737 0.5792044 0.5935754
B10 0.5188249 0.4824849 0.4596825 0.4667185 0.4893612
          B9          B10
B4  0.6179115 0.5188249
B5  0.5648233 0.4824849
B6  0.4733737 0.4596825
B7  0.5792044 0.4667185
B8  0.5935754 0.4893612
B9  1.0000000 0.7889287
B10 0.7889287 1.0000000
> tmp<-corTestDf(data[izbor],method="pearson")
> printCorTestDf(tmp)
          B4          B5          B6          B7          B8          B9          B10
B4  cor          0.595 0.462 0.642 0.631 0.618 0.519
    p          0.000 0.000 0.000 0.000 0.000 0.000
    n    1387  1358  1368  1374  1373  1231  1276
B5  cor 0.595          0.563 0.590 0.564 0.565 0.482
    p 0.000          0.000 0.000 0.000 0.000 0.000
    n 1358  1376  1365  1365  1364  1228  1274
B6  cor 0.462 0.563          0.480 0.441 0.473 0.460
    p 0.000 0.000          0.000 0.000 0.000 0.000
    n 1368  1365  1401  1380  1372  1238  1287
B7  cor 0.642 0.590 0.480          0.832 0.579 0.467
    p 0.000 0.000 0.000          0.000 0.000 0.000
    n 1374  1365  1380  1398  1380  1236  1283
B8  cor 0.631 0.564 0.441 0.832          0.594 0.489

```

```

      p    0.000 0.000 0.000 0.000      0.000 0.000
      n    1373 1364 1372 1380 1392 1236 1285
B9  cor    0.618 0.565 0.473 0.579 0.594      0.789
      p    0.000 0.000 0.000 0.000 0.000      0.000
      n    1231 1228 1238 1236 1236 1247 1227
B10 cor    0.519 0.482 0.460 0.467 0.489 0.789
      p    0.000 0.000 0.000 0.000 0.000 0.000
      n    1276 1274 1287 1283 1285 1227 1295
> #še Kendalllove
> cor(data[izbor],use="pairwise.complete.obs",method="kendall")
      B4      B5      B6      B7      B8
B4  1.0000000 0.4909485 0.3667314 0.5291117 0.5160148
B5  0.4909485 1.0000000 0.4606602 0.4794322 0.4546308
B6  0.3667314 0.4606602 1.0000000 0.3844481 0.3523638
B7  0.5291117 0.4794322 0.3844481 1.0000000 0.7607492
B8  0.5160148 0.4546308 0.3523638 0.7607492 1.0000000
B9  0.5049790 0.4485398 0.3736321 0.4639373 0.4807420
B10 0.4279313 0.3891029 0.3600516 0.3761712 0.3943594
      B9      B10
B4  0.5049790 0.4279313
B5  0.4485398 0.3891029
B6  0.3736321 0.3600516
B7  0.4639373 0.3761712
B8  0.4807420 0.3943594
B9  1.0000000 0.6927734
B10 0.6927734 1.0000000
> tmp<-corTestDf(data[izbor],method="kendall")
> printCorTestDf(tmp)
      B4      B5      B6      B7      B8      B9      B10
B4  cor          0.491 0.367 0.529 0.516 0.505 0.428
      p          0.000 0.000 0.000 0.000 0.000 0.000
      n    1387 1358 1368 1374 1373 1231 1276
B5  cor 0.491          0.461 0.479 0.455 0.449 0.389
      p 0.000          0.000 0.000 0.000 0.000 0.000
      n    1358 1376 1365 1365 1364 1228 1274
B6  cor 0.367 0.461          0.384 0.352 0.374 0.360
      p 0.000 0.000          0.000 0.000 0.000 0.000
      n    1368 1365 1401 1380 1372 1238 1287
B7  cor 0.529 0.479 0.384          0.761 0.464 0.376
      p 0.000 0.000 0.000          0.000 0.000 0.000
      n    1374 1365 1380 1398 1380 1236 1283
B8  cor 0.516 0.455 0.352 0.761          0.481 0.394

```

	p	0.000	0.000	0.000	0.000	0.000	0.000
	n	1373	1364	1372	1380	1392	1236
B9	cor	0.505	0.449	0.374	0.464	0.481	0.693
	p	0.000	0.000	0.000	0.000	0.000	0.000
	n	1231	1228	1238	1236	1236	1247
B10	cor	0.428	0.389	0.360	0.376	0.394	0.693
	p	0.000	0.000	0.000	0.000	0.000	0.000
	n	1276	1274	1287	1283	1285	1227

Pri zanemarljivi stopnji tveganja lahko pri obeh koeficientih ugotovimo, da so vse spremenljivke na populaciji povezane, kar bi glede na to, da merijo sorodne stvari, in glede na velikost vzorca tudi pričakovali. Tudi tu se pokaže, da je linearna povezanost, izmerjena s Pearsonovim koeficientom, večja od tiste, izmerjene s Kendallovim.

## 2.7 Viri za poglobljanje znanja

Za poglobljanje snovi iz tega poglavja so uporabni tudi spletni viri, navedeni v podpoglavju 1.9.1 (Spletni viri), a jih ju tu ponovno ne navajam. Izpostavljam samo enega, s pomočjo katerega je mogoče zelo enostavno in hitro najti ustrezne ukaze za izvedbo zelenih analiz.

**Quick-R** Odličen hitri vodič po **R**-ju za uporabnike drugih statističnih paketov (SPSS, SAS, Stata ...), se pravi za tiste, ki statistiko že znajo. URL: <http://www.statmethods.net/>.

Za poglobljanje znanja iz statistike so primerni praktično vsi splošni statistični učbeniki, na primer:

- Minium, Edward W., Robert C. Clarke in Theodore Coladarci. 1999. *Elements of statistical reasoning*. New York: Wiley.
- Levin, Jack, James Alan Fox in David R. Forde. 2013. *Elementary Statistics in Social Research (12th Edition)*. Pearson.
- Wonnacott, Thomas H in Ronald J. Wonnacott. 1990. *Introductory statistics*. New York: Wiley.
- Košmelj, Blaženka in Jože Rován. 2007. *Statistično sklepanje*. Ljubljana: Ekonomska fakulteta.  
Slovenski učbenik, ki ga uporabljajo na Ekonomski fakulteti Univerze v Ljubljani.

Osnovne statistične analize v **R**-ju pa pokrivajo sledeči učbeniki, ki so bili sicer večinoma navedeni tudi pri virih v prvem poglavju:

- Dalgaard, Peter. 2002. *Introductory statistics with R*. New York: Springer.<sup>13</sup>
- Muenchen, Robert A. 2011. *R for SAS and SPSS users*. New York: Springer.<sup>14</sup>  
Še posebej primeren za tiste, ki že poznajo SPSS ali SAS, sicer pa zelo zgoščena obravnava.
- Verzani, John. 2005. *Using R for introductory statistics*. Boca Raton: Chapman & Hall/CRC.<sup>15</sup>

## 2.8 Vprašanja za ponavljanje

1. Kaj moramo narediti, da nam funkcije, kot so na primer *mean* in *min*, vrnejo veljavno vrednost, tudi če imamo v podatkih manjkajoče vrednosti (*NA*)? Enakovredno vprašanje je, kaj moramo storiti, da izračunajo statistiko samo na podlagi veljavnih (nemanjkajočih) vrednosti?
2. Kateri paketek vsebuje veliko funkcij, ki so zelo uporabne v družboslovju in še posebej v psihologiji?
3. S katerim argumentov večini statističnih funkcij povemo, iz katerega podatkovja (oziroma podatkovnega okvirja) naj črpa podatke (išče spremenljivke)?
4. Kakšna je razlika med parametričnimi in neparametričnimi testi?
5. Katero funkcijo ali funkcije moramo izbrati, če želimo izvesti t-test za en vzorec, za dva odvisna vzorca in za dva neodvisna vzorca? Na podlagi česa se odločimo, katerega izmed teh testov je treba izvesti?
6. Kateri test za preverjanje domneve o srednji vrednosti ne zahteva vsaj intervalne merske lestvice?
7. Ali funkcija *t.test* s privzetimi argumenti pri izvedbi t-testa za neodvisna vzorca predpostavlja enako ali različno velike variance po skupinah?
8. Kako funkcija *table* s privzetimi vrednostmi argumentov v kontingenčnih tabelah izpisuje manjkajoče vrednosti (*NA*)?
9. Kdaj lahko uporabimo Fisherjev natančni test za preverjanje domneve o povezanosti dveh nominalnih spremenljivk?
10. Katere korelacijske koeficiente lahko izračuna funkcija *cor*?

<sup>13</sup>Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>14</sup>Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>15</sup>Starejša in manj obsežna različica knjige je dostopna tudi na <http://www.math.csi.cuny.edu/Statistics/R/simpleR/printable/simpleR.pdf>

11. Kako funkcija `cor` s privzetimi argumenti obravnava manjkajoče vrednosti, če računamo korelacijsko matriko med več spremenljivkami? Kakšne so ostale možnosti?
12. Ali lahko na podlagi izpisa funkcije `cor` povemo, ali lahko ničelno domnevo o nepovezanosti dveh spremenljivk zavrnemo?

## 3. poglavje

# Analiza variance in linearna regresija

V zadnjem poglavju sta predstavljeni analiza variance in linearna regresija. Pri obeh začnemo z najenostavnejšim (bivariatnim) primerom, nato pa dodajamo dodatne elemente. Poglavje kot običajno začnemo s predstavitvijo podatkov. Sledi podpoglavje o analizi variance, kjer začnemo z enofaktorsko analizo variance, ki jo potem nadgradimo v večfaktorsko analizo variance, kjer so možne tudi interakcije med učinki neodvisnih spremenljivk. Prav tako obravnavamo tudi enofaktorsko analizo variance za odvisne vzorce in Kruskal-Wallisov test vsote rangov, neparametrično različico klasične enofaktorske analize variance za neodvisne vzorce.

Analizi variance sledi podpoglavje o linearni regresiji, kjer ponovno začnemo z enostavno bivariatno linearno regresijo. V nadaljevanju linearno regresijo najprej nadgradimo z nelinearno regresijo, večina poglavja pa je namenjena multipli linearni regresiji, torej linearni regresiji z več neodvisnimi spremenljivkami. Pri tem prikazemo tudi, kako lahko kot neodvisne spremenljivke vključimo nominalne in ordinalne spremenljivke ter kako v model vključimo interakcije med vplivi neodvisnih spremenljivk. Proti koncu poglavja posebno pozornost namenimo preverjanju predpostavk. Za bolj zagnane študente pa sta dodana podpoglavje o možnosti nadgrajevanja oziroma izpopolnjevanja predstavljenega empiričnega primera (3.3.8) in podpoglavje s prikazom izračuna ocen parametrov linearne regresije brez uporabe vgrajenih funkcij za linearno regresijo (3.3.9), predvsem z uporabo linearne algebre (matričnega računanja).

### 3.1 Uporabljeni podatki

Za prikaz predstavljenih metod bomo tudi tu uporabili podatke iz Evropske družboslovne raziskave (<http://www.europeansocialsurvey.org/>) za Slovenijo za leto

2004. Uporabljena datoteka je bila pridobljena iz Arhiva družboslovnih podatkov (Toš in drugi 2004). Opis raziskave in povezave do datoteke so dostopne na tem spletnem naslovu: <http://www.adp.fdv.uni-lj.si/opisi/sjm042/>. Iz Arhiva družboslovnih podatkov sem prenesel podatke v SPSS-ovem formatu (".sav").

Uporabili bomo predvsem sledeče spremenljivke:

**G91** bruto plača v 1000 sit,

**F5** kraj bivanja (5 kategorij),

**O1F2** spol.

Preberemo podatke iz SPSS-ove datotke.

```
> #naložimo podatke
> library(foreign)
> data<-read.spss(file="sjm042_f1.sav",
  to.data.frame = TRUE, use.value.labels = TRUE,
  max.value.labels=5, use.missings=TRUE)
> #naložimo tudi dodatne funkcije
> source("UcbenikR-funkcije.R")
```

## 3.2 Analiza variance (ANOVA)

Analiza variance (ANOVA) je splošno ime za metode, ki primerjajo pojasnjeno variabilnost (običajno merjeno z vsoto kvadratov odklonov) z nepojasnjeno. Če je pojasnjena variabilnost dovolj velika v primerjavi z nepojasnjeno, potem lahko zaključimo, da ima tisto, kar jo **pojasnjuje**, nek vpliv na obravnavano odvisno spremenljivko<sup>16</sup>.

Omejili se bomo na probleme, pri katerih preverjamo, ali se aritmetične sredine razlikujejo med vzorci oziroma skupinami. Pri takih primerih je pojasnjena variabilnost med aritmetičnimi sredinami teh vzorcev/skupin (variabilnost med vzorci), nepojasnjena pa variabilnost znotraj vzorcev (variabilnost posameznih vrednosti okoli aritmetičnih sredin posameznih vzorcev/skupin). Pojasnjena variabilnost je običajno pojasnjena z eno ali več nominalnimi<sup>17</sup> spremenljivkami.

<sup>16</sup>V splošnem je sicer odvisnih spremenljivk lahko tudi več, a tega ne bomo obravnavali.

<sup>17</sup>Oziroma spremenljivkami, ki jih obravnavamo kot nominalne.

### 3.2.1 Enofaktorska analiza variance za neodvisne vzorce

Ta verzija analize variance je najbolj znana in najpogosteje uporabljena, tako da večina ljudi ob izrazu "analiza variance" pomisli ravno nanjo.

Klasična enofaktorska analiza variance za neodvisne vzorce ima sledeče predpostavke:

**Normalnost (angl. normality)** Odvisna spremenljivka se znotraj vsake populacije porazdeljuje normalno.

**Enakost varianc (angl. homogeneity of variance)** Variabilnost je enaka v vseh (pod)populacijah (skupinah).

Če je zadoščeno tema dvema predpostavkama, lahko uporabimo F-test (analizo variance) za testiranje domnev o razliki aritmetičnih sredin.

Če imamo velik vzorec (na primer kot 30 enot v vsakem vzorcu), lahko uporabimo F-test, tudi če predpostavki o normalnosti ni popolnoma zadoščeno. Pomembno je, da je porazdelitev v vseh populacijah približno enaka (na primer povsod podobno asimetrična v desno). Pri majhnih vzorcih pa v primeru različnih varianc ne moremo uporabiti običajnega F-testa.

V primeru, da je kršena predpostavka o enakosti varianc, pa lahko uporabimo Welchovo analizo variance (Welch 1951).

Predpostavko o normalnosti običajno preverjamo grafično (histogram po skupinah), predpostavko o enakosti varianc pa preko opisnih statistik in formalnih testov. Nekaj možnih testov:

`bartlett.test` Primeren, če je predpostavka o normalnosti izpolnjena.

`fligner.test` Neobčutljiv na odstopanja od normalnosti. Temelji na rangih.

`leveneTest` Tudi ta je neobčutljiv na odstopanja od normalnosti. Z argumentom `center=mean` (kar ni privzeta možnost) je to test, ki ga uporablja SPSS. Na voljo je v paketku `car`.

Enofaktorsko analizo variance in njeno neparametrično različico lahko izvedemo s sledečimi funkcijami:

`oneway.test` Izvede klasično in Welchovo enofaktorsko analizo variance.

`aov` Bolj splošna funkcija za analizo variance (tudi večfaktorsko), ki pa ne omogoča možnosti za različne variance (vsaj ne na enostaven način).

`kruskal.test` Kruskal-Wallisov test vsote rangov – neparametrična različica analize variance oziroma različica Mann-Whitneyjevega testa za več kot 2 vzorca.

**Opozorilo!**

Funkcija `aov`, tako kot veliko drugih statističnih funkcij, kot glavni argument sprejme formulo, ki pove, vpliv katerih neodvisnih spremenljivk na katero odvisno spremenljivko preučujemo. Če želimo, da se neodvisne spremenljivke obravnavajo kot nominalne spremenljivke (in ne kot intervalne), morajo biti obvezno tipa *faktor* (ali kvečjemu *character*, ki ga funkcija samodejno spremeni v *faktor*). Sicer je tisto, kar dobimo, bolj podobno linearni regresiji.

Za primerjave, katera povprečja se statistično značilno razlikujejo, izvedemo tako imenovane "post hoc" teste. Za ta namen lahko v splošnem uporabimo funkcijo `pairwise.t.test`. Parne primerjave namreč niso nič drugega kot t-testi, kjer dobljene  $p$ -vrednosti popravimo tako, da je skupno tveganje enako izbrani stopnji  $\alpha$ . Pri funkciji `pairwise.t.test` lahko izberemo tudi metodo popravka  $p$ -vrednosti. Priporočena je Holmova metoda (Holm 1979), ki je tudi splošno veljavna<sup>18</sup> in je boljša kot Bofferonijeva.

Ko je izpolnjena predpostavka o enakih variancah in smo za analizo variance uporabili funkcijo `aov`, lahko za te primerjave uporabimo tudi Tukeyjev test preko funkcije `TukeyHSD`.

Najprej preglejmo podatke, ki jih bomo uporabljali v nadaljevanju (ne le v tem podpodglavju).

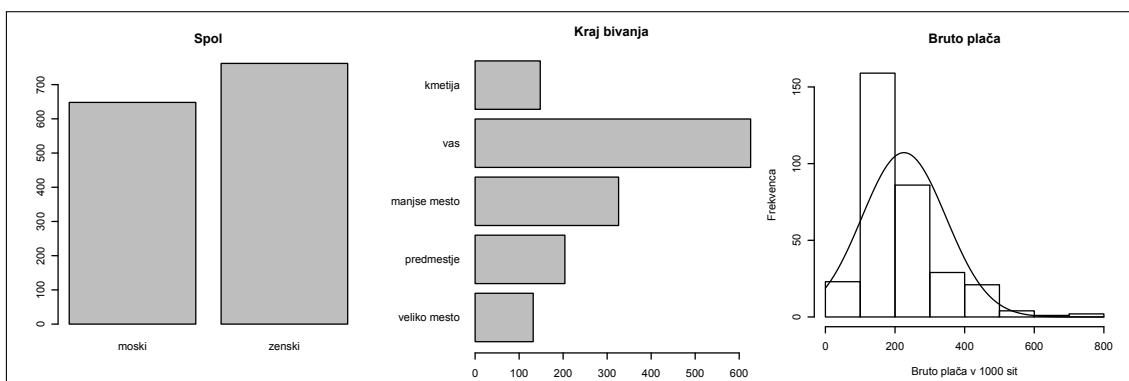
```
> #popravimo faktorje
> table(data$O1F2)
  moski zenski
   648   762
> table(data$F5)
veliko mesto  predmestje manjse mesto      vas
      132      204      326      626
  kmetija
    148
```

Spodaj je nekaj osnovnih izračunov. Porazdelitve osnovnih spremenljivk so prikazane na sliki 3.1.

```
> library(psych)
> describe(data$G91)
  vars  n  mean    sd median trimmed  mad min max
1    1 325 225.71 120.95   200  208.85 103.78  53 760
```

<sup>18</sup>Tudi če na primer ne moremo predpostavljati neodvisnosti med testi.

Slika 3.1: Porazdelitve uporabljenih spremenljivk



```

range skew kurtosis se
1 707 1.46 2.52 6.71
> frekTab(data$F5,dec=2)[,c(1,3)]
      Frekvenca  %
veliko mesto    132  9.19
predmestje     204 14.21
manjše mesto   326 22.70
vas            626 43.59
kmetija        148 10.31
> frekTab(data$O1F2,dec=2)[,c(1,3)]
      Frekvenca  %
moski         648 45.96
zenski        762 54.04
> if(!exists("mar.def")) mar.def<-par("mar")
> par(mfrow=c(1,3))
> plot(data$O1F2, main="Spol")
> par(mar=c(3,7,3,1))
> plot(data$F5,horiz=TRUE,las=1,main="Kraj bivanja")
> par(mar=mar.def)
> h<-hist(data$G91,main="Bruto plača",
  xlab="Bruto plača v 1000 sit", ylab="Frekvenca")
> curve(dnorm(x,mean=mean(data$G91,na.rm=TRUE),
  sd=sd(data$G91,na.rm=TRUE))*diff(h$breaks)[1]*
  sum(!is.na(data$G91)), add=TRUE)
> par(mfrow=c(1,1))

```

Sedaj izračunajmo še opisne statistike za bruto plačo po posameznih tipih krajev bivanja in narišemo ustrezne grafikone.

```

> describeBy(data$G91,group=data$F5,mat=TRUE)
  item      group1 vars   n   mean      sd median
11   1 veliko mesto   1  30 256.1000 133.22715   225
12   2 predmestje    1  45 220.9111  97.05313   200
13   3 manjse mesto   1  77 254.4675 129.83812   220
14   4 vas            1 139 213.8273 122.17005   188
15   5 kmetija        1  34 188.7059  97.12271   170
  trimmed      mad min max range      skew  kurtosis
11 253.2500 169.0164 75 450   375 0.2320304 -1.56727318
12 211.8919 100.8168 95 490   395 0.8192702 -0.03695316
13 237.4286 103.7820 100 750   650 1.3747886  1.80644481
14 196.1593  91.9212  53 760   707 1.7963112  4.18899970
15 172.8929  59.3040  80 528   448 1.7785525  3.18494875
  se
11 24.32384
12 14.46783
13 14.79643
14 10.36233
15 16.65641
> par(mfrow=c(2,3))
> for(i in levels(data$F5)){
  hist(data$G91[data$F5==i], xlab="Bruto plača", main=i,
      ylab="Frekvenca")
}
> par(mfrow=c(1,1))

```

Namesto histograma pa lahko s spodnjo kodo zelo enostavno narišemo tudi škatlaste grafikone po skupinah.

```

> plot(G91~F5,data=data)

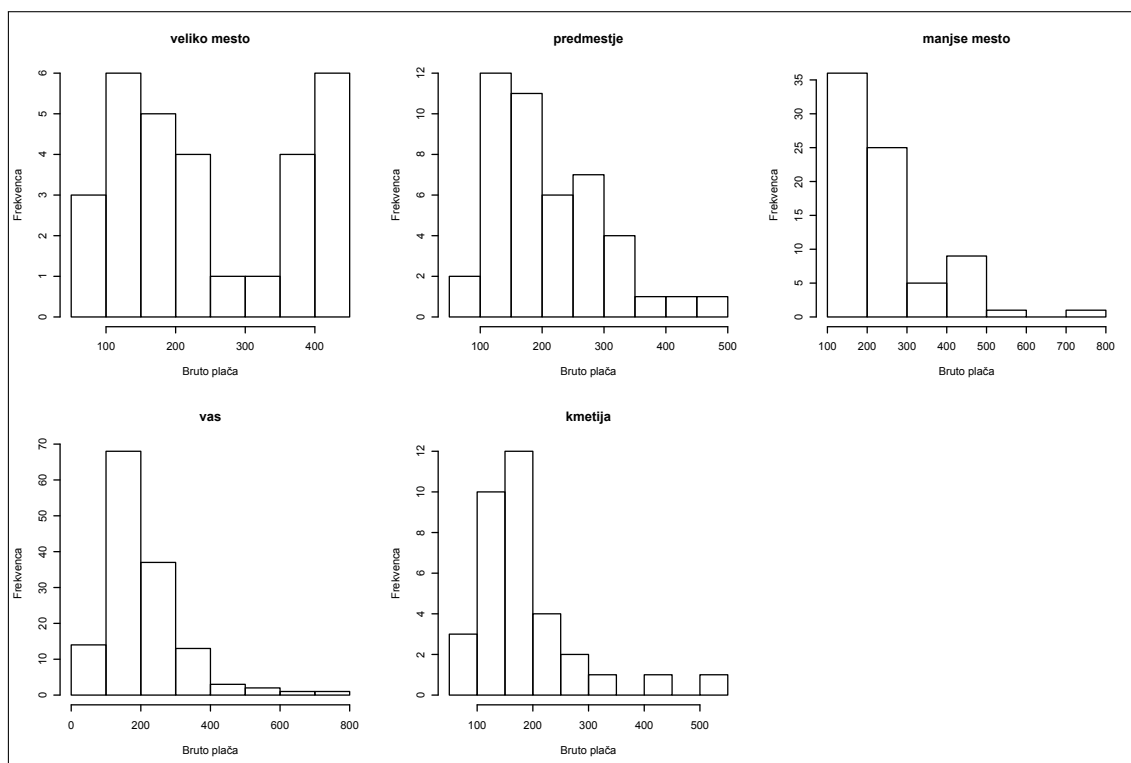
```

V posameznih kategorijah imamo razmeroma malo enot, vendar pa še vedno v vsaki nad 30. Morda bi bilo smiselno narediti manj kategorij, a glede na opisne statistike se kakšno zelo smiselno rekodiranje ravno ne ponuja.<sup>19</sup>

Iz opisnih statistik vidimo, da so standardni odkloni (koreni varianc) po skupinah razmeroma podobni. Tako lahko tudi brez testa sklepamo, da s predpostavko o enakosti varianc ne bi preveč zgrešili. Bolj pa je problematično, ker sta asimetrija in (še bolj) koničavost bistveno višji v zadnjih dveh kategorijah (vas in kmetija). To je razvidno tudi iz slike 3.2. Kljub temu bomo za demonstracijo izvedli tudi vse teste.

<sup>19</sup>Če bi kategorije združevali na podlagi opisnih statistik (predvsem aritmetičnih sredin), bi bilo to narobe. Test, ki bi ga naredili na podlagi tako združenih kategorij ne bi bil veljaven.

Slika 3.2: Porazdelitev bruto plače po krajih bivanja



```

> bartlett.test(G91~F5,data=data)
      Bartlett test of homogeneity of variances

data:  G91 by F5
Bartlett's K-squared = 7.7277, df = 4, p-value =
0.1021
> fligner.test(G91~F5,data=data)
      Fligner-Killeen test of homogeneity of variances

data:  G91 by F5
Fligner-Killeen:med chi-squared = 11.456, df = 4,
p-value = 0.02189
> #neobčutljiv na odstopanja od normalnosti
> library(car)
> leveneTest(G91~F5,data=data)
Levenes Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  4   1.789 0.1307
320

```

```

> #neobčutljiv na odstopanja od normalnosti
> leveneTest(G91~F5,data=data, center=mean)
Levenes Test for Homogeneity of Variance (center = mean)
      Df F value  Pr(>F)
group  4  2.2227 0.06639 .
      320
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
> #različica, kot jo uporablja SPSS
> #malce manj robustna

```

Barlettov in Levenov test pokažeta, da bi lahko domnevo o enakosti varianc zavrnili šele pri več kot 10-% tveganju. Nasprotno pa Fligner-Killeenjev test domnevo zavrne že pri 2.2-% tveganju. Glede na to, da same razlike med standardnimi odkloni niso tako velike, lahko predpostavko o enakosti varianc obdržimo. Kljub temu pa bomo (za demonstracijo) izvedli obe različici analize variance (klasično in Welchovo). Poleg tega bomo izvedli tudi neparametrični Kruskal-Wallisov test.

```

> #predpostavljamo enake variance
> oneway.test(G91~F5,data=data,var.equal=TRUE)
One-way analysis of means

data:  G91 and F5
F = 2.7695, num df = 4, denom df = 320, p-value =
0.02742
> #predpostavljamo različne variance
> #var.equal=FALSE bi lahko tudi izpustili, ker je to
> #privzeta možnost
> oneway.test(G91~F5,data=data,var.equal=FALSE)
One-way analysis of means (not assuming equal
variances)

data:  G91 and F5
F = 2.7964, num df = 4.00, denom df = 103.16, p-value
= 0.02988
> #še preko funkcije aov
> fit<-aov(G91~F5,data=data)
> summary(fit)
      Df  Sum Sq Mean Sq F value Pr(>F)
F5      4   158604   39651    2.77 0.0274 *
Residuals 320 4581392   14317
---

```

```

Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
1117 observations deleted due to missingness
> #rezultat je identičen kot pri prejšnji funkciji,
> #ko smo predpostavljali enake variance
>
> kruskal.test(G91~F5,data=data)
      Kruskal-Wallis rank sum test

data:  G91 by F5
Kruskal-Wallis chi-squared = 12.557, df = 4, p-value
= 0.01365

```

Obe različici ANOVE nam vrneti zelo podoben rezultat. Glede na oba lahko s tveganjem, manjšim kot 3 %, trdimo, da se povprečna bruto plača razlikuje glede na kraj bivanja. Glede na Kruskal-Wallisov test pa lahko predpostavko o enakosti srednjih vrednosti zavrnamo že pri 1.4-% tveganju.

### Opozorilo!

Domneva, ki jo preverjamo s Kruskal-Wallisovim testom ni več enaka.<sup>a</sup>

<sup>a</sup> Kruskal-Wallisov test preverja podobno domnevo kot Mann-Whitneyjev test, ki je podrobneje razložen v podpoglavju 2.3.3.

Poglejmo še, katera povprečja so različna. V primeru enakih varianc je za ta namen najprimernejša funkcija *TukeyHSD* (ki jo uporabimo na rezultatu funkcije *aov*), rezultat katere lahko tudi lepo narišemo. V primeru neenakih varianc pa uporabimo funkcijo *pairwise.t.test*.

```

> TukeyHSD(fit) #fit mora biti rezultat funkcije aov
      Tukey multiple comparisons of means
      95% family-wise confidence level

Fit: aov(formula = G91 ~ F5, data = data)

$F5
              diff            lwr            upr
predmestje-veliko mesto -35.188889 -112.55949  42.181712
manjse mesto-veliko mesto  -1.632468  -72.28018  69.015248
vas-veliko mesto          -42.272662 -108.35533  23.810003
kmetija-veliko mesto      -67.394118 -149.61878  14.830549
manjse mesto-predmestje   33.556421  -28.03788  95.150721
vas-predmestje            -7.083773  -63.38365  49.216108

```

```

kmetija-predmestje      -32.205229 -106.79513 42.384676
vas-manjse mesto       -40.640194  -87.27244  5.992052
kmetija-manjse mesto   -65.761650 -133.35265  1.829353
kmetija-vas            -25.121456  -87.92562 37.682705
      p adj
predmestje-veliko mesto 0.7232946
manjse mesto-veliko mesto 0.9999963
vas-veliko mesto       0.4019543
kmetija-veliko mesto   0.1647508
manjse mesto-predmestje 0.5667485
vas-predmestje        0.9969399
kmetija-predmestje     0.7603006
vas-manjse mesto       0.1204253
kmetija-manjse mesto   0.0608866
kmetija-vas            0.8078518
> par(mar=mar.def+c(0,8,0,0))
> plot(TukeyHSD(fit),las=1) #še grafična predstavitev rezultatov
> par(mar=mar.def)
> # TukeyHSD je primeren le ob predpostavki enakih varianc
>
> pairwise.t.test(x=data$G91,g=data$F5,p.adjust.method= "holm",
  pool.sd=TRUE)
      Pairwise comparisons using t tests with pooled SD

data:  data$G91 and data$F5

      veliko mesto predmestje manjse mesto vas
predmestje 1.00      -          -          -
manjse mesto 1.00    0.82      -          -
vas         0.56    1.00    0.16      -
kmetija     0.20    1.00    0.08    1.00

P value adjustment method: holm
> pairwise.t.test(x=data$G91,g=data$F5,p.adjust.method= "bonf",
  pool.sd=TRUE)
      Pairwise comparisons using t tests with pooled SD

data:  data$G91 and data$F5

      veliko mesto predmestje manjse mesto vas
predmestje 1.00      -          -          -
manjse mesto 1.00    1.00      -          -
vas         0.80    1.00    0.17      -

```

```

kmetija      0.25      1.00      0.08      1.00

P value adjustment method: bonferroni
> #če ne želimo uporabiti predpostavke o enakosti varianc
> #nastavimo pool.sd=FALSE
>
> pairwise.t.test(x=data$G91,g=data$F5,p.adjust.method= "holm",
  pool.sd=FALSE)
      Pairwise comparisons using t tests with non-pooled SD

data:  data$G91 and data$F5

      veliko mesto predmestje manjse mesto vas
predmestje  0.821      -      -      -
manjse mesto 1.000    0.754      -      -
vas          0.754    1.000    0.233      -
kmetija     0.233    0.754    0.041    0.821

P value adjustment method: holm

```

Če predpostavljamo enake variance, lahko najdemo statistično značilne razlike šele pri 10-% tveganju (oziroma 6–8-%) in še to le med manjšim mestom in kmetijo. Podobno je tudi, če privzamemo, da so variance različne, le da lahko v tem primeru to trditev postavimo že pri 4.1% tveganju.

Za konec povprečja še grafično predstavimo na sliki 3.3.

```

> library(gplots)
> plotmeans(G91~F5,data=data)

```

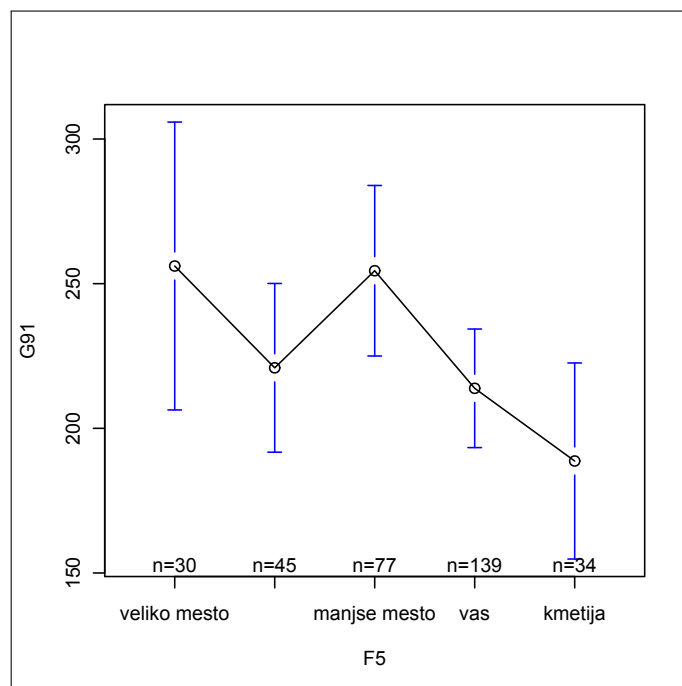
### 3.2.2 Večfaktorska analiza variance za neodvisne vzorce ★

Večfaktorska analiza variance je analiza variance, kjer so skupine določene z več kot enim faktorjem. Z njo torej preverjamo domnevo o vplivu dveh ali več nominalnih (oziroma vsaj tako jih obravnavamo) neodvisnih spremenljivk na odvisno spremenljivko.

Predpostavke večfaktorske analize variance so podobne predpostavkam enofaktorske – porazdelitev v vseh skupinah/celicah (določenih s kombinacijo vseh faktorjev oziroma neodvisnih spremenljivk) je normalna z enako varianco.

Analiza je bistveno enostavnejša, če so neodvisne spremenljivke med seboj neodvisne/nepovezane, kar pa se v družboslovju (oziroma kjerkoli, kjer podatke dobimo z

Slika 3.3: Porazdelitve uporabljenih spremenljivk



opazovanje/anketiranjem in ne eksperimentom) redko zgodi. V primeru, da neodvisne spremenljivke med seboj niso neodvisne, je pomemben vrstni red faktorjev. Model namreč najprej poskuša pojasniti čim več variabilnosti odvisne spremenljivke s 1. faktorjem, nato z 2., 3. ...

Rezultati se torej spremenijo, če zamenjamo vrstni red faktorjev. Od vrstnega reda je tudi odvisno, kaj nek test sploh preverja. Pri prvem faktorju test preverja, ali leta pojasni vsaj nekaj variabilnosti odvisne spremenljivke (oziroma ali vpliva nanjo). Pri vseh ostalih pa preverja, ali faktor pojasni kak del variabilnosti odvisne spremenljivke, ki še ni pojasnjen s faktorji, ki so v modelu pred obravnavanim faktorjem. Vse to velja, če uporabimo "klasično" vsoto kvadratov oziroma vsoto kvadratov tipa I. Obstajajo tudi druge vrste vsote kvadratov (tip II, III, IV, za več glejte Fox (2008, 159), Muenchen (2011, 633–635) in IBM (2011)), vendar jih tu ne bomo obravnavali. Vsote kvadratov tipa II in III lahko izračunate s pomočjo funkcije *Anova* iz paketa *car*. Pri tipu III vrstni red faktorjev ni pomemben.

Preverjamo lahko tudi, ali obstaja interakcija med vplivi posameznih faktorjev. Interakcija med vplivi faktorjev pomeni, da se vpliv nekega faktorja razlikuje glede na vrednosti nekega drugega faktorja.

Na našem primeru bomo preverjali, ali lahko trdimo, da kraj bivanja in spol vplivata na bruto plačo. Poleg tega bomo preverili tudi, ali lahko trdimo, da spol vpliva na bruto plačo, če predhodno izločimo vpliv kraja bivanja.

Pred izračunom bomo pripravili podatke in sicer tako, da bomo v nov podatkovni okvir shranili samo za to analizo potrebne spremenljivke. Tu bomo potem odstranili vse enote, ki imajo manjkajočo vrednost pri katerikoli spremenljivki. To bomo naredili tako, da bomo v nadaljevanju pri vseh analizah upoštevali iste enote (kar nam bo omogočalo, da bomo modele primerjali med seboj).

mnajprej torej pripravimo nov podatkovni okvir in odstranimo enote z manjkajočimi vrednostmi. Nato, podobno kot pri enofaktorski ANOVI, preverimo predpostavke (izračunamo opisne statistike po skupinah).

```
> dataAov<-data[,c("G91", "01F2", "F5")]
> dataAov<-na.omit(dataAov)
> #odstranili smo enote z manjkajočimi vrednostmi
>
> describeBy(dataAov$G91,group=dataAov[c("01F2", "F5")],
  mat=TRUE)
```

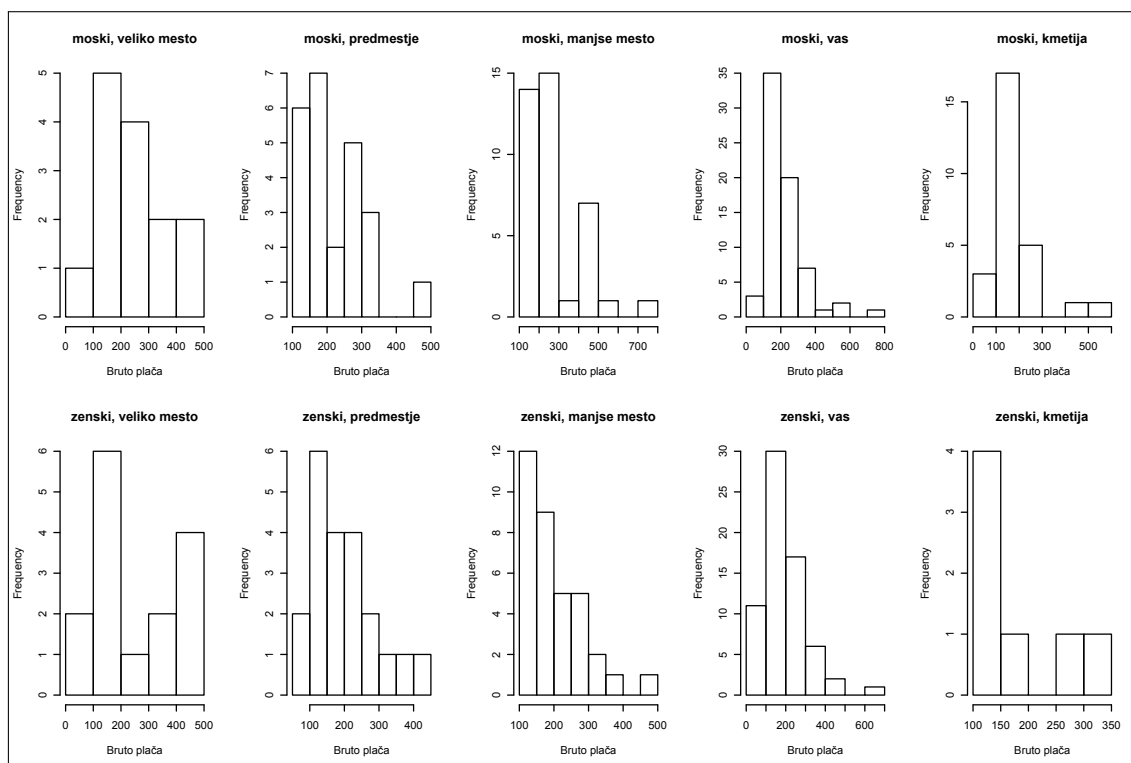
	item	group1	group2	vars	n	mean	sd
11	1	moski	veliko mesto	1	14	252.0000	127.94650
12	2	zenski	veliko mesto	1	15	252.3333	143.55172
13	3	moski	predmestje	1	24	231.1250	95.43917
14	4	zenski	predmestje	1	21	209.2381	99.89640
15	5	moski	manjse mesto	1	39	288.6923	147.34656
16	6	zenski	manjse mesto	1	35	210.1143	92.63679
17	7	moski	vas	1	69	227.0870	125.35992
18	8	zenski	vas	1	67	202.8060	120.44064
19	9	moski	kmetija	1	27	188.8889	98.64713
110	10	zenski	kmetija	1	7	188.0000	98.54441

```

  median trimmed      mad min max range      skew
11      226 250.2500 142.3296  75 450   375 0.3270494
12      200 250.3846 148.2600  80 450   370 0.2540330
13      200 223.3500 107.4885 120 490   370 0.7883115
14      184 197.0000  94.8864  95 450   355 0.8518656
15      250 274.1212  88.9560 110 750   640 1.1844391
16      170 199.2759  77.0952 100 500   400 1.1385801
17      200 208.9825  74.1300  61 760   699 1.9229792
18      180 185.9091 103.7820  53 700   647 1.5723492
19      170 172.3913  59.3040  80 528   448 1.9727940
110     132 188.0000  42.9954 103 350   247 0.6346745

  kurtosis      se
11 -1.42834380 34.19514
12 -1.78216305 37.06489
13 -0.02979449 19.48144
14 -0.23542599 21.79918
```

Slika 3.4: Porazdelitev bruto plače po krajih bivanja in spolu



```

15  0.78377390 23.59433
16  0.89652400 15.65848
17  4.53230346 15.09156
18  3.14171911 14.71417
19  3.93661300 18.98465
110 -1.56138214 37.24628
> par(mfrow=c(2,5))
> for(iGndr in levels(dataAov$01F2)){
  for(iF5 in levels(dataAov$F5)){
    hist(dataAov$G91[(dataAov$F5==iF5)&dataAov$01F2==iGndr],
        xlab="Bruto plača", main=paste(iGndr,iF5,sep=" "))
  }
}
> par(mfrow=c(1,1))

```

Sedaj je v nekaterih skupinah že zelo malo enot. Standardni odkloni med skupinami niso pretirano različni, se pa precej razlikujejo koeficienti asimetrije in sploščenosti. Vsekakor rezultati kažejo na to, da porazdelitev v vseh skupinah ni (niti približno) normalna in tudi ne podobna med skupinami. Predpostavke metode torej niso izpolnjene, zato moramo rezultate metode jemati z veliko rezervo.

Sedaj najprej ponovimo enofaktorsko ANOVO s funkcijo `aov` za vsak faktor posebej. Poleg tega preverimo tudi, ali sta neodvisni spremenljivki med seboj povezani.

```
> fitG91_F5<-aov(G91~F5,data=dataAov)
> summary(fitG91_F5)
              Df  Sum Sq Mean Sq F value Pr(>F)
F5              4  132286   33071   2.298 0.0589 .
Residuals     313 4503794   14389
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
> fitG91_01F2<-aov(G91~01F2,data=dataAov)
> summary(fitG91_01F2)
              Df  Sum Sq Mean Sq F value Pr(>F)
01F2            1   60436   60436   4.174 0.0419 *
Residuals     316 4575644   14480
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```

V primeru, da bi bile predpostavke klasične analize variance (normalnost, enake variance) izpolnjene (prej smo ugotovili, da so kršene), bi lahko pri 5.9-% tveganju trdili, da kraj bivanja vpliva na bruto plačo, in pri 4.2-% tveganju, da spol vpliva na bruto plačo.

Kot smo omenili, je pri večfaktorski analizi variance pomembno, ali sta neodvisni spremenljivki povezani. Preverimo torej to predpostavko in izračunajmo moč povezanosti.

```
> tbl<-table(dataAov$01F2,dataAov$F5)
> prop.table(tbl,margin=2)
              veliko mesto predmestje manjse mesto      vas
moski      0.4827586  0.5333333  0.5270270 0.5073529
zenski     0.5172414  0.4666667  0.4729730 0.4926471

              kmetija
moski  0.7941176
zenski 0.2058824
> #lažje opazimo, ali sta spremenljivki povezani
> chisq.test(tbl)
Pearsons Chi-squared test
```

```

data: tbl
X-squared = 9.8558, df = 4, p-value = 0.04293
> # install.packages("vcd")
> library(vcd)
> assocstats(tbl)

                X^2 df P(> X^2)
Likelihood Ratio 10.5712  4 0.031830
Pearson           9.8558  4 0.042928

Phi-Coefficient   : NA
Contingency Coeff.: 0.173
Cramers V         : 0.176
> #(pričakovana) zanimivost - na vseh podatkih povezanosti ni
> chisq.test(table(data$O1F2,data$F5))
      Pearsons Chi-squared test

data: table(data$O1F2, data$F5)
X-squared = 6.3402, df = 4, p-value = 0.1751

```

V naših podatkih (tistih, ki jih bomo uporabili pri analizi variance<sup>20</sup>) sta spremenljivki kraj bivanja in spol šibko povezani. To je posledica tega, da upoštevamo samo enote z veljavnimi vrednostmi za bruto plačo. Na vseh enotah namreč povezanosti ni oziroma povezanost ni statistično značilna.

Rekli smo, da bomo preverjali, ali lahko trdimo, da kraj bivanja in spol vplivata na bruto plačo. Test celotnega modela je malce zahtevnejši. Standardne funkcije ga ne izvedejo. Lahko pa ga izvedemo zelo enostavno, tako da model primerjamo z ničelnim modelom. Ničelni model je model, kjer nimamo pojasnjevalnih spremenljivk. V tem modelu so napovedi enake povprečju.

Poleg tega bomo preverili tudi, ali lahko trdimo, da spol vpliva na bruto plačo, če predhodno kontroliramo za vpliv kraja bivanja. Tako bomo v model kot 1. spremenljivko dali kraj bivanja, kot 2. pa spol.

```

> #izvedemo dvofaktorsko ANOVO
> fitG91_F5O1F2<-aov(G91~F5+O1F2,data=dataAov)
> summary(fitG91_F5O1F2)

              Df  Sum Sq Mean Sq F value Pr(>F)
F5              4  132286   33071   2.334 0.0556 .
O1F2            1   83242   83242   5.875 0.0159 *
Residuals     312 4420552   14168

```

<sup>20</sup>Izločili smo tiste enote, ki vsaj pri eni spremenljivki niso imele veljavne vrednosti, v veliki večini pri bruto plači.

```

---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
> #rezultat pri spolu nam pove, koliko ta model pojasni
> #več kot model s samo krajem bivanja
> anova(fitG91_F5,fitG91_F501F2)
Analysis of Variance Table

Model 1: G91 ~ F5
Model 2: G91 ~ F5 + 01F2
  Res.Df    RSS Df Sum of Sq    F Pr(>F)
1     313 4503794
2     312 4420552  1     83242 5.8752 0.01592 *
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
> #ocenimo "ničelni" model
> fitG91_1<-update(fitG91_F501F2,~1)
> #funkcija "update" popravi model glede na argumente
> #ali
> #fitG91_1<-aov(G91~1,data=dataAov)
>
> #primerjamo model z ničelnim,
> #da ocenimo statistično značilnost celotnega modela
> anova(fitG91_1,fitG91_F501F2)
Analysis of Variance Table

Model 1: G91 ~ 1
Model 2: G91 ~ F5 + 01F2
  Res.Df    RSS Df Sum of Sq    F Pr(>F)
1     317 4636080
2     312 4420552  5     215527 3.0424 0.01069 *
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

```

Iz rezultatov lahko razberemo:

- Pri 5.6-% tveganju lahko trdimo, da kraj bivanja vpliva na bruto plačo. Stopnja tveganja je manjša kot pri enofaktorski ANOVI (kjer je bila 5.9-%), ker je manjša nepojasnjena varianca (del variance je sedaj pojasnjen s spolom).
- Pri 1.6-% tveganju lahko trdimo, da spol vpliva na bruto plačo, če izločimo vpliv kraja bivanja. Ta stopnja tveganja je bistveno manjša kot prej. Pravzaprav je tudi pojasnjena vsota kvadratov (pri spolu) tu večja. To pomeni, da

so razlike v bruto plači med spoloma večje, če jih gledamo ločeno po krajih bivanja, kot če jih gledamo skupaj.

- S približno 1-% tveganjem lahko trdimo, da spol **ali** kraj bivanja (vsaj ena izmed teh dveh spremenljivk) vplivata na bruto plačo.

Preverimo še, ali obstaja tudi interakcija med vplivom spola in kraja bivanja. Ta "učinek" moramo v model vedno vključiti oziroma navesti na koncu.

```
> #izvedemo dvofaktorsko ANOVO
> fitG91_F501F2I<-aov(G91~F5*O1F2,data=dataAov)
> summary(fitG91_F501F2I)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
F5	4	132286	33071	2.334	0.0557 .
O1F2	1	83242	83242	5.874	0.0159 *
F5:O1F2	4	56064	14016	0.989	0.4136
Residuals	308	4364488	14170		

```
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
> #primerjamo model z ničelnim,
> #da ocenimo statistično značilnost celotnega modela
> anova(fitG91_1,fitG91_F501F2I)
```

Analysis of Variance Table

```
Model 1: G91 ~ 1
Model 2: G91 ~ F5 * O1F2
```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	317	4636080				
2	308	4364488	9	271591	2.1296	0.02692 *

```
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```

Interakcija med učinkoma ni statistično značilna. Ne moremo torej trditi, da spol drugače vpliva na bruto plačo v različnih krajih bivanja. Posledično se tudi skupna značilnost modela zaradi vključitve interakcije zniža.

### 3.2.3 Enofaktorska analiza variance za odvisne vzorce ★

Pogledali si bomo le najenostavnejši primer analize variance za odvisne vzorce. Analiza variance za odvisne vzorce je sorodna t-testu za odvisne vzorce, le da je vzorcev/spremenljivk, katerih povprečje preverjamo, več.

Analizo variance za odvisne vzorce lahko, prav tako kot prejšnje tipe, naredimo preko funkcije `aov` z ustreznimi argumenti, vendar pa moramo pred uporabo podatke prilagoditi. Podatke je treba preurediti v podobno obliko, kot jo imamo pri neodvisnih vzorcih. Potrebujemo torej eno odvisno spremenljivko in spremenljivko, ki "določa" vzorce (oziroma originalne spremenljivke). Poleg tega pa moramo za vsak podatek tudi vedeti, kateri enoti pripada. Za vsako enoto moramo imeti podatke za vse vzorce/spremenljivke.

Za to je uporabna tale funkcija<sup>21</sup>:

```
> razsiriPodatke<-function(X,id=rownames(X),
  varNames=colnames(X)){
  # funkcija za pretvorbo podatkov v obliko,
  # kot jo razumeta funkciji aov ali lm
  n12<-dim(X)
  n<-n12[1]
  m<-n12[2]
  if(is.null(varNames)) varNames<-1:m
  if(is.null(id)) id<-1:n

  res<-NULL
  for(i in 1:m){
    res<-rbind(res,data.frame(id=id,x=X[,i],
      var=varNames[i]))
  }
  return(res)
}
```

Funkcijo `aov` sicer uporabimo kot običajno, le da navedemo, katera spremenljivka nam "določa" enote, oziroma natančneje povemo, napake katerih "zapisov" so korelirane. **Pozor:** Klic funkcije je časovno zahteven.

Preverjali bomo domnevo, ali Slovenci enako zaupamo Državnemu zboru (spremenljivka B4), Evropskemu parlamentu (B9) in Združenim narodom (B10). Preglejmo najprej opisne statistike in pripravimo podatke.

```
> dataB4910<-na.omit(data[c("B4","B9","B10")])
> #za uporabo funkcije razsiriPodatke pripravimo
> #nov podatkovni okvir, ki vsebuje le uporabljene
> #spremenljivke
> describe(dataB4910)
```

<sup>21</sup>Funkcija se nahaja tudi v datoteki "UcbenikR-funkcije.R"

```

      vars      n mean   sd median trimmed  mad min max range
B4      1 1213 4.13 2.38      4   4.10 2.97   0 10   10
B9      2 1213 4.53 2.41      5   4.56 2.97   0 10   10
B10     3 1213 4.57 2.67      5   4.60 2.97   0 10   10
      skew kurtosis   se
B4   0.10    -0.57 0.07
B9  -0.08    -0.70 0.07
B10 -0.04    -0.82 0.08
> dataB4910raz<-razsiriPodatke(dataB4910)
> dataB4910raz[1:10,] #novi podatki izgledajo takole
  id x var
1   1 2 B4
2   2 8 B4
3   4 4 B4
4   5 8 B4
5   6 3 B4
6   7 3 B4
7   8 7 B4
8   9 3 B4
9  11 2 B4
10 14 5 B4

```

Zaupanje v Evropski parlament in Združene narode je bistveno višje kot v Državni zbor. Preverimo, ali je razlika statistično značilna.

```

> aovRep<-aov(x~var+Error(id),data=dataB4910raz)
> summary(aovRep)
Error: id
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 1212  17257    14.24

Error: Within
          Df Sum Sq Mean Sq F value  Pr(>F)
var         2    148   74.10  34.04 2.64e-15 ***
Residuals 2424   5277    2.18

---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
> model.tables(aovRep,type="means")
Tables of means
Grand mean

4.409453

```

var			
var			
	B4	B9	B10
	4.125	4.528	4.575

Pri zanemarljivi stopnji tveganja lahko trdimo, da Slovenci ne zaupamo vsem trem institucijam enako.

### 3.3 Linearna regresija

Z linearno regresijo ugotavljamo, če, in če kako, ena ali več neodvisnih spremenljivk vpliva(jo) na odvisno spremenljivko. To vključuje tudi ugotavljanje moči vpliva, ocenjevanje prilaganja modela oziroma pojasnjevalne moči modela, preverjane predpostavk in še kaj.

Kot že samo ime pove, se omejimo na *linearne* vplive spremenljivk, kar, če malce poenostavimo, pomeni, da se vpliv posamezne spremenljivke ne spreminja v odvisnosti od vrednosti odvisne spremenljivke. Obstaja sicer tudi nelinearna regresija, ki jo lahko izvedemo na več načinov. Pogosto tudi preko ustrezne transformacije spremenljivk nelinearno regresijo prevedemo v linearno regresijo. Na kratko se te teme dotaknemo v podpodpoglavju 3.3.3, sicer pa tema že presega obseg tega učbenika.

V učbeniku tudi ne obravnavamo naprednejših tem, kot so na primer posplošeni linearni modeli in hierarhični, večnivojski ali mešani modeli (glej Faraway 2006; Gelman in Hill 2006; Snijders in Bosker 2012). Prav tako obravnavamo samo podatke, zbrane v eni časovni točki, ne pa na primer panelnih podatkov (glejte na primer Wooldridge 2002).

Še vedno pa v tem podpoglavju obravnavamo precej široko področje, zato se tu še bolj kot v ostalih delih učbenika omejimo na izvedbo analize (linearne regresije) v **R**-ju, same teorije pa ne obravnavamo. Za pridobitev znanja s tega področja priporočam uporabo virov, ki so navedeni v podpodpoglavju 3.4, še posebej Fox (2008).

#### 3.3.1 Dodatne spremenljivke

Poleg spremenljivk, ki smo jih že uporabili pri analizi variance (glejte podpoglavje 3.2), bomo od tu naprej uporabljali še spremenljivki F7 "Število let šolanja" in F21 "Tipično število delovnih ur (vključno z nadurami) na teden". Grafični prikaz njunih porazdelitev je na sliki 3.5.

Pri tem so upoštevane samo enote, ki imajo veljavne vrednosti pri vseh spremenljivkah, ki jih bomo uporabili v regresiji (v kateremkoli modelu) – to so: G91, O1F2, F5,

F7 in F21. Na teh podatkih bomo opravili tudi vse linearne regresije. Ta pristop smo izbrali zato, da lahko rezultate različnih analiz med seboj primerjamo. Funkcija `lm`<sup>22</sup> namreč izvede analizo na enotah, ki imajo pri uporabljenih spremenljivkah veljavne vrednosti. Posledica tega je, da če izvajamo več različnih analiz z različnimi neodvisnimi spremenljivkami na originalnih podatkih, niso v vseh analizah uporabljene iste enote in zato rezultati niso primerljivi<sup>23</sup>.

V praksi je v fazi izbire modela tak pristop (če problem manjkajočih podatkov ni prevelik) priporočljiv, ko pa izberemo končni model (predvsem dokončno določimo neodvisne spremenljivke), pa je bolje, če upoštevamo vse razpoložljive podatke (torej vse enote, ki imajo veljavne vrednosti na spremenljivkah, ki jih uporabimo v nekem modelu).

V primeru manjkajočih vrednosti (torej takorekoč vedno) je najbolje, da upoštevamo vse veljavne vrednosti (torej tudi veljavne vrednosti pri enotah, ki imajo sicer tudi kakšno manjkajočo vrednost). Izmed dobrih pristopov se najpogosteje uporabljajo večkratne imputacije oz. multiple imputacije (Rubin 1987, 1996), ki so tudi zelo dobro podprte v paketku `mice` (Buuren in Groothuis-Oudshoorn 2011; Buuren 2012). Vendar pa to že presega obseg tega učbenika.

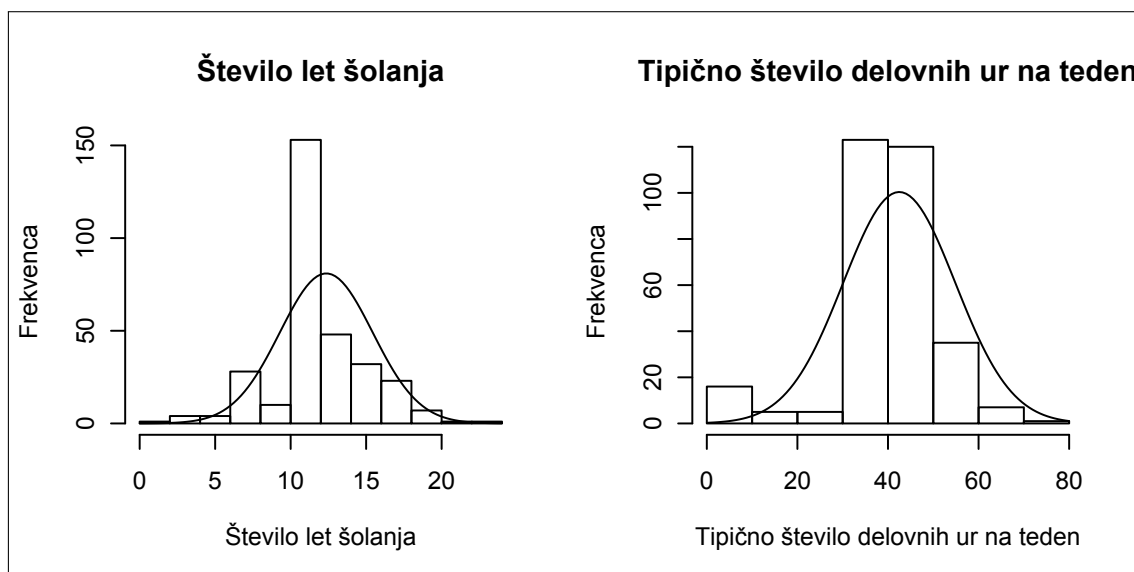
Pripravimo torej najprej podatke ter preglejmo opisne statistike in porazdelitvi za dodatni spremenljivki.

```
> dataLR<-na.omit(data[c("G91", "01F2", "F5", "F7", "F21")])
> describe(dataLR[c("F7", "F21")])
      vars  n mean   sd median trimmed  mad min max range
F7      1 312 12.36 3.08    12   12.32 1.48   1  23    22
F21     2 312 42.49 12.40    42   43.53 4.45   0  80    80
      skew kurtosis  se
F7     0.05     1.15 0.17
F21   -1.33     4.00 0.70
> par(mfrow=c(1,2),mar=mar.def)
> h<-hist(dataLR$F7,main="Število let šolanja",
  xlab="Število let šolanja", ylab="Frekvenca")
> curve(dnorm(x,mean=mean(dataLR$F7,na.rm=TRUE),
  sd=sd(dataLR$F7,na.rm=TRUE))*diff(h$breaks)[1]*
  sum(!is.na(dataLR$F7)), add=TRUE)
> h<-hist(dataLR$F21,main="Tipično število delovnih ur na teden",
  xlab="Tipično število delovnih ur na teden", ylab="Frekvenca")
```

<sup>22</sup>Enako velja za ostale **R**-jeve funkcije za regresijske metode (kar vključuje tudi analizo variance).

<sup>23</sup>Še posebej je to problem, če želimo modele formalno primerjati, na primer s funkcijo `anova` – glejte podpoglavje 3.3.5.

Slika 3.5: Porazdelitve dodatnih spremenljivk



```
> curve(dnorm(x, mean=mean(dataLR$F21, na.rm=TRUE),
  sd=sd(dataLR$F21, na.rm=TRUE))*diff(h$breaks)[1]*
  sum(!is.na(dataLR$F21)), add=TRUE)
> par(mfrow=c(1,1))
```

Opazimo lahko, da porazdelitev obeh spremenljivk ni normalna, še posebej pa od normalnosti odstopa porazdelitev spremenljivke "Tipično število delovnih ur na teden", ki je zelo koničasta in asimetrična v levo.

### 3.3.2 Bivariatna regresija

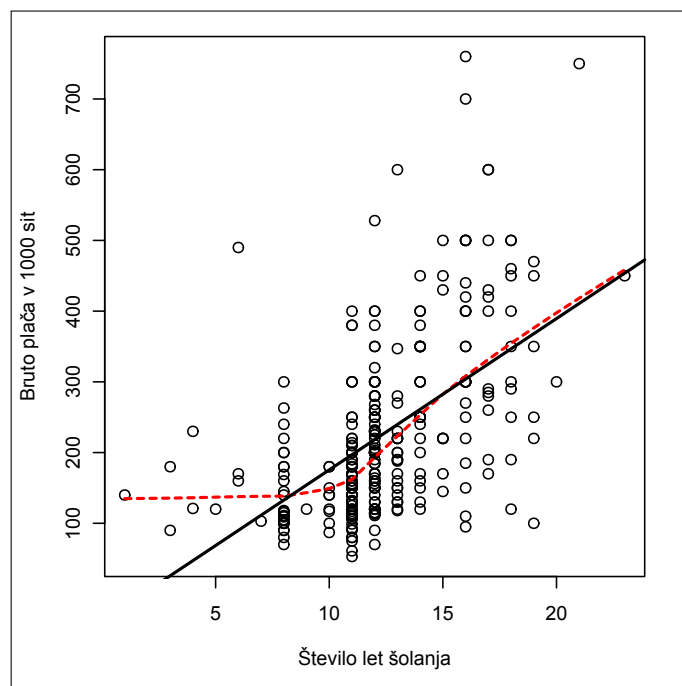
Začnimo z najenostavnejšim primerom bivariatne linearne regresije. Preverimo, kako število let šolanja vpliva na bruto plačo.

Odnos med spremenljivkama je grafično predstavljen na sliki 3.6. Za zdaj odmislimo polno črto/premico na tej sliki. Rdeča črtkana črta pa prikazuje glajena povprečja odvisne spremenljivke glede na vrednosti neodvisne.

```
> par(mar=mar.def+c(0,0,-3,0))
> plot(G91~F7,data=dataLR,ylab="Bruto plača v 1000 sit",
  xlab="Število let šolanja")
> lines(with(data=dataLR,lowess(G91~F7)),lwd=2,lty=2,col="red")
> par(mar=mar.def)
```

Iz slike lahko razberemo naslednje stvari:

Slika 3.6: Odnos med bruto plačo in številom let šolanja



- Če število let šolanja narašča, v povprečju narašča tudi bruto plača.
- Variabilnost bruto plače narašča s številom let šolanja, kar je kršitev ene izmed predpostavk linearne regresije.
- Videti je, da s številom let šolanja bruto plača najprej narašča počasi (ali sploh ne), kasneje pa narašča vse hitreje. Torej je mogoče, da linearna zveza ni primerna.
- Videti je, da začne izobrazba vplivati na plačo šele, ko število let šolanja preseže 12 let.

Kljub pomislekom nadaljujmo najprej z enostavno linearno regresijo. Za ocenjevanje linearne regresije uporabimo funkcijo `lm`, za preverjanje domnev o koeficientih in celotnega modela, za izračun determinacijskega ter nekaterih drugih statistih funkcijo `summary`, za izračun intervalov zaupanja pa funkcijo `confint` iz paketa `car`. Narišemo tudi sliko, ki prikazuje odvisnost bruto plače od števila let šolanja.

```
> (fitG91_F7<-lm(G91~F7,data=dataLR))
Call:
lm(formula = G91 ~ F7, data = dataLR)

Coefficients:
(Intercept)          F7
      -38.41         21.39
```

```

> summary(fitG91_F7) #za bolj bogat izpis
Call:
lm(formula = G91 ~ F7, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-268.09  -65.02  -18.33   47.86  456.09

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -38.41     24.07  -1.596   0.112
F7             21.39      1.89  11.320 <2e-16 ***
---
Signif. codes:
  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 102.5 on 310 degrees of freedom
Multiple R-squared:  0.2925,    Adjusted R-squared:  0.2902
F-statistic: 128.1 on 1 and 310 DF,  p-value: < 2.2e-16
> library(car)
> confint(fitG91_F7,level=0.9) #90% intervala zaupanja
              5 %      95 %
(Intercept) -78.12024  1.296065
F7           18.27676  24.512959
> par(mar=mar.def+c(0,0,-3,0))
> plot(G91~F7,data=dataLR,ylab="Bruto plača v 1000 sit",
      xlab="Število let šolanja")
> lines(with(data=dataLR,lowess(G91~F7)),lwd=2,lty=2,col="red")
> abline(fitG91_F7,lwd=2)
> par(mar=mar.def)

```

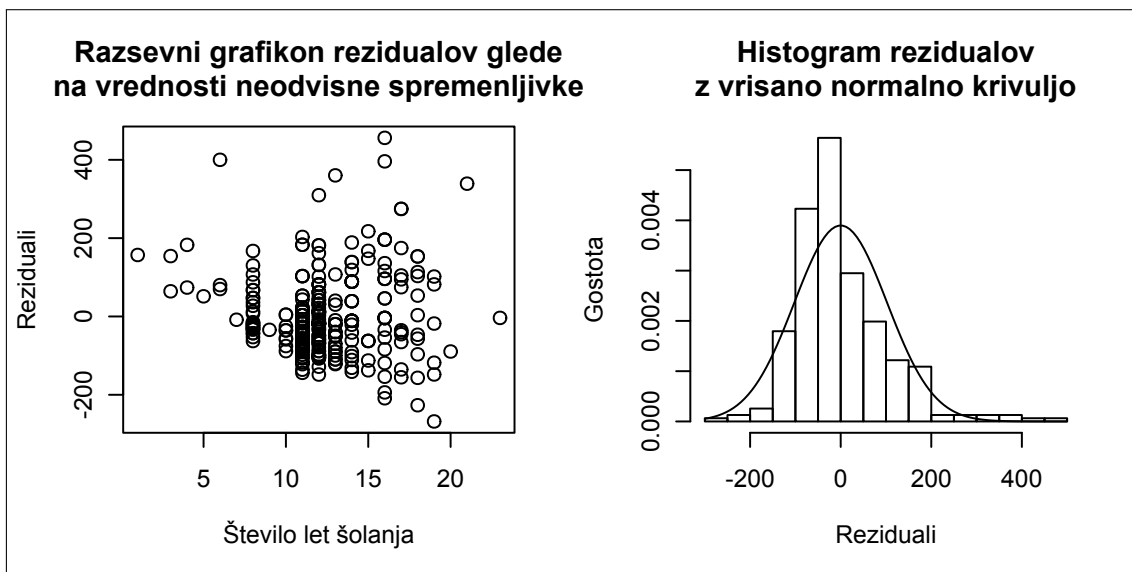
Z variabilnostjo v številu let šolanja lahko pojasnimo 29.2 % variabilnosti v bruto plači. Če se število let šolanja poveča za eno leto, se bo bruto plača v povprečju povečala za 21.4 tisoč sit. Vpliv števila let šolanja na bruto plačo je statično značilen pri zanemarljivi stopnji tveganja.

S pomočjo funkcije `confint` iz paketka `car` smo izračunali tudi 90-% intervala zaupanja za regresijski koeficient in konstanto.

Na podlagi rezultata smo na sliko 3.6 vrisali premico, ki prikazuje ocenjeni odnos med spremenljivkama. Očitno je, da se premica pri velikih in majhnih vrednostih spremenljivke "Število let šolanja" ne prilega.

To je morda še bolj očitno, če si pogledamo graf, kjer na  $y$  os nanašamo rezidualne, na  $x$  os pa vrednosti neodvisne spremenljivke (1. graf na sliki 3.7). Histogram na

Slika 3.7: Porazdelitev rezidualov



sliki 3.7 pa kaže, da je kršena tudi predpostavka o normalni porazdelitvi rezidualov/napak. Koda za sliko sledi.

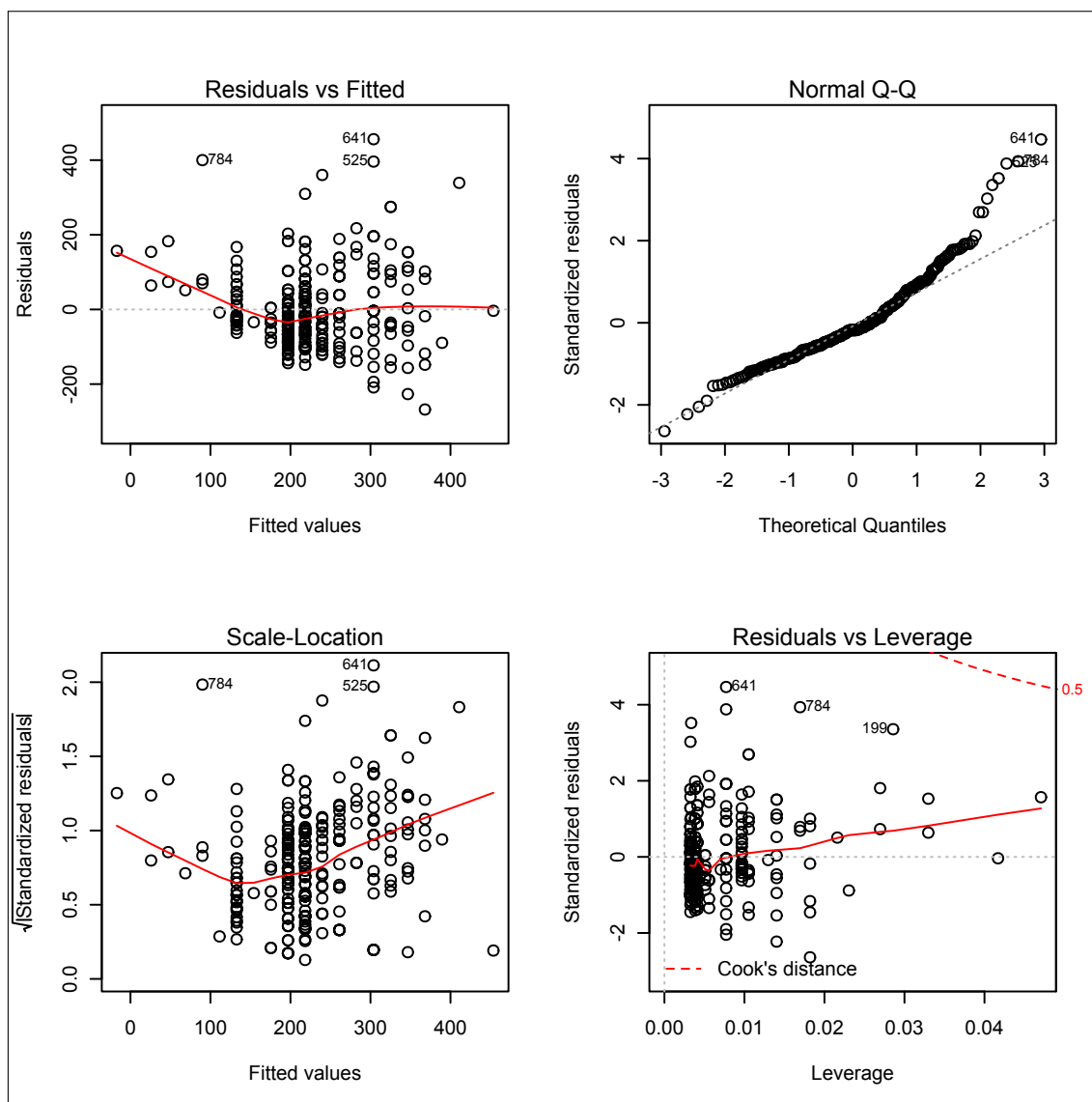
```
> par(mfrow=c(1,2),xpd=NA)
> plot(fitG91_F7$residuals~dataLR$F7,
      ylab="Reziduali", xlab="Število let šolanja",
      main="Razsevni grafikon rezidualov glede
      na vrednosti neodvisne spremenljivke")
> hist(fitG91_F7$residuals, freq=FALSE, br=15,
      xlab="Reziduali", ylab="Gostota",
      main="Histogram rezidualov\nz vrisano normalno krivuljo")
> curve(dnorm(x,mean=0,sd=sd(fitG91_F7$residuals)),add=TRUE)
> par(mfrow=c(1,1),xpd=FALSE)
```

Podobne zaključke lahko izpeljemo tudi na podlagi grafikonov na sliki 3.8, ki so (v **R**-ju) standardni diagnostični prikazi za linearno regresijo (tudi multiplo), zato jih dobimo zelo enostavno.

```
> par(mfrow=c(2,2)) #zahtevamo 4 grafe na eno sliko
> plot(fitG91_F7)
> par(mfrow=c(1,1))
```

Graf levo zgoraj je podoben razsevniemu grafikonu na sliki 3.7, le da so na  $y$  osi napovedane vrednosti in ne neodvisna spremenljivka. Pri bivariatni regresiji gre pravzaprav le za linearno transformacijo skale (množenje in seštevanje), pri multipli

Slika 3.8: Diagnostični grafikoni za linearno regresijo



pa služijo napovedane vrednosti kot nek nadomestek za vse neodvisne spremenljivke (saj so napovedane vrednosti linearne kombinacije neodvisnih spremenljivk).

Graf desno zgoraj pa podaja podobno informacijo kot histogram. Imenuje se Q-Q plot (Q je okrajšava za *quantile*). Če so reziduali normalno porazdeljeni, ležijo vse točke na črtkani premici.

### 3.3.3 Nelinearna regresija ★

Ugotovili smo, da linearna zveza med spremenljivkama na danih podatkih morda ni najbolj optimalna. Odnos je videti dosti bolj linearen, če bruto plačo logaritmujemo.

Na takih podatkih lahko ocenimo tudi linearno regresijo, in sicer tako, da znotraj formule bruto plačo logaritmiramo. Sicer se koda skorajda ne spremeni. Rezultat lahko potem narišemo tudi na originalni lestvici. Oboje je prikazano na sliki 3.9).

```

> fitLnG91_F7<-lm(log(G91)~F7,data=dataLR)
> summary(fitLnG91_F7) #za bolj bogat izpis
Call:
lm(formula = log(G91) ~ F7, data = dataLR)

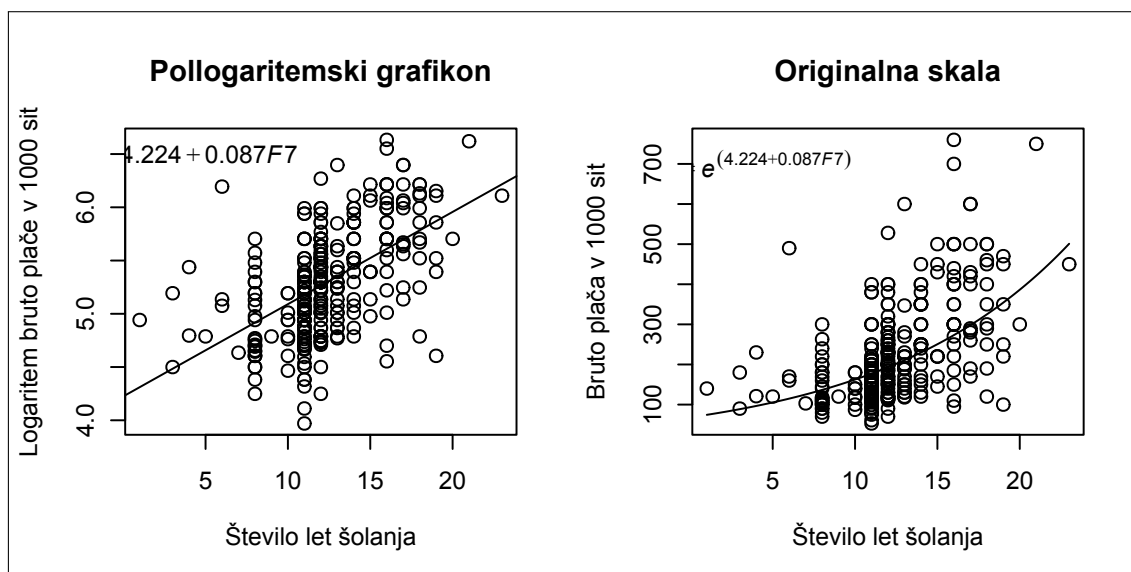
Residuals:
    Min       1Q   Median       3Q      Max
-1.26487 -0.29420  0.02024  0.27755  1.45055

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 4.224075   0.098354   42.95  <2e-16 ***
F7           0.086630   0.007723   11.22  <2e-16 ***
---
Signif. codes:
  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 0.4189 on 310 degrees of freedom
Multiple R-squared:  0.2887,    Adjusted R-squared:  0.2864
F-statistic: 125.8 on 1 and 310 DF,  p-value: < 2.2e-16
> par(mfrow=c(1,2))
> plot(log(G91)~F7,data=dataLR,xlab="Število let šolanja",
      ylab="Logaritem bruto plače v 1000 sit",
      main="Pollogaritemski grafikon")
> abline(fitLnG91_F7)
> text(x=2.5,y=6.5,
      labels=bquote(italic(G91)*minute==
        .(round(coef(fitLnG91_F7)[1],3))
        +.(round(coef(fitLnG91_F7)[2],3))*italic(F7)))
> plot(G91~F7,data=dataLR,ylab="Bruto plača v 1000 sit",
      xlab="Število let šolanja",
      main="Originalna skala")
> curve(exp(coef(fitLnG91_F7)[1]+coef(fitLnG91_F7)[2]*x),
      add=TRUE)
> text(x=2.5,y=700,
      labels=bquote(italic(G91)*minute==
        italic(e)^(.(round(coef(fitLnG91_F7)[1],3))
        +.(round(coef(fitLnG91_F7)[2],3))*italic(F7))))
> par(mfrow=c(1,1))

```

Slika 3.9: Exponentna zveza – transformacija



Ob tem pa se je treba zavedati, da kadarkoli transformiramo odvisno spremenljivko, izračuni niso več pravilni oziroma optimalni. V tem primeru namreč ne minimiziramo več vsote kvadratov odklonov originalne, ampak transformirane spremenljivke (linearna regresija za ocenjevanje uporablja metodo najmanjših kvadratov). Zaradi istega razloga tudi statistike  $R^2$  in podobne niso pravilne. Pravilen izračun za  $R^2$  bi bil:

```
> resLog<-dataLR$G91- exp(fitLnG91_F7$fitted)
> odkloniPov<-dataLR$G91- mean(dataLR$G91)
> R2log<-1-sum(resLog^2)/sum(odkloniPov^2)
> R2log
[1] 0.2825734
```

Dejanski  $R^2$  je torej še nižji. Sedaj smo sicer pravilno izračunali  $R^2$ , a še vedno je ocenjevanje parametrov neoptimalno. Parametre bi namreč morali ocenjevati tako, da bi minimizirali vsoto kvadratov odklonov originalne spremenljivke.

To lahko naredimo takole:

```
> expZveza<-function(b,y,X){
  X<-as.matrix(X)
  ss<-sum((y - exp(cbind(1,X)%*%b))^2)
  return(ss)
}
> optExp<-optim(par=coef(fitLnG91_F7),
  fn=expZveza,y=dataLR$G91,X=dataLR[c("F7")])
```

```

> resOptim<-dataLR$G91 -
      exp(optExp$par[1] + optExp$par[2]*dataLR$F7)
> odkloniPov<-dataLR$G91-mean(dataLR$G91)
> R2optim<-1-sum(resOptim^2)/sum(odkloniPov^2)
> R2optim
[1] 0.316329
> R2log
[1] 0.2825734
> summary(fitG91_F7)$r.sq
[1] 0.2924672
> plot(G91~F7,data=dataLR,ylab="Bruto plača v 1000 sit",
      xlab="Število let šolanja")
> curve(exp(optExp$par[1]+optExp$par[2]*x),add=TRUE,col="blue")
> curve(exp(coef(fitLnG91_F7)[1]+coef(fitLnG91_F7)[2]*x),
      add=TRUE,col="red")
> abline(fitG91_F7,col="green3")
> text(x=19,y=650,
      labels=bquote(italic(G91)*minute==
        italic(e)^((round(optExp$par[1],3)) +
          .(round(optExp$par[2],3))*italic(F7))),col="blue")
> legend(x=12,y=790,legend=c("linearna zveza",
  "eksponentna zveza preko transformacije",
  "eksponentna zveza preko optimizacije"),
  col=c("green3","red","blue"),lty=1,xjust=0.5,yjust=0,
  xpd=TRUE)

```

Rezultat ocenjevanja je prikazan na sliki 3.10.  $R^2$  je sedaj znatno večji kot pri kateremkoli prejšnjem pristopu in znaša 0.316.

### 3.3.4 Multipla regresija

Sedaj razširimo naš model tako, da vključimo še eno intervalno neodvisno spremenljivko, in sicer F21 "Tipično število delovnih ur (vključno z nadurami) na teden".

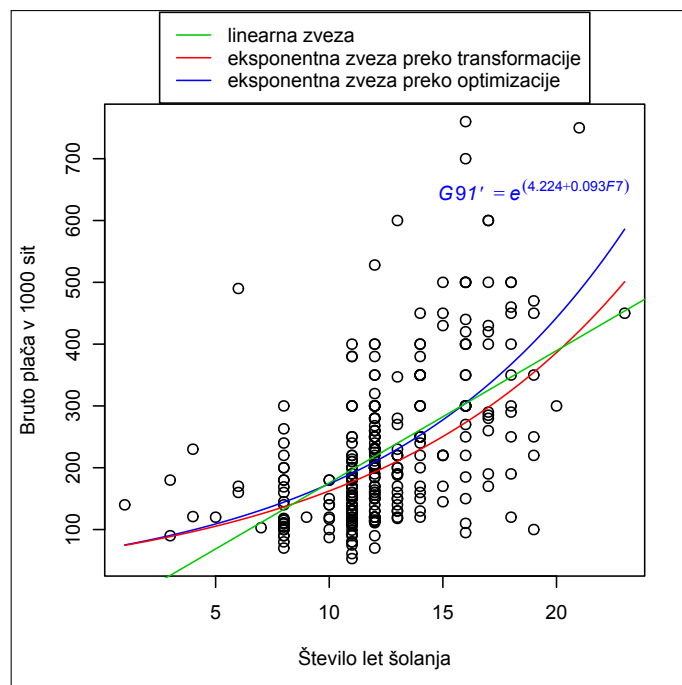
Tudi za to spremenljivko je pred vključitvijo dobro preveriti obliko njene povezanosti z odvisno spremenljivko. Ustrezni prikaz je na sliki 3.11.

```

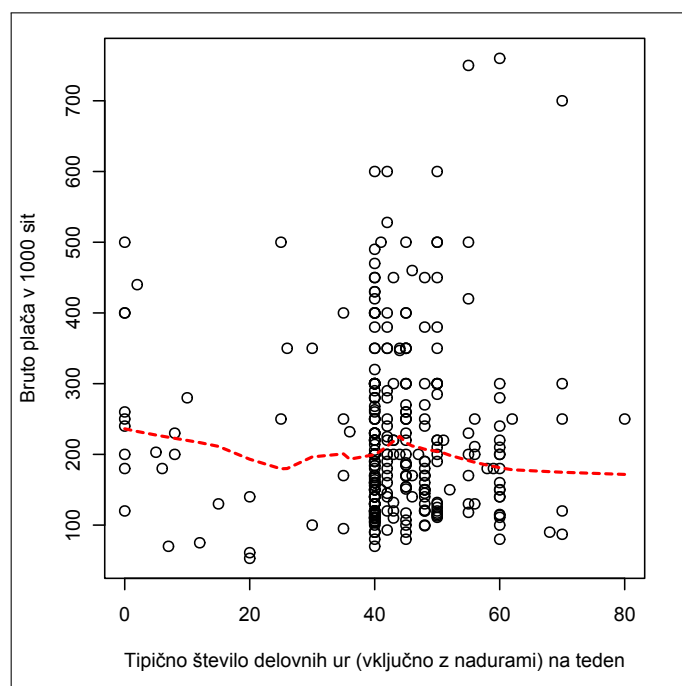
> par(mar=mar.def+c(0,0,-3,0))
> plot(G91~F21,data=dataLR,ylab="Bruto plača v 1000 sit",xlab=
  "Tipično število delovnih ur (vključno z nadurami) na teden")
> lines(with(data=dataLR,lowess(G91~F21)),lwd=2,lty=2,col="red")
> par(mar=mar.def)

```

Slika 3.10: Ocenjevanje nelinearne zveze



Slika 3.11: Odnos med bruto plačo in tipičnim številom delovnih ur na teden



Kakšne posebne povezanosti med spremenljivkama ne vidimo. Ker pa želimo to tudi formalno preveriti, bomo spremenljivko o številu delovnih ur vseeno vključili v linearno regresijo. Preverili bomo torej, ali spremenljivka "Tipično število delovnih ur (vključno z nadurami) na teden" vpliva na bruto plačo, če izločimo vpliv izobrazbe (spremenljivka "Število let šolanja"). Zanima nas torej, ali se bruto plača v povprečju kaj spremeni, če se število delovnih ur na teden poveča, izobrazba pa ostane nespremenjena.

Pri linearni regresiji vrstni red vključitve ni pomemben.

Kljub temu da bi bila glede na naše podatke primernejša nelinearna regresija, bomo (iz pedagoških razlogov) nadaljevali z linearno. Enostaven popravek bi bil, če bi samo zamenjali G91 (bruto plačo) z logaritmom te spremenljivke, kar sicer (kot smo videli) ni optimalno.

```
> fitG91_F7F21<-update(fitG91_F7,~.+F21)
> #prejšnjemu modelu dodamo dodatno neodvisno spremenljivko
> #ali dalje
> #fitG91_F7F21<-lm(G91~F7+F21,data=dataLR)
> summary(fitG91_F7F21)
Call:
lm(formula = G91 ~ F7 + F21, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-265.76  -64.89  -18.66   47.83  458.44

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -32.7354     31.1367  -1.051   0.294
F7           21.4004     1.8929  11.306 <2e-16 ***
F21          -0.1352     0.4695  -0.288   0.774
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 102.7 on 309 degrees of freedom
Multiple R-squared:  0.2927,    Adjusted R-squared:  0.2881
F-statistic: 63.92 on 2 and 309 DF,  p-value: < 2.2e-16
> confint(fitG91_F7F21,level=0.9)
              5 %      95 %
(Intercept) -84.1046492 18.6338373
```

```

F7          18.2775218 24.5233578
F21         -0.9098706 0.6394045
> summary(fitG91_F7)
Call:
lm(formula = G91 ~ F7, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-268.09  -65.02  -18.33   47.86  456.09

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   -38.41      24.07  -1.596   0.112
F7             21.39       1.89  11.320 <2e-16 ***
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 102.5 on 310 degrees of freedom
Multiple R-squared:  0.2925,    Adjusted R-squared:  0.2902
F-statistic: 128.1 on 1 and 310 DF,  p-value: < 2.2e-16

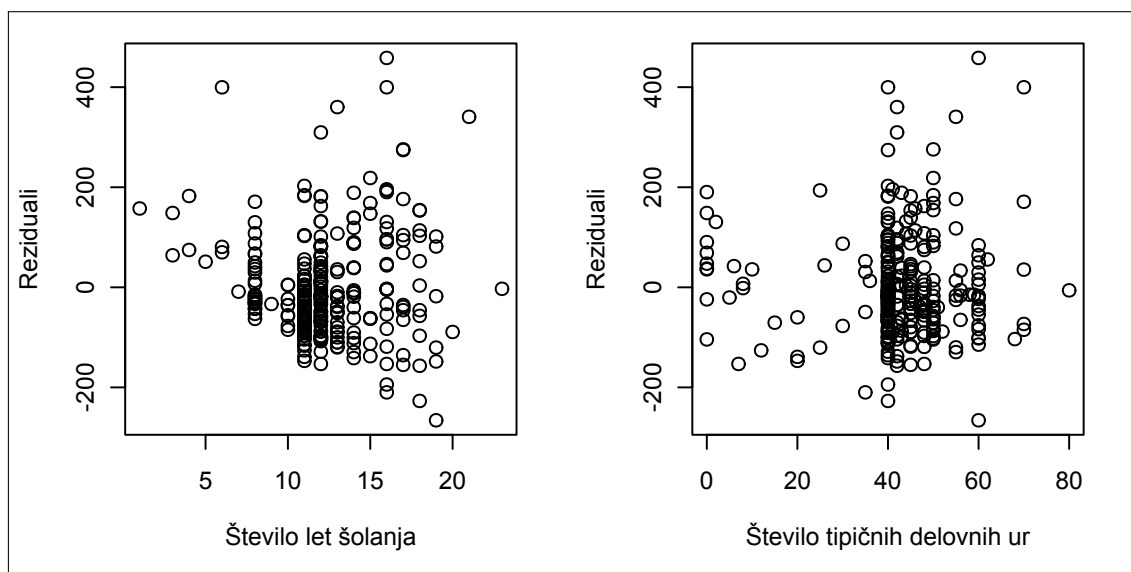
```

Za primerjavo smo izpisali še rezultat modela brez spremenljivke F21. Vpliv spremenljivke F21 (kot smo pričakovali glede na sliko 3.11) ni statistično značilen. Posledično tudi njena vključitev ne vpliva bistveno na rezultate ( $R^2$  se je sicer malce povečal, popravljeni  $R^2$  pa malce zmanjšal, vpliv spremenljivke F7 na G91 pa je ostal skoraj nespremenjen). Vseeno pa je interpretacija regresijskih koeficientov pri multipli regresiji malce drugačna kot pri bivariatni regresiji. Regresijski koeficient za število let šolanja 21.4 na primer pomeni, da se bruto plača poveča za 21.4 tisoč sit, če se število let šolanja poveča za eno leto in ostane tipično število delovnih ur na teden nespremenjeno, oziroma če ostanejo vrednosti vseh ostalih neodvisnih spremenljivk (tu imamo pač samo eno) nespremenjene.

Pri multipli regresiji je pomemben rezultat tudi  $F$  statistika in pripadajoči  $F$ -test. Ta nam v našem primeru pove, da lahko pri zanemarljivi stopnji tveganja trdimo, da vsaj ena izmed neodvisnih spremenljivk vpliva na odvisno spremenljivko.

Tudi tu bi bilo dobro preveriti porazdelitev rezidualov. To storimo podobno kot pri bivariatni regresiji. Ker pa imamo sedaj dve neodvisni spremenljivki, bi morali narisati dva razsevana grafikona (vsakega za svojo neodvisno spremenljivko na  $x$  osi). Taka grafa sta prikazana na sliki 3.12. Tako kot pri bivariatni regresiji se na prvem grafu vidi heteroskedastičnost (variabilnost napak ni povsod enaka), na drugem pa bi težko z gotovostjo prepoznali nek vzorec (na sredini osi je  $x$  sicer variabilnost videti največja, a je tam tudi največ vrednosti).

Slika 3.12: Reziduali v odvisnosti od vrednosti neodvisnih spremenljivk



```

> par(mfrow=c(1,2),mar=mar.def-c(0,0,3,0))
> plot(fitG91_F7F21$residuals~dataLR$F7,
      ylab="Reziduali", xlab="Število let šolanja",
      main="")
> plot(fitG91_F7F21$residuals~dataLR$F21,
      ylab="Reziduali", xlab="Število tipičnih delovnih ur",
      main="")
> par(mfrow=c(1,1),mar=mar.def)

```

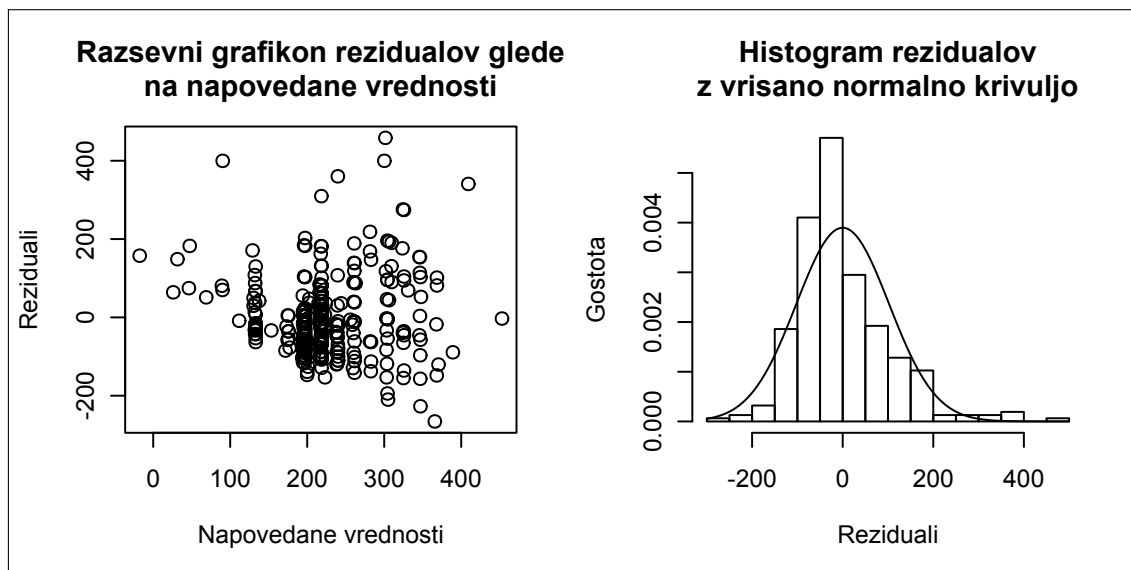
Ker pa je takih grafov pri multipli regresiji lahko veliko, je bolj praktično, da narišemo samo en graf, kjer na  $y$  os nanašamo napovedane vrednosti in ne neodvisnih spremenljivk. Kot smo že omenili, tako napovedane vrednosti služijo kot nadomestek za vse neodvisne spremenljivke (saj so napovedane vrednosti linearne kombinacije neodvisnih spremenljivk). Tak graf je skupaj s histogramom rezidualov prikazan na sliki 3.13.

```

> par(mfrow=c(1,2))
> plot(fitG91_F7F21$residuals~fitG91_F7F21$fitted,
      ylab="Reziduali", xlab="Napovedane vrednosti",
      main="Razsevni grafikon rezidualov glede
      na napovedane vrednosti")
> hist(fitG91_F7F21$residuals, freq=FALSE, br=15,
      xlab="Reziduali", ylab="Gostota",
      main="Histogram rezidualov\nz vrisano normalno krivuljo")

```

Slika 3.13: Porazdelitev rezidualov



```
> curve(dnorm(x,mean=0,sd=sd(fitG91_F7F21$residuals)),add=TRUE)
> par(mfrow=c(1,1))
```

R-jevi standardni diagnostični prikazi za linearno regresijo so prikazani na sliki 3.14.

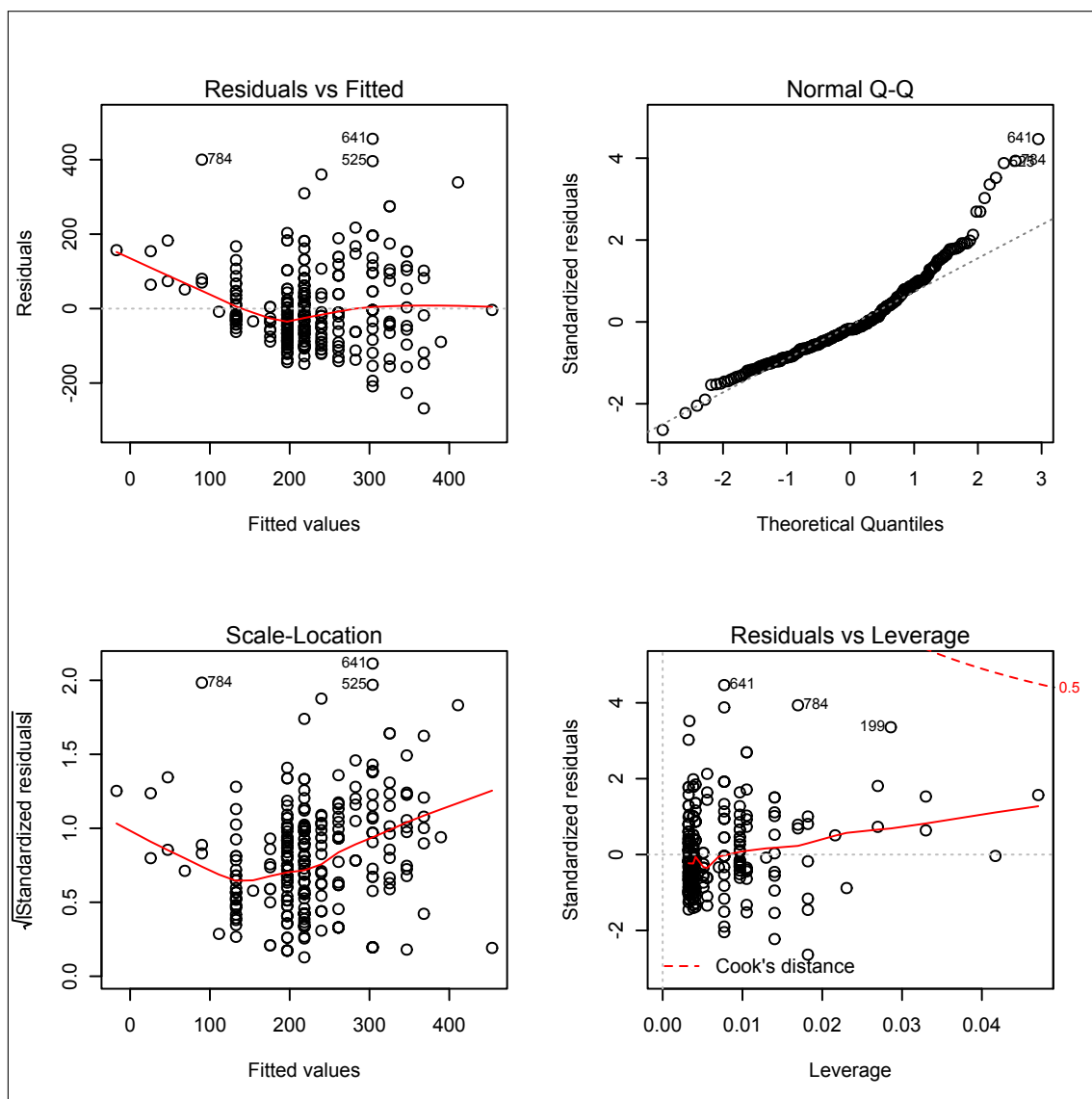
```
> par(mfrow=c(2,2))
> plot(fitG91_F7)
> par(mfrow=c(1,1))
```

### 3.3.5 Vključevanje nominalnih/ordinalnih spremenljivk

V linearno regresijo lahko kot neodvisne spremenljivke vključimo tudi nominalne spremenljivke, in sicer preko umetnih spremenljivk. Umetnih spremenljivk nam ni treba ustvariti, saj jih samodejno ustvari funkcija *lm*. Kot referenčno kategorijo izbere tisto, ki je pri faktorju navedena kot prva v *levels*.

Funkcija *lm* kot nominalne spremenljivke obravnava le spremenljivke tipa *factor* in *character*, ki pa jih pred uporabo spremeni v tip *factor*.

Slika 3.14: Diagnostični grafkoni za linearno regresijo

**Opozorilo!**

Funkcija *lm* obravnava spremenljivke tipa *ordered* oziroma urejen *factor* tako, da za njih na poseben način izračuna – "kontraste". Tega načina ne bomo obravnavali, saj je razmeroma zahteven za interpretacijo in razumevanje, zato je priporočljivo, da tudi za ordinalne spremenljivke uporabljate **neurejen factor**.

Poglejmo najprej najenostavnejši primer, ko ima nominalna spremenljivka samo 2 vrednosti. V naš model torej vključimo še spol.

```

> fitG91_F7F2101F2<-lm(G91~F7+F21+01F2,data=dataLR)
> summary(fitG91_F7F2101F2)
Call:
lm(formula = G91 ~ F7 + F21 + 01F2, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-247.70  -69.39  -17.86   52.78  444.16

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.0737    31.2701  -0.546  0.58546
F7           21.6213     1.8731  11.543 < 2e-16 ***
F21          -0.2170     0.4651  -0.467  0.64108
01F2zenski  -33.0047    11.5776  -2.851  0.00466 **
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 101.5 on 308 degrees of freedom
Multiple R-squared:  0.3108,    Adjusted R-squared:  0.3041
F-statistic: 46.31 on 3 and 308 DF,  p-value: < 2.2e-16

```

Med rezultati je dodaten regresijski koeficient "01F2Ženski". Iz njega vidimo, da je bila kot referenčna kategorija vzeta kategorija "Moški", saj te ni med koeficienti. Vrednost tega koeficienta ( $-33.0$ ) nam pove, da imajo ženske v povprečju pri enakih vrednostih ostalih spremenljivk (izobrazba, število delovnih ur na teden) za 33 tisoč sit nižjo plačo kot moški. Pri takem kodiranju je vrednost regresijskega koeficienta vedno primerjava izbrane kategorije z referenčno. Vpliv spola je statistično značilen pri tveganju 0.47 %.

### Opozorilo!

Klasična interpretacija, da se odvisna spremenljivka spremeni za  $b_x$ , če se  $x$  poveča za eno enoto in ostale spremenljivke ostanejo nespremenjene, tu ni primerna, ker se spol pri neki osebi načeloma ne spreminja.

Dodajmo kot spremenljivko še kraj bivanja.

```

> fitG91_F7F2101F2F5<-lm(G91~F7+F21+01F2+F5,data=dataLR)
> summary(fitG91_F7F2101F2F5)
Call:
lm(formula = G91 ~ F7 + F21 + 01F2 + F5, data = dataLR)

```

```

Residuals:
    Min       1Q   Median       3Q      Max
-248.20  -66.53  -17.92   49.71  449.09

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -10.5263    37.5909  -0.280  0.77965
F7            21.4129     1.9568  10.943 < 2e-16 ***
F21          -0.2492     0.4765  -0.523  0.60132
O1F2zenski  -33.1629    11.7890  -2.813  0.00523 **
F5predmestje -15.8647    24.6244  -0.644  0.51989
F5manjse mesto 11.4633    22.5129   0.509  0.61099
F5vas        -6.2163    21.2181  -0.293  0.76974
F5kmetija    -3.2327    26.8104  -0.121  0.90411
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 101.8 on 304 degrees of freedom
Multiple R-squared:  0.3159,    Adjusted R-squared:  0.3002
F-statistic: 20.06 on 7 and 304 DF,  p-value: < 2.2e-16

```

Ker je bilo "Veliko mesto" izbrano kot referenčna kategorija, imamo v izpisu regresijske koeficiente za vse ostale kategorije. Vsak izmed teh regresijskih koeficientov nam poda primerjavo med kategorijo in velikim mestom. Tako nam vrednost koeficienta "F5Vas"  $-6.2$  pove, da imajo osebe, ki živijo na vasi, v povprečju pri enakih vrednostih ostalih spremenljivk (izobrazba, število delovnih ur na teden, spol) za 6.2 tisoč sit nižjo bruto plačo kot tisti, ki živijo v velikih mestih.

Če želimo nastaviti kot referenčno kakšno drugo kategorijo, to najlažje storimo tako, da zamenjamo vrstni red *levels* pri faktorju pred klicem funkcije *lm*. Če bi želeli, da je na primer referenčna kategorija "manjše mesto", lahko to dosežemo takole:

```

> dataLR$F5a<-factor(dataLR$F5,
                    levels=levels(dataLR$F5)[c(3,1,2,4,5)])
> lm(G91~F7+F21+O1F2+F5a,data=dataLR)
Call:
lm(formula = G91 ~ F7 + F21 + O1F2 + F5a, data = dataLR)

Coefficients:
(Intercept)           F7           F21
      0.9370         21.4129        -0.2492

```

```

      01F2zenski  F5aveliko mesto    F5apredmestje
      -33.1629      -11.4633         -27.3280
      F5avas      F5akmetija
      -17.6796      -14.6960
> #sedaj je referenčna kategorija "manjse mesto"
> summary(lm(G91~F7+F21+01F2+F5a,data=dataLR))
Call:
lm(formula = G91 ~ F7 + F21 + 01F2 + F5a, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-248.20  -66.53  -17.92   49.71  449.09

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.9370    34.9314   0.027  0.97862
F7             21.4129     1.9568  10.943 < 2e-16 ***
F21            -0.2492     0.4765  -0.523  0.60132
01F2zenski    -33.1629    11.7890  -2.813  0.00523 **
F5aveliko mesto -11.4633    22.5129  -0.509  0.61099
F5apredmestje -27.3280    19.9023  -1.373  0.17073
F5avas        -17.6796    14.9753  -1.181  0.23869
F5akmetija    -14.6960    22.0729  -0.666  0.50605
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 101.8 on 304 degrees of freedom
Multiple R-squared:  0.3159,    Adjusted R-squared:  0.3002
F-statistic: 20.06 on 7 and 304 DF,  p-value: < 2.2e-16

```

Pri enakih vrednostih ostalih neodvisnih spremenljivk bi v vseh drugih krajih bivanja v povprečju pričakovali nižjo bruto plačo kot v velikem mestu, vendar pa ni nobeden od regresijskih koeficientov statistično značilen (niti pri 10-% tveganju).

Vendar na podlagi tega izpisa ne sklepate o domnevi, da kraj bivanja ne vpliva na bruto plačo (oziroma ne moremo ugotoviti, da tega ne moremo trditi na primer pri 10-% tveganju). Za preverjanje te domneve bi morali s pomočjo  $F$ -testa ta model primerjati z modelom brez kraja bivanja (torej s prejšnjim modelom). Medtem ko izbor referenčne kategorije vpliva na posamezne regresijske koeficiente in posledično tudi na teste značilnosti, le-ta ne vpliva na rezultate  $F$ -testa, kjer preverjamo vpliv "celotne" nominalne spremenljivke (ali katerekoli druge posamezne spremenljivke ali kombinacije več spremenljivk).

```
> anova(fitG91_F7F2101F2,fitG91_F7F2101F2F5)
Analysis of Variance Table

Model 1: G91 ~ F7 + F21 + 01F2
Model 2: G91 ~ F7 + F21 + 01F2 + F5
  Res.Df    RSS Df Sum of Sq    F Pr(>F)
1     308 3173025
2     304 3149602  4     23423 0.5652 0.6881
```

Šele ta primerjava nam pove, da bi morali tvegati več kot 70 %, če bi želeli trditi, da kraj bivanja vpliva na bruto plačo, če izločimo vplive stopnje izobrazbe, števila delovnih ur in spola.

### 3.3.6 Interakcije med spremenljivkami

Interakcije med (učinki) spremenljivk vključimo v model tako, da kot dodatno neodvisno spremenljivko v model vključimo zmnožek spremenljivk. V primeru nominalnih spremenljivk morajo biti te ustrezno kodirane (na primer kot umetne spremenljivke).

#### Opozorilo!

Ko navajamo v **R**-ju "formulo" (npr:  $y \sim x_1 + x_2$ ), znak "\*" pomeni, da želimo v model vključiti izbrane spremenljivke in vse možne interakcije med njimi in ne dejanskega množenja (upoštevajo se na primer tudi tipi spremenljivk). Če želimo vključiti samo interakcijo, potem damo med spremenljivki(ke) znak ":". Tako je na primer  $y \sim x_1*x_2*x_3$  enako kot  $y \sim x_1 + x_2 + x_3 + x_1:x_2 + x_1:x_3 + x_2:x_3$ . Če želimo, da se nek operator (na primer \*) interpretira dobesedno, damo izraz kot argument funkciji *I* (na primer  $I(x_1*x_2)$ ).

Če torej želimo vključiti interakcijo med spolom in izobrazbo (edini spremenljivki, ki sta imeli statistično značilen vpliv), lahko to naredimo tako, kot prikazuje naslednji izpis.

```
> fitG91_F7F2101F2F5int <- lm(G91 ~ F7*01F2 + F21 + F5,
  data=dataLR)
> #ali takole
> #fitG91_F7F2101F2F5int <- lm(G91 ~ F7 + F21 + 01F2 + F5
  + F7:01F2, data=dataLR)
> summary(fitG91_F7F2101F2F5int)
Call:
lm(formula = G91 ~ F7 * 01F2 + F21 + F5, data = dataLR)
```

```

Residuals:
  Min       1Q   Median       3Q      Max
-233.87  -65.01  -20.10   50.17  436.74

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -50.52516   46.38880  -1.089   0.277
F7            24.55524    2.89956   8.469 1.09e-15 ***
O1F2zenski   37.00699    49.28161   0.751   0.453
F21          -0.20133    0.47672  -0.422   0.673
F5predmestje -16.07825    24.57842  -0.654   0.514
F5manjse mesto  8.29630    22.57400   0.368   0.713
F5vas        -7.01999    21.18518  -0.331   0.741
F5kmetija    -0.02075    26.84935  -0.001   0.999
F7:O1F2zenski -5.63564    3.84354  -1.466   0.144
---
Signif. codes:
  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 101.6 on 303 degrees of freedom
Multiple R-squared:  0.3207,    Adjusted R-squared:  0.3028
F-statistic: 17.88 on 8 and 303 DF,  p-value: < 2.2e-16

```

Vpliv interakcije ni statistično značilen. Prav tako pa ni več statistično značilen vpliv spola. To je posledica multikolinearnosti, ki jo bomo omenili v naslednjem podpodpoglavju. Problem lahko zmanjšamo, če intervalne spremenljivke pred računanjem interakcije centriramo (od vsake vrednosti odštejemo povprečje). Zaradi lepšega izpisa je bolje, da to naredimo pred klicem funkcije *lm*.

```

> dataLR$F7centGndrZenski<- (dataLR$F7-mean(dataLR$F7))*
  (dataLR$O1F2=="zenski")
> fitG91_F7F21O1F2F5int2<-lm(G91~F7+F21+O1F2+F5 +
  F7centGndrZenski,data=dataLR)
> summary(fitG91_F7F21O1F2F5int2)
Call:
lm(formula = G91 ~ F7 + F21 + O1F2 + F5 + F7centGndrZenski,
    data = dataLR)

Residuals:
  Min       1Q   Median       3Q      Max
-233.87  -65.01  -20.10   50.17  436.74

```

```

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -50.52516   46.38880  -1.089  0.2769
F7           24.55524    2.89956   8.469 1.09e-15 ***
F21          -0.20133    0.47672  -0.422  0.6731
O1F2zenski  -32.64380   11.77205  -2.773  0.0059 **
F5predmestje -16.07825   24.57842  -0.654  0.5135
F5manjse mesto  8.29630   22.57400   0.368  0.7135
F5vas        -7.01999   21.18518  -0.331  0.7406
F5kmetija    -0.02075   26.84935  -0.001  0.9994
F7centGndrZenski -5.63564    3.84354  -1.466  0.1436
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 101.6 on 303 degrees of freedom
Multiple R-squared:  0.3207,      Adjusted R-squared:  0.3028
F-statistic: 17.88 on 8 and 303 DF,  p-value: < 2.2e-16

```

Spremenil se je le koeficient za spol, vsi ostali rezultati (vključno z  $R^2$  in  $F$ -statistiko) pa so ostali nespremenjeni. Pravzaprav se je spremenilo samo "kodiranje spremenljivk". Če vpliv spremenljivk, vključenih v interakcije, preverjamo na pravilen način, torej tako, da preverjamo vpliv celotne spremenljivke, se pravi "glavni" učinek in vse interakcije, v katerih nastopa preko  $F$ -testa, potem kodiranje ni pomembno. Pravilno bi učinek spola torej preverjali tako, kot sledi v nadaljevanju, kjer način kodiranja na rezultat testa ne vpliva.

```

> #ocenimo model brez spola
> fitG91_F7F21F5int <- lm(G91 ~ F7 + F21 + F5, data=dataLR)
> #preverimo skupni vpliv spola, tako da primerjamo
> #model brez spola z modelom s spolom
> anova(fitG91_F7F21F5int, fitG91_F7F21O1F2F5int)
Analysis of Variance Table

Model 1: G91 ~ F7 + F21 + F5
Model 2: G91 ~ F7 * O1F2 + F21 + F5
  Res.Df    RSS Df Sum of Sq    F    Pr(>F)
1     305 3231587
2     303 3127412  2     104176 5.0465 0.006983 **
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

```

Ugotovimo lahko, da je vpliv spola (kjer upoštevamo tudi njegov vpliv preko interakcije z izobrazbo) statistično značilen pri stopnji tveganja 0.7 %.

### 3.3.7 Preverjanje predpostavk

#### Analiza rezidualov

Deloma smo preverjanje predpostavk že obdelali, ko smo si ogledali diagnostične grafikone za porazdelitev rezidualov. Veliko predpostavk je namreč vezanih na porazdelitev rezidualov. Še na zadnjem modelu (z interakcijo 2) pogledjmo standardne grafikone za diagnostiko. Grafi so prikazani na sliki 3.15.

```
> par(mfrow=c(2,2))
> plot(fitG91_F7F2101F2F5int2)
> par(mfrow=c(1,1))
```

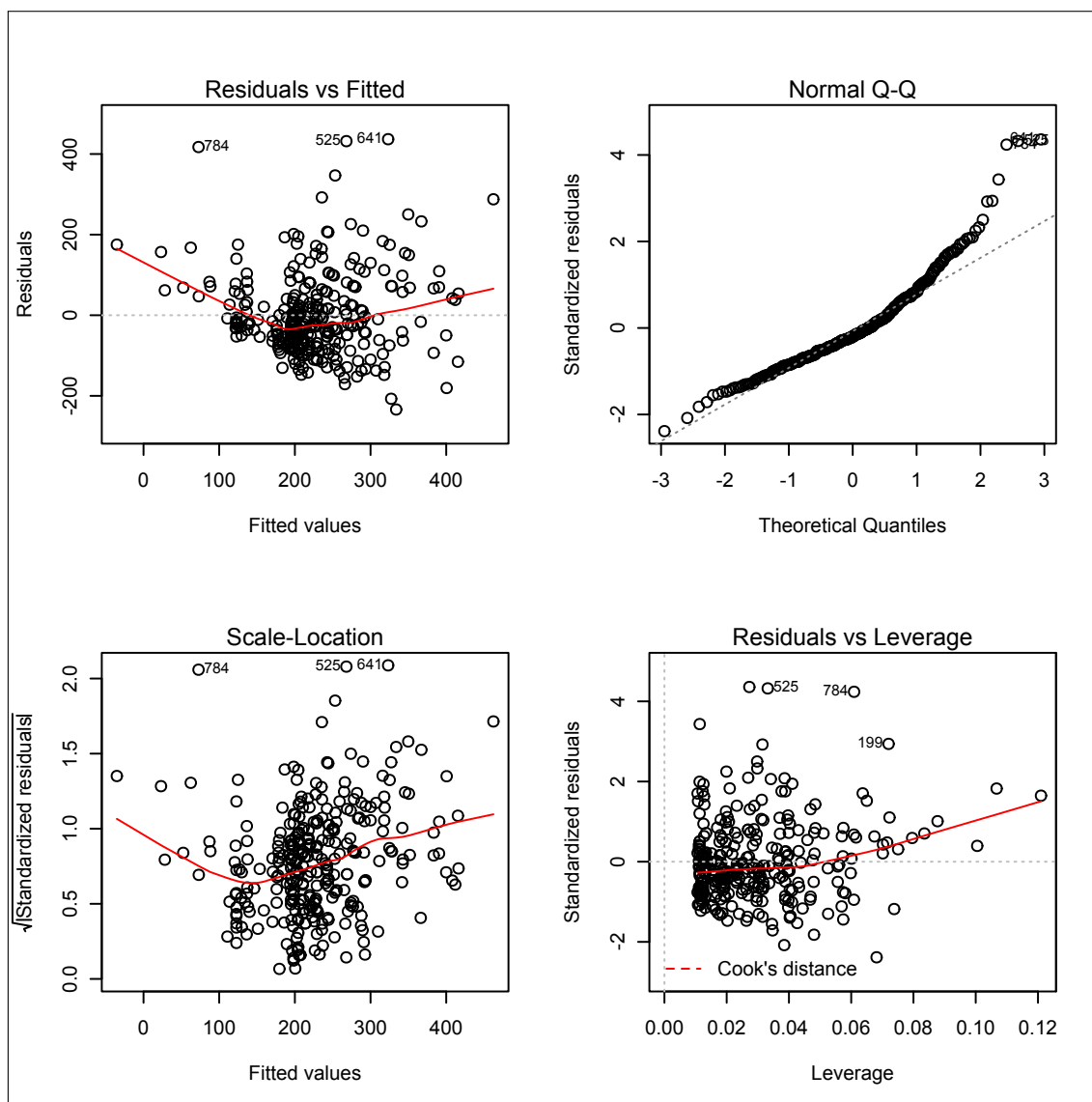
Iz grafov na sliki 3.15 lahko razberemo:

- Že na 1. grafu (levo zgoraj) se vidi, da je variabilnost rezidualov večja pri večjih plačah kot pri manjših. To je še bolj razvidno iz 3. grafa (levo spodaj). Graf je podoben zgornjemu (1.), le da sedaj namesto "surovih" rezidualov na  $y$  os nanašamo korenjene absolutne standardizirane rezidualne. Pri tem je bistveno predvsem to, da nanašamo absolutne vrednosti. Rdeča črta, ki prikazuje glajena povprečja, bi morala biti v primeru homoskedastičnosti (ko je ta predpostavka izpolnjena) ravna, pri nas pa je očitno, da od takrat, ko imamo neko večje število enot, vseskozi narašča. Torej je prisotna heteroskedastičnost.
- Na 2. grafu (desno zgoraj) vse točke ne ležijo na premici, kar pomeni, da se reziduali ne porazdeljujejo normalno. Predvsem imamo preveč zelo velikih vrednosti. Enako lahko razberemo tudi iz histograma rezidualov z vrisano normalno krivuljo na sliki 3.16.

```
> par(mar=mar.def-c(0,0,3,0))
> hist(fitG91_F7F2101F2F5int2$res, freq=FALSE, xlab="Reziduali",
      ylab="Gostota", main="", br=15)
> curve(dnorm(x, sd=sd(fitG91_F7F2101F2F5int2$res)), add=TRUE)
> par(mar=mar.def)
```

Za preverjanje heteroskedastičnosti obstaja v paketku `car` tudi formalni test (funkcija `ncvTest`). Prav tako ta paketek ponuja še dodatni grafikon, ki nariše logaritmirane absolutne vrednosti studentiziranih rezidualov v odvisnosti od logaritmiranih napovedanih vrednosti (funkcija `spreadLevelPlot`). Zaradi uporabe logaritmov izpusti vse negativne napovedane vrednosti (in v tem primeru izpiše opozorilo).

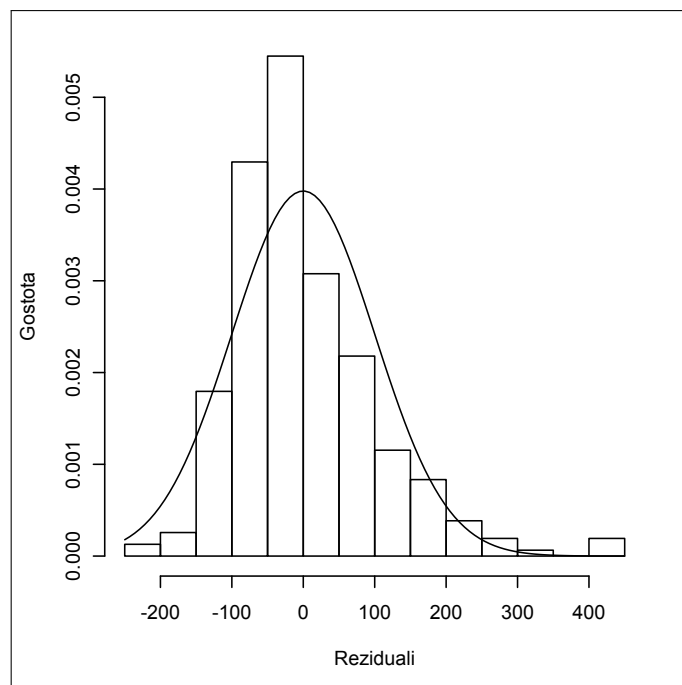
Slika 3.15: Diagnostični grafikoni za linearno regresijo – model z interakcijo



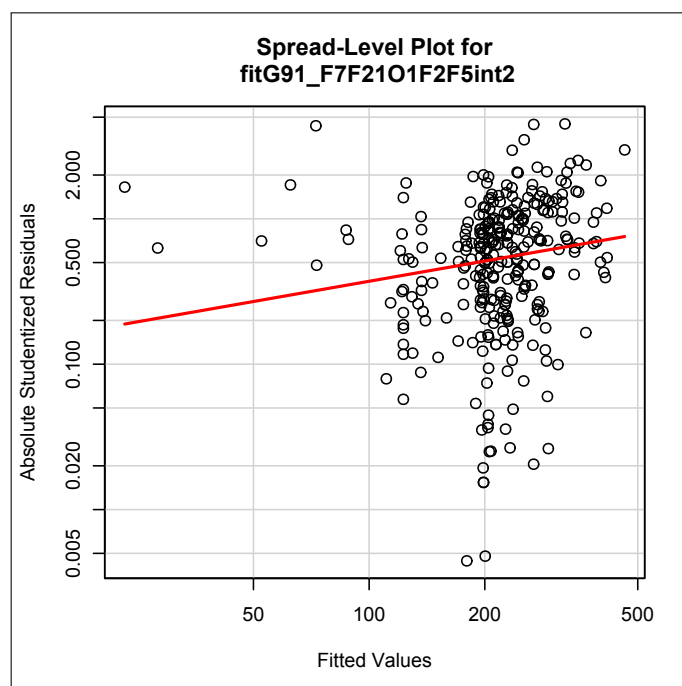
Funkcija predlaga tudi transformacijo odvisne spremenljivke za odpravljanje problema heteroskedastičnosti. Graf je prikazan na sliki 3.17.

```
> ncvTest(fitG91_F7F2101F2F5int2)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 14.84605    Df = 1    p = 0.0001166521
> spreadLevelPlot(fitG91_F7F2101F2F5int2)
Suggested power transformation: 0.5373901
```

Slika 3.16: Histogram rezidualov



Slika 3.17: Grafikon za ocenjevanje heteroskedastičnosti



Na podlagi testa lahko domnevo o homoskedastičnosti zavrnilo pri zanemarljivi stopnji tveganja. Iz grafa na sliki 3.17 pa zopet vidimo, da variabilnost rezidualov z višanjem napovedanih vrednosti narašča. Na podlagi regresijskega koeficienta premice na tem grafu funkcija predlaga tudi potenco, na katero naj bi potencirali odvisno spremenljivko za odpravo problema heteroskedastičnosti.

## Multikolinearnost

Multikolinearnost lahko ocenjujemo z več statistikami. Poglejmo si najprej izračun faktorja povečanja variance (variance inflation factor – vif) in toleranc. Vif lahko izračunamo s funkcijo `vif` iz paketa `car`, tolerance pa so kar  $1/vif$ . Ti dve statistiki sta priporočljivi, ker sta razumljivi in pokažeta, pri katerih spremenljivkah se problem pojavlja. Mere bomo izračunali za oba modela z interakcijo (ki se razlikujeta v parametrizaciji interakcije).

*Opomnik:* Drugo parametrizacijo smo izbrali ravno zato, ker zmanjšuje multikolinearnost.

```
> #prvi model
> vif(fitG91_F7F2101F2F5int) #variance inflation factor
      GVIF Df GVIF^(1/(2*Df))
F7      2.396263  1      1.547987
01F2    18.183935  1      4.264263
F21     1.052758  1      1.026040
F5      1.216437  4      1.024793
F7:01F2 19.529971  1      4.419273
> 1/vif(fitG91_F7F2101F2F5int)[,1] #tolerance
      F7      01F2      F21      F5      F7:01F2
0.41731645 0.05499360 0.94988557 0.82207330 0.05120335
> #drugi model
> vif(fitG91_F7F2101F2F5int2) #variance inflation factor
      GVIF Df GVIF^(1/(2*Df))
F7      2.396263  1      1.547987
F21     1.052758  1      1.026040
01F2    1.037582  1      1.018618
F5      1.216437  4      1.024793
F7centGndrZenski 2.259556  1      1.503182
> 1/vif(fitG91_F7F2101F2F5int2)[,1] #tolerance
      F7      F21      01F2
0.4173164      0.9498856      0.9637791
      F5 F7centGndrZenski
0.8220733      0.4425648
```

Tolerance so pri spolu in pri interakciji med spolom in izobrazbo veliko višje pri drugem modelu, faktorji povečanja variance pa nižji.

Dober pokazatelj multikolinearnosti je tudi korelacijska matrika med **ocenami** regresijskih koeficientov. Ta kaže na problem, če so korelacije po absolutni vrednosti blizu 1. S funkcijo `vcov` dobimo variančno/kovariančno matriko, ki pa jo lahko s funkcijo `cov2cor` pretvorimo v korelacijsko.

```
> #prvi model
> #vcov(fitG91_F7F2101F2F5int) #kovariančna matrika
> cov2cor(vcov(fitG91_F7F2101F2F5int)) #korelacijska matrika
```

	(Intercept)	F7	01F2zenski
(Intercept)	1.0000000	-0.80916407	-0.60845623
F7	-0.8091641	1.00000000	0.71746753
01F2zenski	-0.6084562	0.71746753	1.00000000
F21	-0.4241575	0.02872814	0.08249693
F5predmestje	-0.3471743	0.03199666	0.00178113
F5manjse mesto	-0.2765647	-0.03856876	-0.08880756
F5vas	-0.3666217	0.06890147	-0.02361653
F5kmetija	-0.4616874	0.22093944	0.11067464
F7:01F2zenski	0.5880620	-0.73911673	-0.97107735

	F21	F5predmestje	F5manjse mesto
(Intercept)	-0.424157534	-0.347174290	-0.27656470
F7	0.028728137	0.031996658	-0.03856876
01F2zenski	0.082496930	0.001781130	-0.08880756
F21	1.000000000	0.008758751	-0.09876780
F5predmestje	0.008758751	1.000000000	0.64435236
F5manjse mesto	-0.098767796	0.644352359	1.00000000
F5vas	-0.142169932	0.689732959	0.76571906
F5kmetija	-0.057753038	0.555335692	0.59905425
F7:01F2zenski	-0.068533086	0.005924216	0.09568067

	F5vas	F5kmetija	F7:01F2zenski
(Intercept)	-0.36662171	-0.46168741	0.588061950
F7	0.06890147	0.22093944	-0.739116728
01F2zenski	-0.02361653	0.11067464	-0.971077352
F21	-0.14216993	-0.05775304	-0.068533086
F5predmestje	0.68973296	0.55533569	0.005924216
F5manjse mesto	0.76571906	0.59905425	0.095680667
F5vas	1.00000000	0.66452253	0.025873515
F5kmetija	0.66452253	1.00000000	-0.081587125
F7:01F2zenski	0.02587352	-0.08158713	1.000000000

```

> #drugi model
> #vcov(fitG91_F7F2101F2F5int2) #kovariančna matrika
> cov2cor(vcov(fitG91_F7F2101F2F5int2)) #korelacijska matrika
      (Intercept)          F7          F21
(Intercept)  1.0000000 -0.80916407 -0.424157534
F7            -0.8091641  1.00000000  0.028728137
F21          -0.4241575  0.02872814  1.000000000
01F2zenski   -0.1742628  0.02108728  0.068815958
F5predmestje -0.3471743  0.03199666  0.008758751
F5manjse mesto -0.2765647 -0.03856876 -0.098767796
F5vas        -0.3666217  0.06890147 -0.142169932
F5kmetija    -0.4616874  0.22093944 -0.057753038
F7centGndrZenski  0.5880620 -0.73911673 -0.068533086
      01F2zenski F5predmestje F5manjse mesto
(Intercept)   -0.174262837 -0.347174290   -0.27656470
F7             0.021087276  0.031996658   -0.03856876
F21           0.068815958  0.008758751   -0.09876780
01F2zenski    1.000000000  0.031361616    0.01431086
F5predmestje  0.031361616  1.000000000    0.64435236
F5manjse mesto 0.014310857  0.644352359    1.00000000
F5vas         0.005537691  0.689732959    0.76571906
F5kmetija     0.134101643  0.555335692    0.59905425
F7centGndrZenski -0.030070878  0.005924216    0.09568067
      F5vas  F5kmetija F7centGndrZenski
(Intercept) -0.366621708 -0.46168741  0.588061950
F7           0.068901467  0.22093944 -0.739116728
F21         -0.142169932 -0.05775304 -0.068533086
01F2zenski  0.005537691  0.13410164 -0.030070878
F5predmestje 0.689732959  0.55533569  0.005924216
F5manjse mesto 0.765719055  0.59905425  0.095680667
F5vas        1.000000000  0.66452253  0.025873515
F5kmetija    0.664522533  1.00000000 -0.081587125
F7centGndrZenski 0.025873515 -0.08158713  1.000000000

```

Ponovno je glavna razlika med obema modeloma povezana s spolom in interakcijo med spolom in izobrazbo. Tokrat je dejanska razlika prav v korelaciji med ocenama teh dveh koeficientov. Medtem ko je ta korelacija pri prvem modelu po absolutni vrednosti zelo blizu 1 ( $-0.97$ ), je ta korelacija pri drugem modelu skoraj 0 ( $-0.03$ ). Razmeroma visoka ( $-0.75$ ) je tudi korelacija med ocenama koeficientov za izobrazbo in interakcijo med spolom in izobrazbo, ki je  $-0.75$ . Preostale razmeroma visoke korelacije so le še med sklopom spremenljivk, ki merijo kraj bivanja (ker merijo isto spremenljivko, je to pričakovano) in med nekaterimi spremenljivkami in konstanto (kar pa ni tako pomembno).

Multikolinearnost lahko ocenimo tudi s pomočjo indeksov pogojnosti in lastnih vrednosti matrike neodvisnih spremenljivk (v katero so namesto nominalnih spremenljivk vključene umetne spremenljivke). Obstaja več načinov, kako izračunati indekse pogojnosti. V nadaljevanju sta predstavljena dva. Pri tem bomo uporabili funkcijo `colinEigen`, ki je definirana spodaj, sicer pa se nahaja tudi v datoteki "UcbenikR-funkcije.R". Funkcija temelji na izračunu lastnih vrednosti matrike križnih produktov neodvisnih spremenljivk.

```
> colinEigen<-function(fit,SPSS=TRUE){
  #funkcija za računanje lastnih vrednosti in indeksov
  # pogojnosti
  X<-fit["x"]
  if(is.null(X)){
    stop("Fit must include x. See ?lm for details\n")
  }else{
    X<-fit$x
    tXX<-t(X)%*%X
    if(SPSS){
      tXX<-cov2cor(tXX)
    }
    e <- eigen(tXX)
    return(list(eigen=e$values,
               condIndex=sqrt(e$val[1]/e$val)))
  }
}
```

Izračunajmo torej indekse pogojnosti na oba načina za oba modela.

```
> fitG91_F7F2101F2F5int2<-lm(G91~F7+F21+01F2+F5 +
  F7centGndrZenski,data=dataLR,x=TRUE)
> #ponovno smo ocenili model s parametrom x=TRUE
> #da dobimo kot rezultat tudi matriko neodvisnih spremenljivk
> fitG91_F7F2101F2F5int<-lm(G91~F7*01F2+F21+F5,data=dataLR
  ,x=TRUE)
> #prvi model
> #1. način
> colinEigen(fitG91_F7F2101F2F5int,SPSS=FALSE)
$eigen
[1] 6.652435e+05 1.558161e+04 4.912678e+03 9.900302e+01
[5] 5.302790e+01 3.528826e+01 1.873120e+01 5.532393e+00
[9] 2.662550e+00
```

```

$condIndex
[1] 1.000000 6.534074 11.636735 81.972106 112.005173
[6] 137.301463 188.454950 346.763845 499.851942
> # oziroma taki, kot jih vrne SPSS
> colinEigen(fitG91_F7F2101F2F5int, SPSS=TRUE)
$eigen
[1] 4.943854688 1.088057228 1.002859625 1.000235359
[5] 0.766591055 0.092809934 0.066547583 0.031866833
[9] 0.007177695

$condIndex
[1] 1.000000 2.131606 2.220306 2.223216 2.539516
[6] 7.298534 8.619195 12.455564 26.244625
> #drugi model
> #1. način
> colinEigen(fitG91_F7F2101F2F5int2, SPSS=FALSE)
$eigen
[1] 6.558974e+05 6.585408e+03 1.173476e+03 1.011405e+02
[5] 7.781142e+01 5.243155e+01 3.498490e+01 9.599113e+00
[9] 3.705997e+00

$condIndex
[1] 1.000000 9.979909 23.641818 80.529554 91.811327
[6] 111.846291 136.923385 261.398078 420.693096
> # oziroma taki, kot jih vrne SPSS
> colinEigen(fitG91_F7F2101F2F5int2, SPSS=TRUE)
$eigen
[1] 4.340401831 1.109346570 1.012708187 1.000586611
[5] 0.923653402 0.473392997 0.076159218 0.053759618
[9] 0.009991567

$condIndex
[1] 1.000000 1.978023 2.070250 2.082752 2.167756
[6] 3.027987 7.549249 8.985389 20.842421

```

Ne glede na izračun so indeksi pogojnosti pri prvem modelu bistveno večji, kar nakazuje, da je tam multikolinearnost večji problem. Kljub vsemu pa tudi tu indeks pogojnosti na način, kot ga izračuna SPSS, ne preseže meje 30, kar bi bil jasen signal za multikolinearnost.

### Ocena oblike zveze

Povedali smo že, da je za vsako neodvisno spremenljivko, vključeno v model, dobro pogledati razsevni grafikon z odvisno spremenljivko. Vendar pa ta grafikon vča-

sih ne razkrije prave zveze, ker je prava zveza "zakrita" z vplivi ostalih neodvisnih spremenljivk. V tem primeru pride prav "component + residual plot" (oziroma graf delnih ostankov), kjer na  $y$  os nanašamo vrednosti odvisne spremenljivke, od katere prej odštejemo vplive ostalih spremenljivk. Take grafe lahko narišemo s pomočjo funkcije `crPlots` iz paketa `car`. Pogoj za uporabo funkcije je, da v modelu ni interakcij, zato jih bomo uporabili na zadnjem modelu, kjer še nismo imeli interakcije. Rezultat je prikazan na sliki 3.18

```
> # "component + residual plot"
> crPlots(fitG91_F7F2101F2F5)
```

Nadgradnja teh grafikonov so CERES grafi, ki so na voljo v funkciji `ceresPlots` v istem paketu. Za razliko od prejšnje le-ta ne nariše grafov za nominalne spremenljivke. Rezultat je na sliki 3.19.

```
> # "Ceres" graf - naprednejša verzija zgornjega
> ceresPlots(fitG91_F7F2101F2F5)
```

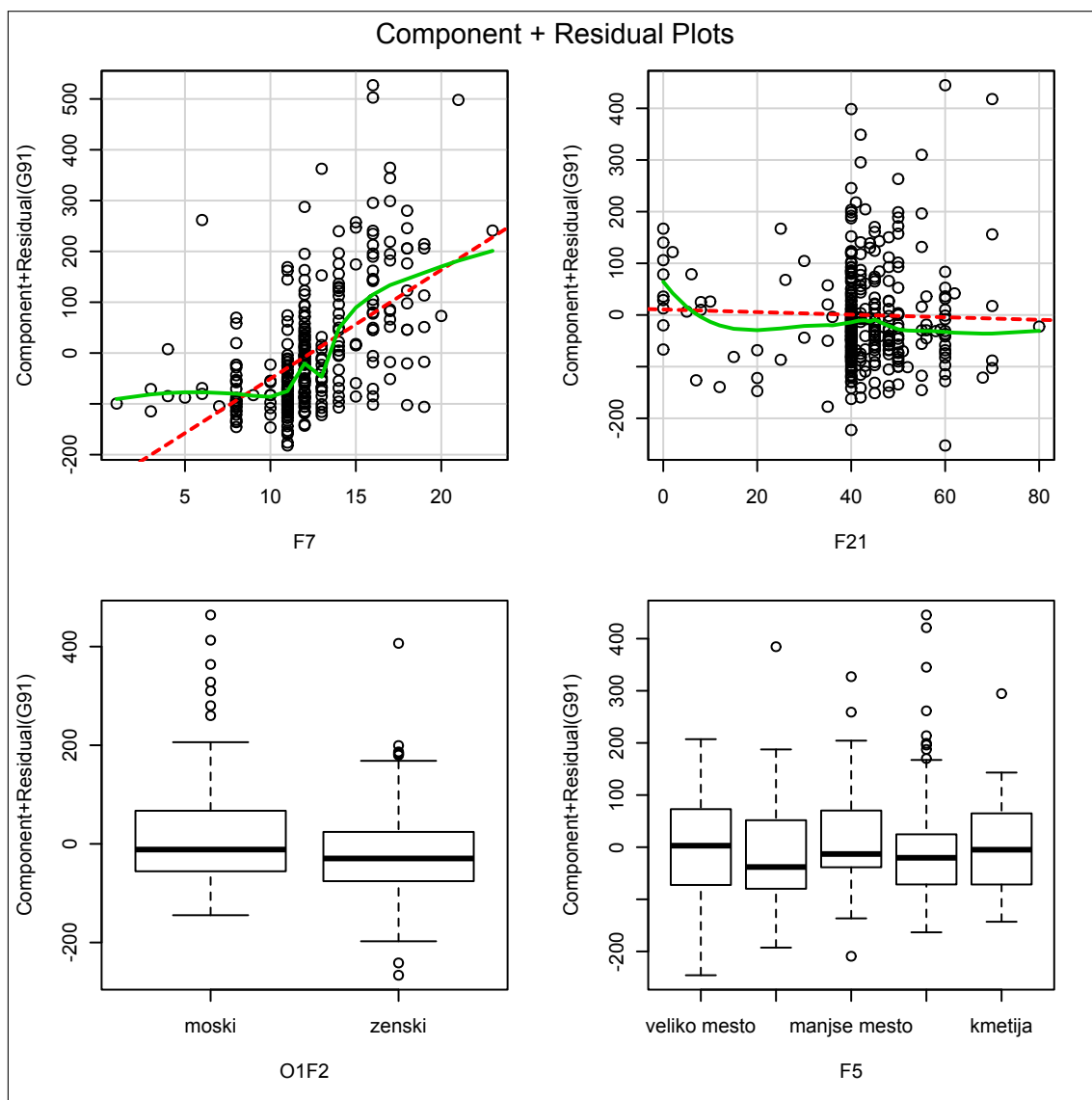
Najzanimivejši zaključek, ki je bil opazen že na navadnih razsevnih grafikonih, je, da izobrazba vpliva na plačo šele, ko število let šolanja preseže 12 leta. Kot rešitev tega problema lahko v model vključimo še eno spremenljivko, in sicer "število let šolanja nad 12 let", ki bo imela vrednost 0 za vse, ki imajo 12 let šolanja ali manj.

```
> dataLR$F7nad12 <- dataLR$F7 - 12
> dataLR$F7nad12[dataLR$F7nad12<0]<-0
> fitG91_F7F2101F2F5F7nad12<-lm(G91~F7 +01F2+F7nad12+F21+F5,
    data=dataLR)
> summary(fitG91_F7F2101F2F5F7nad12)
Call:
lm(formula = G91 ~ F7 + 01F2 + F7nad12 + F21 + F5, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-280.67  -56.50  -17.33   42.23  425.37

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   130.7917    48.0829   2.720   0.0069 **
F7              7.3834     3.6457   2.025   0.0437 *
01F2zenski   -42.4936    11.6175  -3.658   0.0003 ***
F7nad12       24.0146     5.3287   4.507 9.42e-06 ***
F21           -0.2669     0.4621  -0.578  0.5640
```

Slika 3.18: Grafikoni delnih ostankov



F5predmestje	-9.4125	23.9206	-0.393	0.6942
F5manjse mesto	19.3332	21.9000	0.883	0.3780
F5vas	5.6539	20.7426	0.273	0.7854
F5kmetija	-10.5924	26.0487	-0.407	0.6846

---

Signif. codes:

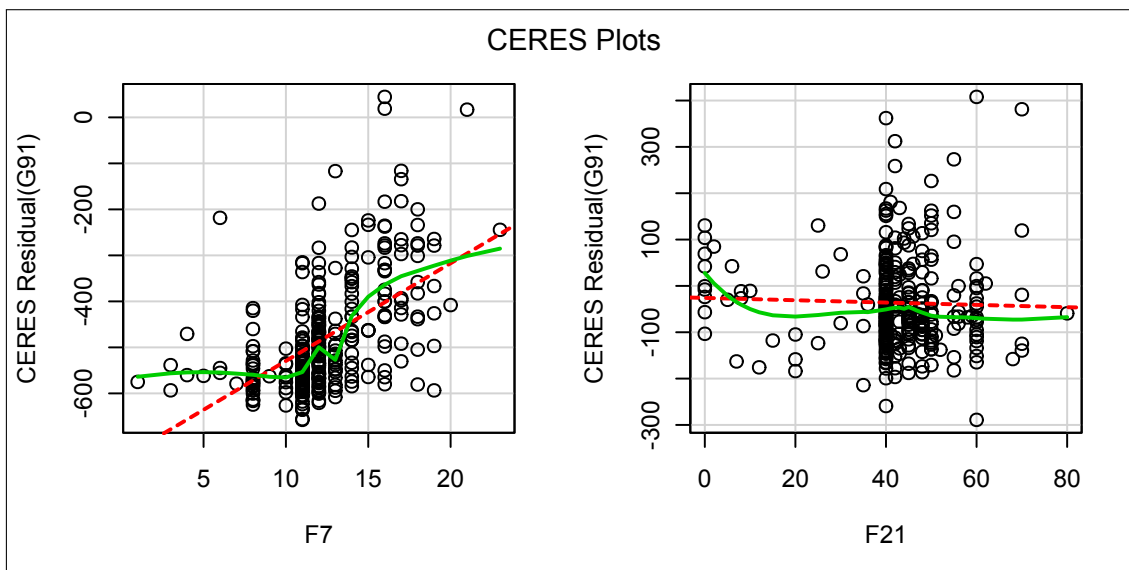
0 \*\*\* 0.001 \*\* 0.01 \* 0.05 . 0.1 1

Residual standard error: 98.7 on 303 degrees of freedom

Multiple R-squared: 0.3589, Adjusted R-squared: 0.342

F-statistic: 21.2 on 8 and 303 DF, p-value: &lt; 2.2e-16

Slika 3.19: Ceres grafikoni



```
> summary(fitG91_F7F2101F2F5) #za primerjavo
Call:
lm(formula = G91 ~ F7 + F21 + 01F2 + F5, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-248.20  -66.53  -17.92   49.71  449.09

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -10.5263    37.5909  -0.280  0.77965
F7             21.4129     1.9568  10.943 < 2e-16 ***
F21            -0.2492     0.4765  -0.523  0.60132
01F2zenski   -33.1629    11.7890  -2.813  0.00523 **
F5predmestje -15.8647    24.6244  -0.644  0.51989
F5manjse mesto 11.4633    22.5129   0.509  0.61099
F5vas         -6.2163    21.2181  -0.293  0.76974
F5kmetija     -3.2327    26.8104  -0.121  0.90411
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 101.8 on 304 degrees of freedom
Multiple R-squared:  0.3159,    Adjusted R-squared:  0.3002
F-statistic: 20.06 on 7 and 304 DF,  p-value: < 2.2e-16
```

```
> par(mfrow=c(2,2))
> plot(fitG91_F7F2101F2F5F7nad12)
> par(mfrow=c(1,1))
```

Vpliv spremenljivke F7 je bistveno manjši in sedaj komaj statistično značilen pri 5-% tveganju, kar nam pove, da izobrazba do 12 let ne vpliva močno na bruto plačo. Ker pa je koeficient spremenljivke F7nad12 bistveno večji in močno statistično značilen, vidimo, da začne izobrazba močneje vplivati na bruto plačo šele, ko število let šolanja preseže 12 let oziroma po srednji šoli. O tem, da je model sedaj bistveno boljši, pričča tudi znatno večji  $R^2$ .

Diagnostični grafikoni so prikazani na sliki 3.20. Rezultati so sicer malce boljši, a problem heteroskedastičnosti ostaja.

### 3.3.8 V razmislek ★

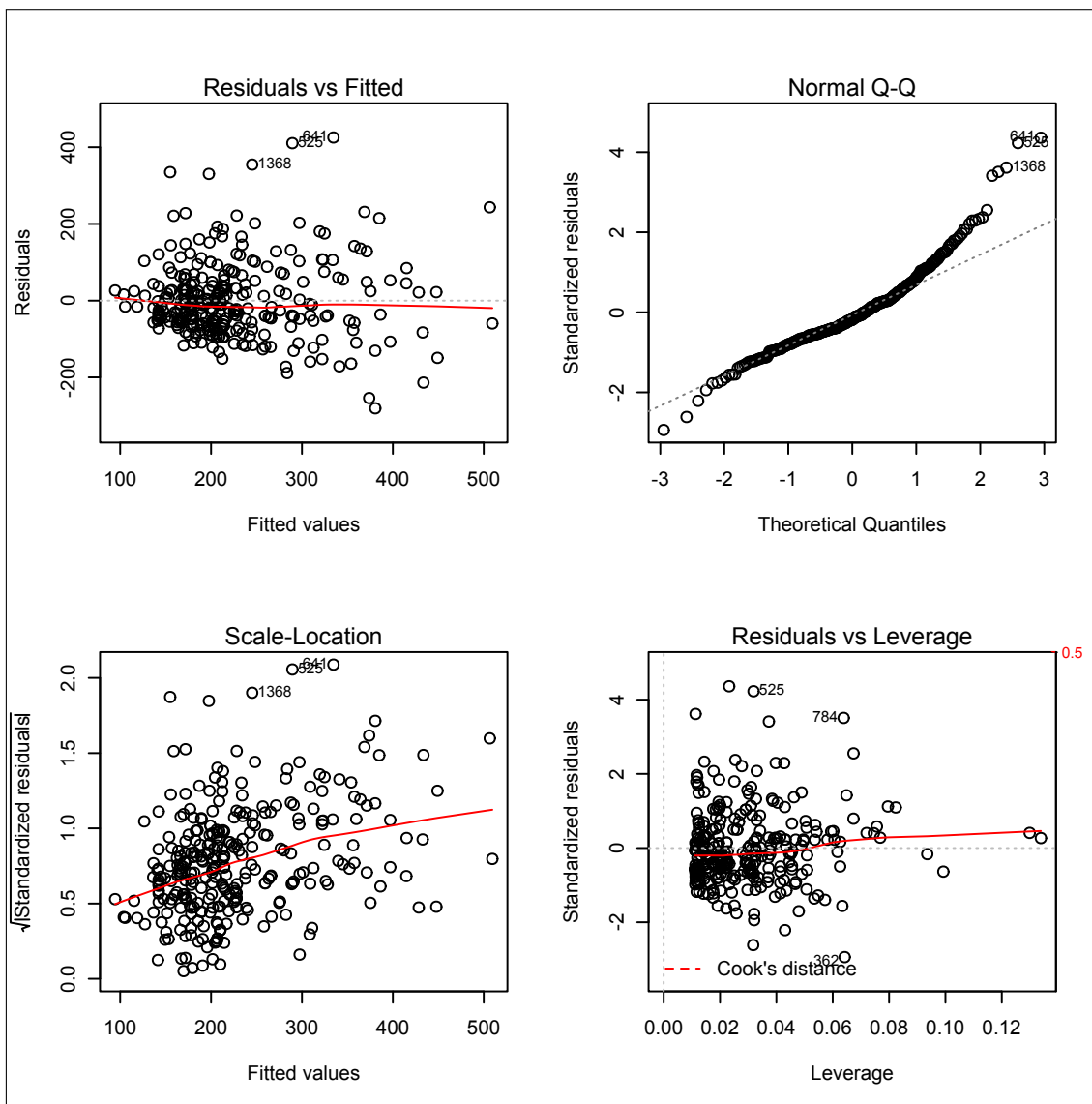
Model bi se dalo še nadgrajevati. Že samo vpliv izobrazbe na plačo za različne skupine je zelo kompleksna stvar. Kaj se zgodi, če v novi model dodamo interakcijo?

```
> dataLR$F7centGndrZenski <- (dataLR$F7 - mean(dataLR$F7)) *
  (dataLR$O1F2 == "zenski")
> dataLR$F7nad12centGndrZenski <- (dataLR$F7nad12 -
  mean(dataLR$F7nad12)) * (dataLR$O1F2 == "zenski")
> fitG91_F7F2101F2F5F7nad12int <- lm(G91 ~ F7 + F7nad12 + O1F2 + F21 + F5 +
  F7centGndrZenski + F7nad12centGndrZenski, data = dataLR)
> summary(fitG91_F7F2101F2F5F7nad12int)
Call:
lm(formula = G91 ~ F7 + F7nad12 + O1F2 + F21 + F5 +
    F7centGndrZenski + F7nad12centGndrZenski, data = dataLR)

Residuals:
    Min       1Q   Median       3Q      Max
-253.91  -58.85  -16.94   45.57  421.66

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    170.1907    63.2377   2.691 0.007516
F7              2.8595     5.2133   0.549 0.583754
F7nad12        35.8691     7.3064   4.909 1.5e-06
O1F2zenski    -41.5327    11.5258  -3.603 0.000367
F21            -0.1829     0.4595  -0.398 0.690894
F5predmestje  -8.6485    23.7140  -0.365 0.715590
```

Slika 3.20: Diagnostični grafkoni za linearno regresijo



F5manjse mesto	17.3586	21.8196	0.796	0.426918
F5vas	5.9867	20.5770	0.291	0.771297
F5kmetija	-9.1023	25.9186	-0.351	0.725693
F7centGndrZenski	10.0282	7.0140	1.430	0.153828
F7nad12centGndrZenski	-25.1789	10.3129	-2.441	0.015203
(Intercept)	**			
F7				
F7nad12	***			
O1F2zenski	***			
F21				

```

F5predmestje
F5manjse mesto
F5vas
F5kmetija
F7centGndrZenski
F7nad12centGndrZenski *
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 97.83 on 301 degrees of freedom
Multiple R-squared: 0.3744, Adjusted R-squared: 0.3536
F-statistic: 18.01 on 10 and 301 DF, p-value: < 2.2e-16

```

Vidimo, da se odstotek pojasnjene variance še malce poveča. To nam pravzaprav nakazuje, da bi morda morali že na začetku gledati graf na sliki 3.21.

```

> plot(G91~F7,data=dataLR,ylab="Bruto plača v 1000 sit",
      xlab="Število let šolanja",pch=as.numeric(01F2),
      col=ifelse(01F2=="moski","blue","red"))
> dataLRzen<-dataLR[dataLR$01F2=="zenski",]
> lines(with(data=dataLRzen,lowess(G91~F7)),lwd=2,lty=2,col="red")
> dataLRmos<-dataLR[dataLR$01F2=="moski",]
> lines(with(data=dataLRmos,lowess(G91~F7)),lwd=2,lty=2,
      col="blue")
> legend(x=1,y=750,legend=c("Moški","Ženske"),lty=2,
      col=c("blue","red"),pch=1:2,yjust=1,xjust=0,
      merge=FALSE,lwd=2)

```

### 3.3.9 Izračun "na roke" ★

Za boljše razumevanje je koristno, da naredimo izračune brez vgrajenih funkcij (oziroma s čim bolj enostavnimi). Ocenimo drugi model z interakcijo iz podpodpoglavja 3.3.6 (ki smo ga shranili v objekt `fitG91_F7F2101F2F5int2`) še na tak način.

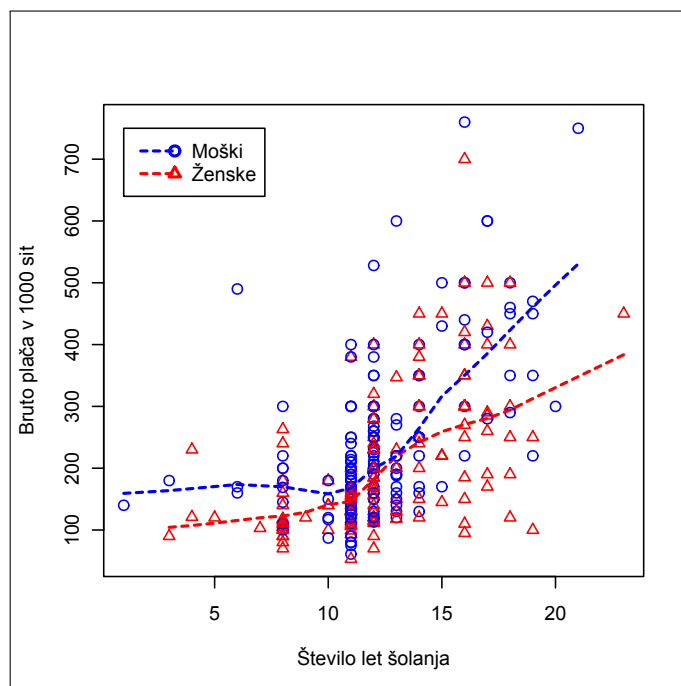
Najprej pripravimo podatke. Pri tem je pomembno predvsem, da naredimo umetne spremenljivke za nominalne spremenljivke.

```

> y<-dataLR$G91          #odvisna spremenljivka
> X<-cbind(konstanta=1,dataLR[c("F7","F21")],
          #konstanta in intervalni neodvisni spremenljivki
          01F2Zenski=as.numeric(dataLR$01F2=="zenski"),

```

Slika 3.21: Odnos med izobrazbo in bruto plačo po spolu



```

F5Predmestje=as.numeric(dataLR$F5=="predmestje"),
"F5Manjše mesto"=as.numeric(dataLR$F5=="manjše mesto"),
F5Vas=as.numeric(dataLR$F5=="vas"),
F5Kmetija=as.numeric(dataLR$F5=="kmetija"),
F7centGndrZenski=(dataLR$F7-mean(dataLR$F7))*
(dataLR$01F2=="zenski")
)
> X<-as.matrix(X)
> #preverimo rezultat
> X[1:10,]
  konstanta F7 F21 01F2Zenski F5Predmestje F5Manjše mesto
1          1  6  42           0           0           0
4          1 12  50           1           0           0
6          1 12  40           1           0           0
16         1 11  62           0           0           0
22         1 14  48           0           0           1
23         1 12   0           0           0           1
26         1 16  45           1           0           0
35         1 12  40           1           0           0
41         1 18  46           0           0           1
49         1 11  40           1           0           1
  F5Vas F5Kmetija F7centGndrZenski
1      0          1          0.000000

```

```

4      1      0      -0.3589744
6      1      0      -0.3589744
16     1      0      0.0000000
22     0      0      0.0000000
23     0      0      0.0000000
26     0      1      3.6410256
35     0      0      -0.3589744
41     0      0      0.0000000
49     0      0      -1.3589744
> #to je enako kot
> fitG91_F7F2101F2F5int2$x[1:10,]
      (Intercept) F7 F21 01F2zenski F5predmestje
1              1  6  42            0            0
4              1 12  50            1            0
6              1 12  40            1            0
16             1 11  62            0            0
22             1 14  48            0            0
23             1 12   0            0            0
26             1 16  45            1            0
35             1 12  40            1            0
41             1 18  46            0            0
49             1 11  40            1            0
      F5manjse mesto F5vas F5kmetija F7centGndrZenski
1              0   0   1            0.0000000
4              0   1   0            -0.3589744
6              0   1   0            -0.3589744
16             0   1   0            0.0000000
22             1   0   0            0.0000000
23             1   0   0            0.0000000
26             0   0   1            3.6410256
35             0   0   0            -0.3589744
41             1   0   0            0.0000000
49             1   0   0            -1.3589744

```

Spomnimo se, da je matrična formula za izračun regresijskih koeficientov:

$$b = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'y$$

Formuli za izračun napovedi in rezidualov pa:

$$y' = \mathbf{X}b$$

$$e = y - y'$$

Na podlagi tega lahko izračunamo  $s[e]$ ,  $R^2$ , popravljen  $R_{pop}^2$  in  $F$ -statistiko:

$$s_e = \frac{\sum_{i=1}^n e^2}{n - k}$$

$$R^2 = \frac{\text{var}(y')}{\text{var}(y)}$$

$$R_{pop}^2 = 1 - \frac{s_e^2}{\text{var}(y)} = 1 - (1 - R^2) \frac{n - 1}{n - k}$$

$$F = \frac{\frac{\sum_{i=1}^n y'_i - \bar{y}}{k - 1}}{\frac{\sum_{i=1}^n y_i - y'_i}{n - k}} = \frac{R^2}{\frac{1 - R^2}{n - k}}$$

Izračunajmo:

```
> b<-solve(t(X) %*% X) %*% t(X) %*% y
> b
              [,1]
konstanta      -50.52516310
F7              24.55524459
F21            -0.20132728
O1F2Zenski    -32.64379964
F5Predmestje  -16.07824506
F5Manjše mesto  8.29629531
F5Vas         -7.01999244
F5Kmetija     -0.02074509
F7centGndrZenski -5.63564466
> #kar je enako kot (do natančnosti računalnika)
> coef(fitG91_F7F21O1F2F5int2)
      (Intercept)          F7          F21
-50.52516310      24.55524459      -0.20132728
      O1F2zenski      F5predmestje      F5manjse mesto
-32.64379964     -16.07824506          8.29629531
          F5vas          F5kmetija      F7centGndrZenski
-7.01999244     -0.02074509      -5.63564466
> yNap <- X %*% b
> e <- y - yNap
> n<-dim(X)[1]
> k<-length(b)
```

```

> sErr<-sqrt(sum(e^2)/(n-k))
> sErr
[1] 101.5947
> #kar je enako kot
> summary(fitG91_F7F2101F2F5int2)$sigma
[1] 101.5947
> R2=var(yNap)/var(y)
> R2
      [,1]
[1,] 0.3207477
> #kar je enako kot
> summary(fitG91_F7F2101F2F5int2)$r.squared
[1] 0.3207477
> R2pop<-1-sErr^2/var(y)
> R2pop
[1] 0.3028137
> #kar je enako kot
> summary(fitG91_F7F2101F2F5int2)$adj.r.squared
[1] 0.3028137
> F=(R2/(k-1))/((1-R2)/(n-k))
> F
      [,1]
[1,] 17.88484
> (df1<-k-1)
[1] 8
> (df2<-n-k)
[1] 303
> pf(q=F,df1=df1,df2=df2,lower.tail =FALSE)
      [,1]
[1,] 7.405752e-22
> cat(capture.output(summary(fitG91_F7F2101F2F5int2))[27],"\n")
F-statistic: 17.88 on 8 and 303 DF,  p-value: < 2.2e-16
> #p-vrednosti se malce razlikujeta, a sta obe praktično 0
> #verjetno gre za vprašanje računske natančnosti

```

Izračunajmo še variančno-kovariančno matriko za ocene regresijskih koeficientov in iz nje še standardne napake za ocene regresijskih koeficientov. Na podlagi variančno-kovariančne matrike lahko izračunamo še korelacijsko matriko ocen regresijskih koeficientov (za ocenjevanje multikolinearnosti):

$$S_b = s_e^2 \cdot X'X^{-1}$$

$$se(b_i) = \sqrt{S_b[i, i]}$$

$$w = \text{diag} \left( \frac{1}{\sqrt{\text{diag}(S_b)}} \right)$$

$$C_b = w \cdot S_b \cdot w$$

Na podlagi ocen standardnih napak pa lahko izračunamo  $t$ -statistike kot  $t = b_i/se(b_i)$ .

```

> Sb<-sErr^2*solve(t(X) %*% X)
> Sb
      konstanta          F7          F21
konstanta  2151.921149 -108.83837598 -9.38010552
F7         -108.838376   8.40745537  0.03971066
F21        -9.380106    0.03971066  0.22726613
O1F2Zenski -95.163435         0.71978854  0.38619681
F5Predmestje -395.835418      2.28029391  0.10262743
F5Manjše mesto -289.613230     -2.52450622 -1.06289703
F5Vas        -360.299394    4.23246067 -1.43584413
F5Kmetija    -575.035961    17.20043074 -0.73922389
F7centGndrZenski 104.849908     -8.23715261 -0.12557392
      O1F2Zenski F5Predmestje F5Manjše mesto
konstanta    -95.1634352 -395.8354175   -289.613230
F7            0.7197885   2.2802939     -2.524506
F21           0.3861968   0.1026274     -1.062897
O1F2Zenski   138.5811962   9.0741196      3.802999
F5Predmestje  9.0741196   604.0986074   357.508039
F5Manjše mesto  3.8029995   357.5080389   509.585361
F5Vas         1.3810615   359.1427268   366.193057
F5Kmetija     42.3857670   366.4741109   363.085097
F7centGndrZenski -1.3605987   0.5596501     8.301651
      F5Vas      F5Kmetija F7centGndrZenski
konstanta    -360.299394 -575.0359613   104.8499077
F7            4.232461   17.2004307    -8.2371526
F21          -1.435844   -0.7392239    -0.1255739
O1F2Zenski    1.381062   42.3857670    -1.3605987
F5Predmestje  359.142727   366.4741109    0.5596501
F5Manjše mesto  366.193057   363.0850975    8.3016508
F5Vas         448.811895   377.9859757    2.1067811
F5Kmetija     377.985976   720.8876587    -8.4195178
F7centGndrZenski  2.106781   -8.4195178    14.7728263
> #kar je enako kot
> #vcov(fitG91_F7F21O1F2F5int2)
> max(abs(Sb-vcov(fitG91_F7F21O1F2F5int2)))
[1] 1.364242e-12
> seb<-sqrt(diag(Sb))
> seb

```

```

      konstanta          F7          F21
      46.3888041        2.8995612        0.4767244
      01F2Zenski      F5Predmestje    F5Manjše mesto
      11.7720515        24.5784175        22.5739974
      F5Vas          F5Kmetija F7centGndrZenski
      21.1851810        26.8493512        3.8435435
> #enakost z izračunom s funkcijo lm bomo preverili kasneje
>
> w<-diag(1/seb)
> Cb<-w %*% Sb %*% w
> #kar je enako kot
> #cov2cor(vcov(fitG91_F7F2101F2F5int2))
> max(abs(Cb-cov2cor(vcov(fitG91_F7F2101F2F5int2))))
[1] 1.44329e-15
> #t-statistika
> tb<-b/seb
> pb<-2*pt(-abs(tb),df=n-k)
> #tabela
> cbind(b=as.vector(b), "Std. napaka"=as.vector(seb),
        t=as.vector(tb), "p-vrednost"=as.vector(pb))
      b Std. napaka          t    p-vrednost
[1,] -50.52516310  46.3888041 -1.0891671826  2.769457e-01
[2,]  24.55524459   2.8995612  8.4686070004  1.087724e-15
[3,]  -0.20132728   0.4767244 -0.4223137835  6.730955e-01
[4,] -32.64379964  11.7720515 -2.7729915791  5.898550e-03
[5,] -16.07824506  24.5784175 -0.6541611171  5.135042e-01
[6,]   8.29629531  22.5739974  0.3675155600  7.134911e-01
[7,]  -7.01999244  21.1851810 -0.3313633444  7.405991e-01
[8,]  -0.02074509  26.8493512 -0.0007726479  9.993840e-01
[9,]  -5.63564466   3.8435435 -1.4662627672  1.436137e-01
> #kar je enako kot
> summary(fitG91_F7F2101F2F5int2)$coef
      Estimate Std. Error      t value
(Intercept)  -50.52516310  46.3888041 -1.0891671826
F7            24.55524459   2.8995612  8.4686070004
F21          -0.20132728   0.4767244 -0.4223137835
01F2zenski   -32.64379964  11.7720515 -2.7729915791
F5predmestje -16.07824506  24.5784175 -0.6541611171
F5manjse mesto  8.29629531  22.5739974  0.3675155600
F5vas        -7.01999244  21.1851810 -0.3313633444
F5kmetija    -0.02074509  26.8493512 -0.0007726479
F7centGndrZenski -5.63564466  3.8435435 -1.4662627672
      Pr(>|t|)

```

(Intercept)	2.769457e-01
F7	1.087724e-15
F21	6.730955e-01
01F2zenski	5.898550e-03
F5predmestje	5.135042e-01
F5manjse mesto	7.134911e-01
F5vas	7.405991e-01
F5kmetija	9.993840e-01
F7centGndrZenski	1.436137e-01

Seveda bi se na tak način dalo izračunati še marsikaj, a bomo tu zaključili.

### 3.4 Viri za poglobljanje znanja

Za poglobljanje snovi iz tega poglavja so uporabni tudi spletni viri, navedeni v podpodpoglavju 1.9.1 (Spletni viri), a jih tu posebej ne navajam. Tu izpostavljam samo enega, s pomočjo katerega je mogoče zelo enostavno in hitro najti ustrezne ukaze za izvedbo zelenih analiz.

**Quick-R** Odličen hitri vodič po **R**-ju za uporabnike drugih statističnih paketov (SPSS, SAS, Stata ...), se pravi za tiste, ki statistiko že znajo. URL: <http://www.statmethods.net/>

Za poglobljanje znanja iz analize variance in linearne regresije je primerna večina učbenikov s teh področij, na primer:

- Fox, John. 2008. *Applied regression analysis and generalized linear models*. Los Angeles: Sage.
- Košmelj, Blaženka. 2005. *Analiza odvisnosti za vzorčne podatke*. Ljubljana: Ekonomska fakulteta. Slovenski učbenik, ki ga uporabljajo na Ekonomski fakulteti Univerze v Ljubljani
- Iversen, Gudmund R. in Helmut Norpoth. 2002. *Analysis of variance*. Newbury Park; London; New Delhi: Sage.

Vsebina tega poglavja je tudi vsaj deloma pokrita v nekaterih učbenikih, omenjenih v prejšnjem poglavju.

Izvedbo analize variance in linearne regresije v **R**-ju pa pokrivajo sledeči učbeniki:

- Faraway, Julian James. 2005. *Linear models with R*. Boca Raton: Chapman & Hall/CRC.<sup>24</sup>  
Najbolj priporočam prav ta vir.
- Dalgaard, Peter. 2002. *Introductory statistics with R*. New York: Springer.<sup>25</sup>
- Muenchen, Robert A. 2011. *R for SAS and SPSS users*. New York: Springer.<sup>26</sup>  
Še posebej primerna za tiste, ki že poznajo SPSS ali SAS, sicer pa zelo zgoščena obravnava.
- Verzani, John. 2005. *Using R for introductory statistics*. Boca Raton: Chapman & Hall/CRC.<sup>27</sup>  
V bolj omejenem obsegu.

### 3.5 Vprašanja za ponavljanje

1. Katere so predpostavke klasične enofaktorske analize variance in kako jih preverjamo?
2. Analizo variance lahko izvedemo z več funkcijami. Katerimi? Kakšne so razlike med njimi oziroma katere so prednosti in slabosti posameznih funkcij?
3. Ali je vrstni red faktorjev pri večfaktorski analizi variance pomemben? Pojasnite!
4. V kakšni obliki morajo biti podatki, če želimo (s funkcijo `aov`) izvesti enofaktorsko analizo variance za odvisne vzorce?
5. Ali lahko, če na nekem podatkovju z manjkajočimi podatki večkrat zaženemo linearno regresijo z različnimi neodvisnimi spremenljivkami, pričakujemo, da so vse analize izvedene na istih enotah?
6. Kako v linearno regresijo s funkcijo `lm` vključimo nominalne spremenljivke? Ali jih moramo predhodno kaj transformirati? Na kaj moramo biti pri vključevanju pozorni?
7. Kako lahko s funkcijami za linearno regresijo vsaj približno ocenimo tudi nelinearne zveze? Kakšne so pomanjkljivosti tega pristopa?
8. Kako (s katero funkcijo in kaj uporabimo kot argument) narišemo standardne grafe za diagnostiko oziroma preverjanje predpostavk linearne regresije?

<sup>24</sup>Starejša in manj obsežna različica knjige je prostodostopna tudi na <http://cran.r-project.org/doc/contrib/Faraway-PRA.pdf>.

<sup>25</sup>Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>26</sup>Dostopna preko SpringerLink z računalnikov Fakultete za družbene vede Univerze v Ljubljani.

<sup>27</sup>Starejša in manj obsežna različica knjige je dostopna tudi na <http://www.math.csi.cuny.edu/Statistics/R/simpleR/printable/simpleR.pdf>.

9. Kako preverimo, ali neka nominalna spremenljivka z več kot dvema različnima vrednostma vpliva na odvisno spremenljivko, če izločimo vplive ostalih neodvisnih spremenljivk?
10. Ali lahko na podlagi standardnega izpisa ene analize ugotavljamo, ali so povprečne vrednosti katerihkoli dveh kategorij nominalne spremenljivke statistično značilno različne (pri izbrani stopnji tveganja) ob enakih vrednostih ostalih neodvisnih spremenljivk?
11. Kako v formuli (argumentu funkcije  $lm$ ) označimo, da želimo vključiti interakcijo?
12. Navedite vsaj en način ocenjevanja multikolinearnosti, ki pove tudi, pri katerih spremenljivkah se multikolinearnost pojavlja!
13. Kaj nam povedo grafikoni delnih ostankov (oziroma component + residual plot) oziroma za kakšen namen jih uporabljamo?



## 4. poglavje

### Za konec

V tem učbeniku je precej natančno predstavljena uporaba **R**-ja, tako za opravljanje splošnih opravil, ki se uporabljajo pri večini analiz (priprava podatkov, risanje grafov), kot tudi za izvedbo univariatnih in bivariatnih statističnih analiz ter multiple regresije. Sama statistična teorija je obravnavana le toliko, kolikor se mi je zdelo nujno potrebno.

Učbenik je sicer prvenstveno namenjen tistim, ki že poznajo osnove statistike, radi pa bi se naučili le-to izvajati s pomočjo **R**-ja. Poleg tega pa je namenjen tudi vsem, ki bi radi le spoznali osnovno delovanje **R**-ja, ne glede na to, na katerem področju ga nameravajo uporabljati. Za te je predvsem koristno prvo, najsplošnejše in tudi najobsežnejše poglavje. Naj učbenik zaključim s pregledom nekaterih področij, kjer se **R** lahko uporablja, ter navedbo nekaterih virov za uporabo **R**-ja na teh področjih:

**Multivariatna analiza** je običajno naslednja stopnja pri učenju statistike, ki sledi spoznavanju multiple regresije <sup>28</sup>. Kot že samo ime pove gre pri multivariatni analizi predvsem za sočasno obravnavo več spremenljivk. Najbolj znane metode multivariatne analize so razvrščanje v skupine, metoda glavnih komponent in faktorska analiza. Nekatere metode multivariatne analize so predstavljene tudi v Uvodnem primeru. Tukaj navajam par relevantnih učbenikov:

- Everitt, Brian. 2005. *An R and S-PLUS companion to multivariate analysis*. London: Springer.
- Everitt, Brian in Torsten Hothorn. 2011. *An introduction to applied multivariate analysis with R*. New York: Springer.

**Podatkovno rudarjenje** je ime za metode, katerih glavni cilj je odkrivanje uporabnega, veljavnega, nepričakovanega in razumljivega znanja iz podatkov. Pogosto se uporablja na ogromnih podatkih, pogosto ne-strukturiranih in pridobljenih iz različnih virov. Kljub navidezno velikim razlikam se v veliki meri

---

<sup>28</sup>Multipla regresija pravzaprav že sodi v področje Multivariatne analize.

prekriva s statistiko, še bolj pa s področjem strojnega učenja (nekateri smatrajo celo, da sta to le dve različni imeni za isto področje), zato ti področji omenjam skupaj. Nekaj učbenikov, ki obravnavajo podatkovno rudarjenje ali strojno učenje v **R**-ju:

- Williams, Graham J. 2011. *Data mining with Rattle and R: the art of excavating data for knowledge discovery*. New York: Springer.  
To še posebej priporočam za začetnike.
- Zhao, Yanchang in Yonghua Cen. 2013. *Data Mining Applications with R*, 1. izdaja. Amsterdam, Boston: Academic Press.
- Zhao, Yanchang. 2012. *R and Data Mining: Examples and Case Studies*, 1. izdaja. Amsterdam: Academic Press.
- Lantz, Brett. 2013. *Machine Learning with R*. Packt Publishing.

**Metode ponovnega vzorčenja** so metode, ki na podlagi ponovnega vzorčenja omogočajo preverjanje domnev ali izračun intervalov v primerih, ko se spremenljivke ne porazdeljujejo (niti približno) normalno, kar običajno predvidevajo klasične metode. Spodaj navajam tri učbenike Phillipa Gooda, ki pokrivajo uporabo metod ponovnega vzorčenja, še posebej v **R**-ju na različnih nivojih:

- Good, Phillip I. 2011. *Practitioners Guide to Resampling for Data Analysis, Data Mining, and Modeling*. Zany Books.  
Zraven je koda tako za **R** kot za Stato.
- ———. 2012. *Introduction to Statistics Through Resampling Methods and R*, 2. izdaja. Wiley.  
Še posebej primeren za začetnike in zelo vezan na **R**.
- ———. 2013. *Resampling Methods*, 3. izdaja. Birkhäuser.  
**R** je le eden od obravnavanih programskih jezikov/paketov.

**Statistična analiza finančnih podatkov** je tudi področje, kjer se pogosto uporablja **R**. Kot samo ime pove, gre za uporabo statistike in finančne matematike za analizo finančnih podatkov za (med drugim) upravljanje s tveganji, napovedovanje vrednosti finančnih kazalnikov, upravljanje portfelja ... Nekaj učbenikov, ki obravnavajo uporabo **R**-ja na tem področju:

- Carmona, René. 2013. *Statistical Analysis of Financial Data in R*, 2. izdaja. Springer New York.
- Daróczi, Gergely, Edina Berlinger, Péter Csóka, Daniel Havran, Márton Michaletzky, Zsolt Tulassay, Kata Váradi in Agnes Vidovics-Dancs. 2013. *Introduction to R for Quantitative Finance*. Packt Publishing.
- Arratia, Argimiro. 2014. *Computational Finance: An Introductory Course with R*. Atlantis Press.

**Biostatistika** se ukvarja s statistično analizo na področju biologije, medicine in sorodnih ved. Ukvarja se na primer z ugotavljanjem vplivov različnih dejavnikov na rast in razvoj rastlin in živali ter na njihove lastnosti, vplivom zdravil in načinov zdravljenja na potek bolezni in še bi lahko naštevali. Na tem področju sta **R** njemu soroden S-PLUS med najpogosteje uporabljenimi programskimi paketi. Nekaj knjig, ki obravnavajo metode s tega področja v **R**-ju:

- Broström, Göran. 2012. *Event History Analysis with R*. Boca Raton: CRC Press.
- MacFarland, Thomas W. 2013. *Introduction to Data Analysis and Graphical Presentation in Biostatistics with R*, 1. izdaja. Springer International Publishing.
- Shahbaba, Babak. 2011. *Biostatistics with R*, 1. izdaja. New York: Springer.

**Analiza ogromnih podatkov in spleta** postaja vse bolj pomembno področje. "Big data" (oz. ogromni podatki) je tako imenovani "buzzword", ki pridobiva na pomenu. Gre za analizo podatkov, ki jih ne moremo shraniti v delovni spomin računalnika, zato so za analizo potrebni posebni pristopi. Pogosto se podatki pridobijo neposredno na svetovnem spletu. Tudi sicer narašča zanimanje za analizo podatkov, ki so dostopni na spletu, še posebej na socialnih omrežjih. Tudi za ta namen se lahko uporabi **R**. Po eno delo z vsakega področja v povezavi z **R**-jem navajam spodaj:

- Danneman, Nathan in Richard Heimann. 2014. *Social Media Mining with R*. Packt Publishing.
- Prajapati, Vignesh. 2013. *Big Data Analytics with R and Hadoop*. Packt Publishing.

To seveda niso vsa področja, kjer se **R** uporablja, vendar pa se iz navedenega vidi, da je **R** res uporaben na zelo različnih področjih, lahko bi rekli, da je uporaben povsod, kjer se uporablja analiza podatkov. Učenje **R**-ja je torej koristno za vsakega, ki se namerava resno ukvarjati z analizo podatkov, ne glede na siceršnje področje dela. Upam, da bo pričujoč učbenik v pomoč vsaj pri začetnem spoznavanju tega res "močnega" programskega okolja.



# Literatura

- Arratia, Argimiro. 2014. *Computational Finance: An Introductory Course with R*. Atlantis Press.
- Broström, Göran. 2012. *Event History Analysis with R*. Boca Raton: CRC Press.
- Buuren, Stef in Karin Groothuis-Oudshoorn. 2011. MICE: Multivariate imputation by chained equations in R. *Journal of Statistical Software* 45 (3). URL <http://doc.utwente.nl/78938/>.
- Buuren, Stef van. 2012. *Flexible imputation of missing data*. Boca Raton, FL: CRC Press.
- Carmona, René. 2013. *Statistical Analysis of Financial Data in R*, 2. izdaja. Springer New York.
- Dalgaard, Peter. 2002. *Introductory statistics with R*. New York: Springer.
- Danneman, Nathan in Richard Heimann. 2014. *Social Media Mining with R*. Packt Publishing.
- Daróczi, Gergely, Edina Berlinger, Péter Csóka, Daniel Havran, Márton Michaletzky, Zsolt Tulassay, Kata Váradi in Agnes Vidovics-Dancs. 2013. *Introduction to R for Quantitative Finance*. Packt Publishing.
- Everitt, Brian. 2005. *An R and S-PLUS companion to multivariate analysis*. London: Springer.
- Everitt, Brian in Torsten Hothorn. 2011. *An introduction to applied multivariate analysis with R*. New York: Springer.
- Faraway, Julian James. 2005. *Linear models with R*. Boca Raton: Chapman & Hall/CRC.
- . 2006. *Extending the linear model with R: generalized linear, mixed effects and nonparametric regression models*. Boca Raton: Chapman & Hall/CRC.
- Ferligoj, Anuška. 1994. *Osnove statistike na prosojnicah*. Ljubljana: samozaložba Zenel Batagelj.

- Ferligoj, Anuška, Tina Kogovšek, Willem E. Saris, Germa Coenders in Valentina Hlebec. 2000. *Kakovost merjenja egocentričnih socialnih omrežij [datoteka podatkov]*. Ljubljana: Univerza v Ljubljani, Fakulteta za družbene vede, Center za metodologijo in informatiko [izdelava], 2000. Slovenija, Ljubljana: Univerza v Ljubljani, Arhiv družboslovnih podatkov [distribucija], 2012.
- Fox, John. 2008. *Applied regression analysis and generalized linear models*. Los Angeles: Sage.
- Gandrud, Christopher. 2013. *Reproducible research with R and RStudio*. Boca Raton: Chapman & Hall/CRC.
- Gelman, Andrew in Jennifer Hill. 2006. *Data Analysis Using Regression and Multi-level/Hierarchical Models*. Cambridge; New York: Cambridge University Press.
- Good, Phillip I. 2011. *Practitioners Guide to Resampling for Data Analysis, Data Mining, and Modeling*. Zany Books.
- . 2012. *Introduction to Statistics Through Resampling Methods and R*, 2. izdaja. Wiley.
- . 2013. *Resampling Methods*, 3. izdaja. Birkhäuser.
- Holm, Sture. 1979. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics* 6 (2): 65–70. URL <http://www.jstor.org/stable/4615733>.
- IBM. 2011. Sum of Squares (IBM SPSS help). URL <http://pic.dhe.ibm.com/infocenter/spssstat/v20r0m0/topic/com.ibm.ibmcom.doc/banner.htm>.
- Iversen, Gudmund R. in Helmut Norpoth. 2002. *Analysis of variance*. Newbury Park; London; New Delhi: Sage.
- Jesenko, Jože in Manca Jesenko. 2007. *Multivariatne statistične metode*. Kranj: Moderna organizacija.
- Johnson, Richard A in Dean W Wichern. 2007. *Applied multivariate statistical analysis*. Upper Saddle River: Pearson Prentice Hall, Pearson Education International.
- Kastelec, Damijana in Katarina Košmelj. 2009. *Statistična analiza podatkov s programoma Excel 2003 in R*. Ljubljana: Biotehniška fakulteta. URL [http://www.bf.uni-lj.si/fileadmin/groups/2763/%C5%A1tudijsko\\_gradivo/SAP\\_2003.pdf](http://www.bf.uni-lj.si/fileadmin/groups/2763/%C5%A1tudijsko_gradivo/SAP_2003.pdf).
- Košmelj, Blaženka. 2005. *Analiza odvisnosti za vzorčne podatke*. Ljubljana: Ekonomska fakulteta.
- Košmelj, Blaženka in Jože Rován. 2007. *Statistično sklepanje*. Ljubljana: Ekonomska fakulteta.

- Lantz, Brett. 2013. *Machine Learning with R*. Packt Publishing.
- Leisch, Friedrich. 2002. Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis. V *Compstat*, ur. Professor Dr Wolfgang Härdle in Profesor Dr Bernd Rönz, 575–580. Physica-Verlag HD. URL [http://link.springer.com/chapter/10.1007/978-3-642-57489-4\\_89](http://link.springer.com/chapter/10.1007/978-3-642-57489-4_89). DOI: 10.1007/978-3-642-57489-4\_89.
- Levin, Jack, James Alan Fox in David R. Forde. 2013. *Elementary Statistics in Social Research (12th Edition)*. Pearson.
- MacFarland, Thomas W. 2013. *Introduction to Data Analysis and Graphical Presentation in Biostatistics with R*, 1. izdaja. Springer International Publishing.
- Matloff, Norman. 2011. *The Art of R Programming: A Tour of Statistical Software Design*. San Francisco: No Starch Press.
- Minium, Edward W., Robert C. Clarke in Theodore Coladarci. 1999. *Elements of statistical reasoning*. New York: Wiley.
- Muenchen, Robert A. 2011. *R for SAS and SPSS users*. New York: Springer.
- Murrell, Paul. 2011. *R graphics*. Boca Raton: CRC Press.
- Prajapati, Vignesh. 2013. *Big Data Analytics with R and Hadoop*. Packt Publishing.
- Rubin, Donald B. 1987. *Multiple imputation for nonresponse in surveys*. New York: Wiley.
- . 1996. Multiple Imputation After 18+ Years. *Journal of the American Statistical Association* 91 (434): 473–489. doi: 10.2307/2291635. URL <http://www.jstor.org/stable/2291635>.
- Shahbaba, Babak. 2011. *Biostatistics with R*, 1. izdaja. New York: Springer.
- Snijders, Tom A. B. in Roel J. Bosker. 2012. *Multilevel analysis: an introduction to basic and advanced multilevel modeling*. Los Angeles: Sage.
- Tabachnick, Barbara G. in Linda S. Fidell. 2007. *Using multivariate statistics*. Boston: Pearson/Allyn & Bacon.
- Toš, Niko, Brina Malnar in skupina. 2004. *Slovensko javno mnenje 2004/2: Evropska družboslovna raziskava [datoteka podatkov]*. Slovenija, Ljubljana: Fakulteta za družbene vede, Center za raziskovanje javnega mnenja in množičnih komunikacij [izdelava], 2004. Slovenija, Ljubljana: Univerza v Ljubljani, Arhiv družboslovnih podatkov [distribucija], 2009.
- Venables, William N. in Brian D. Ripley. 2000. *S programming*. New York: Springer.
- Verzani, John. 2005. *Using R for introductory statistics*. Boca Raton: Chapman & Hall/CRC.

- Welch, B. L. 1951. On the Comparison of Several Mean Values: An Alternative Approach. *Biometrika* 38 (3/4): 330–336. doi: 10.2307/2332579. URL <http://www.jstor.org/stable/2332579>.
- Wickham, Hadley. 2009. *ggplot2: Elegant Graphics for Data Analysis*. Dordrecht; New York: Springer. URL <http://public.eblib.com/EBLPublic/PublicView.do?ptiID=511468>.
- Wilkinson, Leland in Graham Wills. 2005. *The grammar of graphics*. New York: Springer.
- Williams, Graham J. 2011. *Data mining with Rattle and R: the art of excavating data for knowledge discovery*. New York: Springer.
- Wonnacott, Thomas H in Ronald J. Wonnacott. 1990. *Introductory statistics*. New York: Wiley.
- Wooldridge, Jeffrey M. 2002. *Econometric analysis of cross section and panel data*. Cambridge, Massachusetts: MIT Press.
- Xie, Yihui. 2013. *Dynamic report generation with r and knitr*. Boca Raton: Chapman & Hall Crc.
- Zhao, Yanchang. 2012. *R and Data Mining: Examples and Case Studies*, 1. izdaja. Amsterdam: Academic Press.
- Zhao, Yanchang in Yonghua Cen. 2013. *Data Mining Applications with R*, 1. izdaja. Amsterdam, Boston: Academic Press.
- Zuur, Alain F., Elena N. Ieno in Erik H.W.G. Meesters. 2009. *A Beginner's Guide to R*. New York: Springer.