

# **Upoznavanje sa sintaksom jezika R i njegova primjena u osnovnoj statističkoj i grafičkoj analizi podataka**

S200



Sveučilište u Zagrebu  
Sveučilišni računski centar

Ovu inačicu priručnika izradio je autorski tim Srca u sastavu:

Autor: dr.sc. Andreja Radović

Recenzent: mr.sc. Melita Perčec-Tadić

Urednik: Sabina Rako

## TEČAJEVISrca

Sveučilište u Zagrebu

Sveučilišni računski centar

Josipa Marohnića 5, 10000 Zagreb

[edu@srce.hr](mailto:edu@srce.hr)

Verzija priručnika: S200-ver2



Ovo djelo dano je na korištenje pod licencom Creative Commons Imenovanje-Nekomercijalno-Dijeli pod istim uvjetima 4.0 međunarodna. Licenca je dostupna na stranici: <http://creativecommons.org/licenses/by-nc-sa/4.0/>.

## Sadržaj

<b>1 Uvod u jezik R</b>	<b>1</b>
1.1 Neke prednosti sustava R . . . . .	1
1.2 Instalacija sustava R . . . . .	3
1.2.1 Nadogradanja sustava R kontribuiranim paketima . . . . .	3
1.2.1.1 Izbor repozitorija (spremišta) paketa . . . . .	4
1.2.1.2 Instalacija paketa sa spremišta (engl. <i>repository</i> ) . . . . .	5
1.2.1.3 Instalacija paketa putem lokalne datoteke . . . . .	5
1.2.1.4 Instalacija paketa s osobnih Git/GitHub spremišta . . . . .	5
1.2.2 Pretraživanje lokalno instaliranog sustava (engl. <i>offline</i> ) . . . . .	6
1.2.3 Pretraživanje globalnog sustava (engl. <i>online</i> ) . . . . .	7
1.2.3.1 Traženje dokumentacije o instaliranim paketima . . . . .	7
1.2.3.1.1 Obavezna dokumentacija paketa na CRAN-u . . . . .	7
1.2.3.1.2 Opcionalna dokumentacija paketa na CRAN-u . . . . .	7
1.2.3.2 Komunikacijske liste u sustavu R . . . . .	8
1.2.3.3 Grupe specijalnog interesa . . . . .	8
1.2.3.4 Specijalizirane konferencije . . . . .	9
1.2.3.5 Ostala literatura o sustavu R . . . . .	9
1.2.4 Određivanje okoline za rad . . . . .	10
1.2.5 Stvaranje individualiziranog sustava . . . . .	10
1.2.6 Skupovi podataka za vježbu . . . . .	10
1.2.7 Početak rada . . . . .	11
1.2.7.1 Zadavanje naredbi . . . . .	11
1.2.7.2 Postavljanje radnog direktorija . . . . .	12
1.2.7.3 Neke osnovne funkcije za rad u sustavu: . . . . .	12
<b>2 Osnovne vrste podataka u sustavu R</b>	<b>14</b>
2.0.0.1 Atomski vektori . . . . .	15
2.0.0.2 Indeksiranje i selekcija podskupa vektora . . . . .	16
2.0.0.2.1 Selekcija putem indeksa (položaja elementa u objektu) . . . . .	16
2.0.0.2.2 Selekcija logičkim uvjetima . . . . .	17
2.1 Opcionalni atributi objekata u sustavu R . . . . .	18
2.1.1 Opcionalni atributi na vektorima, klasa objekta <i>vector</i> . . . . .	18
2.1.2 Liste . . . . .	19
2.1.2.1 Atributi objekata klase <i>list</i> . . . . .	21
2.1.3 Indeksiranje i selekcija podskupa matrice . . . . .	24
2.2 Skupovi podataka . . . . .	25
2.3 Učitavanje postojećih podataka u R . . . . .	27
2.3.1 Funkcije u sustavu R . . . . .	28
2.3.2 Spajanje argumenata funkcija . . . . .	29
2.3.3 Operatori u sustavu R . . . . .	31
2.3.4 Distribucije vjerojatnosti u sustavu R . . . . .	34
2.3.4.1 Funkcija gustoće <i>d()</i> . . . . .	34
2.3.4.2 Funkcija vjerojatnosti <i>p()</i> . . . . .	34
2.3.4.3 Funkcija kvantila <i>q()</i> . . . . .	34
2.3.4.4 Funkcija generiranja slučajne varijable prema zadanoj raspodjeli <i>r()</i> . . . . .	34
2.3.4.5 Dodatne funkcije nužne za početak rada . . . . .	37
2.3.5 Osnove o datumima u sustavu R . . . . .	39
2.3.5.1 Format zapisa datumskih varijabli . . . . .	39
2.3.5.2 Klase za čuvanje datumskih varijabli unutar sustava R . . . . .	40
<b>3 Uvod u grafičke sustave u R-u</b>	<b>42</b>
3.1 Grafika osnovnog sustava (engl. <i>base</i> ) . . . . .	42
3.1.1 Grafički uređaji . . . . .	45
3.2 <i>Lattice</i> (osnova paket <i>grid</i> ) . . . . .	64
<b>4 Uvod u statistiku uz sustav R</b>	<b>81</b>
4.1 Osnovni pojmovi u statistici . . . . .	81

4.1.1	Statistička populacija . . . . .	81
4.1.2	Statističko zaključivanje . . . . .	83
4.1.3	Uzorak . . . . .	83
4.1.3.1	Uzorci bez definirane vjerojatnosti (ne-vjerojatnosni uzorci) . . . . .	83
4.1.3.2	Vjerojatnosni uzorci . . . . .	83
4.1.3.2.1	Jednostavan slučajni uzorak . . . . .	83
4.1.3.2.2	Sistematski (sustavni) uzorak . . . . .	85
4.1.3.2.3	Stratificirani uzorak . . . . .	85
4.1.3.2.4	Klaster uzorak . . . . .	88
4.1.3.2.5	Kombinirani uzorak . . . . .	88
4.1.3.3	Zavisnost i nezavisnost dva uzorka . . . . .	88
4.1.3.3.1	Nezavisni uzorci . . . . .	90
4.1.3.3.2	Zavisni uzorci . . . . .	90
4.1.4	Varijable . . . . .	91
4.1.4.1	Mjerenje varijabli . . . . .	91
4.1.4.1.1	Tipovi mjernih skala . . . . .	91
4.1.5	Podaci . . . . .	97
4.2	Deskriptivna vs. inferencijalna statistika . . . . .	97
4.2.1	Deskriptivna statistika . . . . .	98
4.2.1.1	Distribucija frekvencija jedne varijable . . . . .	98
4.2.2	Grafičke metode za prikaz kvantitativnih varijabli . . . . .	101
4.2.2.1	Grafički prikazi distribucije frekvencije jedne varijable . . . . .	101
4.2.2.1.1	Histogram . . . . .	101
4.2.2.1.2	Stupčasti dijagram . . . . .	101
4.2.2.1.3	Strukturni dijagram (engl. <i>pie</i> ) . . . . .	104
4.2.2.1.4	Graf stabljika - list (engl. <i>stem and leaf</i> ) . . . . .	104
4.2.2.1.5	Grafički prikaz kutija i brk (engl. <i>Box-Whisker</i> ) . . . . .	106
4.2.2.1.6	Kumulativne relativne frekvencije . . . . .	109
4.2.3	Deskriptivna statistika - numeričko sumiranje podataka . . . . .	109
4.2.3.1	Mjere centralne tendencije . . . . .	110
4.2.3.2	Medijan . . . . .	110
4.2.3.3	Mod . . . . .	111
4.2.4	Mjere varijabilnosti podataka . . . . .	113
4.2.4.1	Devijacija . . . . .	113
4.2.4.2	Varijanca . . . . .	113
4.2.4.2.1	Varijanca populacije nasuprot varijanci uzorka . . . . .	113
4.2.4.3	Standardna devijacija . . . . .	114
4.2.4.4	Raspon podataka . . . . .	114
4.2.4.5	Interkvartilni raspon (IQR) . . . . .	114
4.2.4.6	Koeficijent varijacije . . . . .	115
4.2.4.7	Mjere relativnog položaja . . . . .	115
4.3	Slučajna varijabla . . . . .	119
4.3.1	Kontinuirana slučajna varijabla . . . . .	121
4.3.1.1	Neke važne kontinuirane distribucije . . . . .	121
4.3.1.1.1	Normalna (Gaussova) distribucija . . . . .	121
4.3.1.1.2	Z vrijednost (Z skor) . . . . .	123
4.3.2	Diskretna slučajna varijabla . . . . .	130
4.3.2.1	Neke važne diskretne distribucije . . . . .	130
4.3.2.1.1	Binomna distribucija . . . . .	130
4.3.2.1.2	Poisson distribucija . . . . .	134
4.3.2.2	Studentova <i>t</i> distribucija . . . . .	142
4.3.2.3	Fisherova F distribucija . . . . .	145
4.3.3	Distribucija statistike uzorka (engl. <i>sampling</i> distribucija) . . . . .	146
4.3.3.1	Centralni granični teorem (engl. <i>Central Limit Theorem - CLT</i> ) . . . . .	146
4.3.3.2	Ponašanje distribucije sredina za različito velike uzorke . . . . .	148



Tečaj *Upoznavanje sa sintaksom jezika R i njegova primjena u osnovnoj statističkoj i grafičkoj analizi podataka* (S720) obrađuje osnove rada u okruženju programskog jezika R te njegovo korištenje u pripremi, vizualizaciji i statističkoj analizi podataka. Ovaj tečaj osmišljen je s ciljem upoznavanja sudionika s mogućnostima sustava, dajući generalni pregled tema i područja važnih za osobe koje se bave znanostima o podacima. Tečaj je namijenjen najširoj publici i raznolikim strukama. Pojedine će teme sudionicima biti od većeg ili manjeg značaja za njihovo specifično područje, a proizašle su iz višegodišnjeg iskustva autorice tečaja kao nužne. Polaznicima je pokušano dati što bolji temelj i informacije o dijelovim sustava o kojima je barem bitno znati da postoje kako bi se njihove te funkcionalnosti potražile i koristile prema potrebi. Savladavanjem gradiva tečaja polaznici će u potpunosti biti u mogućnosti pratiti nadogradnje putem drugih tečajeva Srca vezanih uz sustav R i manipulacije podacima te reproducibilno istraživanje.

Tečaj je namijenjen polaznicima koji se po prvi puta susreću kako s okruženjem programskog jezika R tako i s analizama podataka.

Za pohađanje ovog tečaja potrebno je poznavanje osnova rada s računalom i operacijskim sustavom MS Windows (iako većina napisanog vrijedi i za npr. Ubuntu sustav) te poznavanje osnova rada na Internetu. Također, pasivno poznavanje engleskog jezika nužno je u radu sa sustavom R.

Preporučamo samostalno upoznavanje s materijelima on-line tečaja Srca "Zašto učiti R?" na [lms.srce.hr](http://lms.srce.hr).

Minimalno iskustvo programiranja je prednost.

## 1 Uvod u jezik R

R je programski jezik i okruženje za statističke izračune i vizualizaciju. R je slobodan programski jezik (engl. free) što znači da se može slobodno koristiti i distribuirati te da je otvorenog kôda (engl. *open-source*). Izraz "okruženje" ističe da je R dobro planiran i konzistentan sustav, a ne sustav koji se postepeno dopunjava specifičnim i nefleksibilnim programskim alatima, što je često slučaj kod drugih programa za analizu podataka.

Program R pruža širok izbor statističkih metoda za linearno i nelinearno modeliranje, klasične statističke testove, analize vremenskih serija, klasteriranje. Lako je proširiv s velikim izborom grafičkih tehnika.

R je implementacija jezika S kojeg je razvio John Chambers s kolegama u Bell Laboratories te Robert Gentleman sa sveučilišta u Aucklandu, Novi Zeland. Ista grupa razvila je i jezik C i UNIX. Trenutno jezik R razvija tzv. *jezgra*, osnovna grupa za razvoj jezika R (engl. *R Core Team*). Postoje neke važne razlike, ali mnogi programi (kôd) napisani u jeziku S rade nepromijenjeni i u jeziku R.

R je proizvod aktivnog pokreta među statističarima koji ima cilj stvaranje moćnog, programabilnog, prijenosnog, otvorenog računalnog okruženja primjenjivog pri rješavanju većine kompleksnih problema, ali i provedbi rutinskih analiza.

Statističari su razvili stotine specijaliziranih statističkih procedura za širok raspon uporabe putem tzv. pridodanih paketa (engl. *contributed packages*) koji su slobodno dostupni i integrirani direktno sa sustavom R. Donedavno je postojalo uvriježeno mišljenje da sustav R koristi uglavnom akademska zajednica dok je za analize podataka u tvrtkama standard sustav SAS. No to se danas promjenilo. Sustav R je npr. Federalna agencija za lijekove (Federal Drug Administration - FDA) identificirala je dijelove R-a pogodnim za tumačenje podataka iz kliničkih istraživanja.

Također, velik je broj kompanija koje prelaze na sustav R za analizu i prezentaciju svojih podataka. Neke od njih nalaze se na popisu koji se može pronaći na sljedećoj poveznici:<http://www.listendata.com/2016/12/companies-using-r.html>.

Usmjerenje baza podataka tvrtke ORACLE na povezivanje sa sustavom R ([https://blogs.oracle.com/R/entry/announcing\\_oracle\\_r\\_enterprise\\_1](https://blogs.oracle.com/R/entry/announcing_oracle_r_enterprise_1)) (<http://www.oracle.com/technetwork/topics/bigdata/r-offerings-1566363.html>) te prvi puta stavljanje sustava R na Gartnerov magični kvadrant uvjerilo je i najveće skeptike u vjerodostojnost sustava R kao relevantnog sustava za analitiku velikih skupova podataka, naprednu analitiku i vizualizaciju

Kusum i Legović (2004) navode značajnu razliku u filozofiji između sustava SAS i R: u sustavu S (R) statistička se analiza obično provodi u nizu koraka pohranjujući međurezultate pojedinih koraka u objekte. Na primjer, neki poznatiji komercijalni paketi kao što su SAS i SPSS će dati obilne rezultate iz regresijske ili diskriminantne analize dok će sustav R dati minimalan rezultat i pohraniti rezultate u odgovarajuće objekte za kasnija ispitivanja drugim funkcijama sustava R.

### 1.1 Neke prednosti sustava R

Neke od prednosti sustava R su:

R je dostupan kao slobodan softver pod uvjetima Free Software Foundation GNU Opće javne licence u obliku izvornog kôda ([http://en.wikipedia.org/wiki/GNU\\_Project](http://en.wikipedia.org/wiki/GNU_Project)):

- Sloboda uporabe programa u bilo koje svrhe - sloboda 0
- Sloboda proučavanja načina rada programa i mogućnost prilagodbe vlastitim potrebama - sloboda 1
- Sloboda kopiranja i distribucije programa - preduvjet za to je slobodan kôd (engl. *free-source*) sloboda 2
- Sloboda poboljšavanja programa i mogućnost dijeljenja poboljšanog programa u zajednici - sloboda 3
- Dostupan je putem Interneta
- Radi na raznim platformama UNIX i sličnim sustavima (uključujući FreeBSD i Linux), Windows i Mac OS
- Proizvod je međunarodne suradnje vrhunskih statističara i dizajnera programskih jezika
- Dozvoljava statističke analize i vizualizaciju te njihovo neograničeno unaprjeđenje
- Imo mogućnost rada na velikim i kompleksnim objektima - ograničenja su vezana uz operacijski sustav koji koristimo, a ne sustav R.
- Postoji vrlo dobra tehnička dokumentacija o paketima i njihovim funkcijama te velik broj udžbenika i priručnika (<https://cran.r-project.org/manuals.html>) koje, između ostalih, pripremaju sami korisnici. Također, postoji i velik broj knjiga koje objašnjavaju kako provesti pojedine vrste analiza koristeći sustav R. Poveznica na glavnu dokumentacijsku stranicu sustava je <https://www.r-project.org/other-docs.html>.
- Svaki korak u analizi je sačuvan i takva povijest (.Rhistory datoteka) može se naknadno koristiti u novim analizama ili kao dokumentacija postojećoj.
- Radi svoje organizacije potiče kritičko razmišljanje tijekom provedbe analize, za razliku od čestog poticanja izvršavanja analiza pritiskom na jednu tipku.
- U potpunosti je programabilan i u podlozi nosi vrlo sofisticiran jezik S.
- Pisanjem skripti i korisničkih funkcija analize se jednostavno automatiziraju. Na taj način je i pojednostavljeni priprema korisničkih paketa i njihovo stavljanje zajednici na uporabu i provjeru.
- Programski kôd svake funkcije je javan i vidljiv je svaki korišteni algoritam. Uz to, stručnjaci - statističari osiguravaju korektnost ponuđenih funkcija.

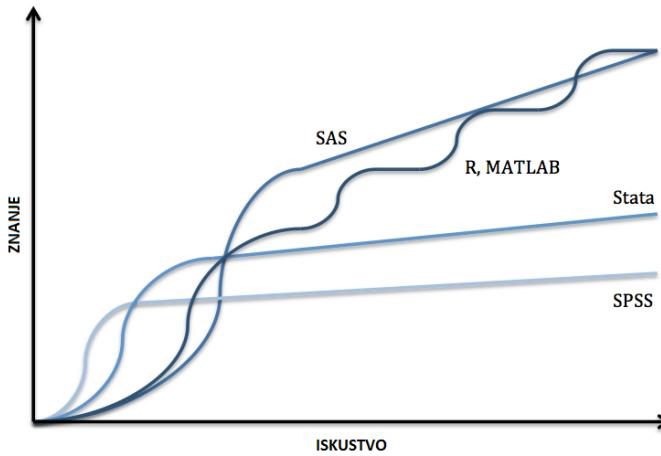
## ##Neki nedostaci sustava R

Veoma se često navode neki od nedostataka sustava R kao što su:

-R nije jednostavan za početnike i nemali broj osoba koje susame započinjale edukaciju su u tome odustale. Ipak, nakon početnog usvajanja znanja o osnovnom načinu funkcioniranja sustava i strukturama podataka, linija učenja je za R veoma strma i slične je ostalim statističkim paketima.

- U svojoj suštini temelji se na 40 godina staroj tehnologiji
- Postoji slaba podrška analizama putem grafičkih sučelja (engl. *Graphical User Interface - GUI*). Za rad u R-u ulavnom je nužno je pisanje naredbi kako bi se provele analize i napravila vizualizacija rezultata. Mnogi korisnici ovo ne smatraju nedostatkom radi uspostave potpune kontrole nad sustavom. Ipak, noviji projekti rješavaju i problem korisnika koji ne žele učiti R sintaksu preporučamo upoznavanje s projektom Deducer (<http://www.deducer.org/pmwiki/index.php?n>Main.DeducerManual?from>Main.HomePage>). Od starijih rješenja postoji paket Rcmdr koji u određenoj mjeri omogućava način rada GUI i određen skup analiza.
- Korisnik odlučuje o svakom koraku analize i provodi je po koracima. Ovakav način rada omogućuje prednosti kao što je čuvanje tijeka analize (engl. *processing log*) koji se koristi u svrhu izvještavanja ili ponavljanja analize. Mnogi navedeno smatraju prednošću.
- Korisnik treba prihvati novi način razmišljanja o podacima koje analizira, te o objektima i metodama. Mnogi ovo smatraju prednošću jer klasa objekta određuje koje metode ima smisla primjeniti
- U velikom broju slučajeva korisnik treba učiti jezik S/R kako bi bio u mogućnosti specificirati analizu, iako danas postoji velik broj projekta i tvrtki koje nude različite oblike podrške korisnicima sustava R (npr. bivši Revolution Analytics, sada Microsoft) ili mogućnost on-line analize u oblaku (engl. *cloud*), kao npr. StatAce.





Slika 1: Krivulja učenja najčešćih statističkih programa i paketa; izvor: <https://sites.google.com/a/nyu.edu/statistical-software-guide/summary>

Ipak, postoje novi pristupi koji omogućavaju interaktivno učenje različitih softvera pa tako i R-a putem, primjerice *Docker* sustava (projekt *Rocker* <https://github.com/rocker-org/>), u oblaku (engl. *cloud computing*) ili lokalno korištenjem vitualnih strojeva. Ovakav pristup jedan je od ključnih elemenata reproducibilne znanosti jer omogućava pohranu sustava kakav je bio u trenutku izrade analiza. O ovoj temi u narednim tečajevima i radionicama Srca.

## 1.2 Instalacija sustava R

Kako je već spomenuto, R je sustav koji radi na različitim platformama, kao što su UNIX, Windows, Mac OS te njihovim operacijskim sustavima. Prilikom instalacije programskog okruženja treba pronaći odgovarajući kôd te pratiti upute o instalaciji. S web-stranice <http://cran.r-project.org/bin/> potrebno je odabrati verziju kompatibilnu operacijskom sustavu na računalu. Instalacija na operacijski sustav Windows je klasična i ovom prilikom nećemo je detaljnije opisivati.

Napomenut ćemo da velik broj istraživača priprema i vlastite pakete koji su dostupni jedino putem njihovih osobnih Github spremišta.

###Programski paketi koji dolaze s instalacijom sustava R

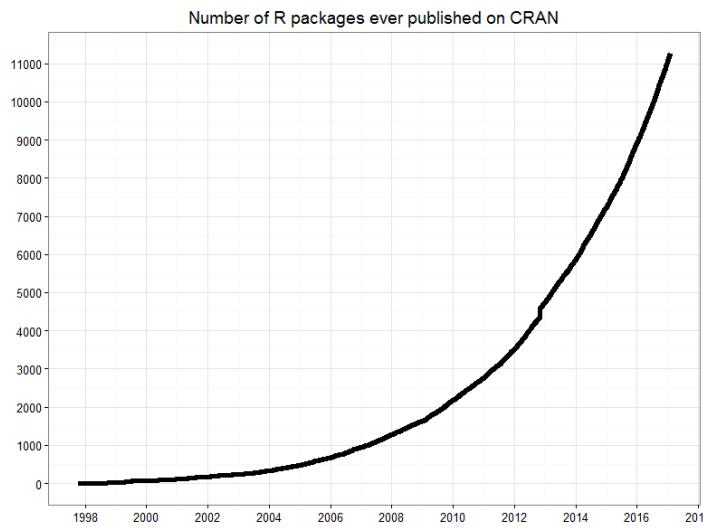
Određene osnovne funkcionalnosti sustava R dobivaju se sa sustavom instalacijom. To su tzv. osnovni paketi koji podržavaju programski jezik S, osnovnu statistiku i osnovnu (oznaka prioritea - engl. *priority base*) grafiku. Neki od paketa uključeni su u instalaciju sustava R su: *base*, *compiler*, *datasets*, *graphics*, *grDevices*, *grid*, *methods*, *parallel*, *splines*, *stats*, *stats4*, *tcltk* i *utils*. Ovaj skup paketa ne mijenja se između dvije verzije sustava R i svi nose oznaku temeljnih paketa (*priority="base"*). Pakete iz ove grupe nije moguće pronaći neovisne na spremištu CRAN. Drugi skup paketa sustava je dio i same instalacije sustava R, ali se njihove verzije mogu nadograđivati između pojedinih verzija i kao takvi se mogu i pojedinačno pronaći na spremištu CRAN (oznaka prioritea - engl. *priority recommended*). Neki od njih su *KernSmooth*, *MASS*, *Matrix*, *boot*, *class*, *cluster*, *codetools*, *foreign*, *lattice*, *mgcv*, *nnet*, *rpart*, *spatial*, *survival*.

Paket *base* (library (*base*)) čini okosnicu sustava R sadržavajući osnovne funkcije: aritmetika, ulaz i izlaz podataka, osnovne programske potpore, itd. Nešto kasnije upoznat ćemo se detaljnije s nekim od funkcija paketa *base*. Radi lakšeg snalaženja, mnogobrojni paketi dostupni unutar spremišta CRAN grupirani su u manji broj tzv. spaktara aplikacija (engl. *Task Views*), čiji se popis se može pronaći na stranici <http://cran.r-project.org/web/views>.

### 1.2.1 Nadogradnja sustava R kontribuiranim paketima

Mogućnosti jezika R proširene su putem paketa koje razvijaju korisnici i time omogućuju uporabu specijaliziranih statističkih tehniki, grafičkih uređaja, uvoz/izvoz s preko 11000 dodatnih paketa (stanje srpanj 2017.) dostupno je na sveobuhvatnoj arhivi sustava R (CRAN: The comprehensive R Archive Network), Bioconductor (repozitorij paketa za analizu genomske podataka) i drugim spremištima (engl. *repository*). Budući se svi paketi neophodni za praćenje ovog tečaja nalaze na spremištu CRAN, ovaj će se izraz u dalnjem tekstu odnositi upravo na ovo spremište.





Slika 2: Broj R paketa na CRAN repozitoriju; izvor: <http://blog.revolutionanalytics.com/2017/01/cran-10000.html>

Da bi se neki paket mogao staviti na CRAN spremište neophodno je da zadovolji konzistentnost rada na različitim operacijskim sustavima, prođe provjeru korektnosti zapisa o potrebama provjera paketa na kojima se ovaj paket temelji (engl. *dependency*), provjere inverzne ovisnosti o neštetnosti paketa na već postojeće (engl. *reverse dependency*) te da ima pripremljenu obaveznu dokumentaciju o funkcionalnostima koje paket nudi, tzv. referentni priručnik (engl. *reference manual*). Svaki paket koji je prošao provjere i pohranjen je na spremištu CRAN ima svoju web stranicu. Upoznat ćemo se s izgledom stranice nekog paketa na primjeru paketa *akima*. Iako je moguće zatražiti informacije iz sustava, ovaj puta ćemo u web pretraživač unijeti upit, primjerice "package akima in r". Tražilice će nam na ovako postavljen upit redovno kao prvu poveznicu nuditi web stranicu formata (za paket *akima*) <https://cran.r-project.org/web/packages/akima/index.html>. Upoznajmo se sa strukturu stranice te informacijama koje pruža.

Jednako vrijedi i za sve ostale pakete na spremištu.

Mogućnosti sustava možemo nadograditi instalacijom dodatnih paketa funkcija te unošenjem njihovih funkcionalnosti u sustav. To radimo u sljedećih nekoliko koraka:

**1.2.1.1 Izbor repozitorija (spremišta) paketa** Naredbom `setRepositories()` odabiremo na kojim spremištima paketa želimo pretraživati funkcionalnosti. Do istog dijaloga može se doći putem padajućeg izbornika na konzoli sustava R Packages -> Select repositories.

Funkcije za izbor spremišta (repositorija) paketa te izbor zrcalnih stranica (engl. *mirror pages*).

`setRepositories()`

Uz pomoć naredbe `chooseCRANmirror()` možemo izabrati željenu zrcalnu stranicu spremišta CRAN. Uz pomoć naredbe `getCRANmirrors()` možemo doći do informacija gdje se sve nalaze zrcalne stranice spremišta CRAN. Ova funkcija bila je znatno važnija u ranijim vremenima znatno sporijeg prometa podataka na internetu.

`chooseCRANmirror()`

Izabrana opcija ostat će aktivna do trenutka izlaska iz sustava R.

---

#### Napomena:

Ako unutar RStudio-a želite vidjeti sintaksu određene naredbe dovoljno je njezin naziv napisati u konzolu sustava R i pritisnuti [enter].

---

`chooseCRANmirror()`



Na konzoli će se ispisati sintaksa. Ako želite provesti naredbu sa zadanim vrijednostima tada treba upisati naredbu te potom znakove (). Ako za traženu funkciju ne postoje zadane (engl. *default*) vrijednosti na konzoli će se pojaviti poruka koja nas o tome obavještava. Ukoliko želite vidjeti sam kôd neke funkcije tada naziv funkcije upisite u R konzolu bez otvaranja zagrade i unosa argumenata. Na ovaj način, ranije pripremljene funkcije unutar sustava korisnik može ili koristiti ili na željeni način dodatno prilagoditi svojim potrebama bez kretanja od nule.

**1.2.1.2 Instalacija paketa sa spremišta (engl. repository)** Instalacija paketa s nekod od spremišta/repositorija paketa radi sef unkcijom *install.packages()*, a kao argument unijeti pod navodnicima ime paketa. Upoznajte se s funkcijom te instalirajte paket *ggplot2*.

```
?install.packages
```

```
install.packages("ggplot2")
```

Sama instalacija paketa ne čini funkcije paketa dostupnima. Svaki put prilikom početka rada u sustavu potrebno je učitati u radni prostor funkcije željenog paketa funkcijom *library()* ili *require()* pri čemu funkcija *library()* učitava funkcionalnost instaliranog paketa, dok funkcija *require()* prethodno provjeri da li su funkcionalnosti paketa već unesene u radnu konzolu te tek prema potrebi učita njegova funkcionalnost.

```
library("ggplot2")
```

Ukoliko želimo instalirali sve pakete koje su dostupni u željenom spektru aplikacija (engl. *Task View*) najprije je potrebno instalirati paket ctv funkcijom *install.packages()*. Ipak, kao i u svakom slučaju kada želimo koristiti neku funkciju unutar sustava, nema potrebe za učenjem sintakse već se najprije treba upoznati sa sintaksom željene funkcije, otvaranjem njezine informacijske kartice pa tek onda njezinom upotrebom.

```
install.packages("ctv")
```

```
library("ctv")
```

Tek tada su funkcionalnosti paketa ctv u sustavu, u mogućnosti smo sve pakete iz nekog željenog spektra zadataka odjednom unijeti u svoj sustav.

Ovo je primjer za spektar zadataka vezanih za analize multivariatnih podataka

```
install.views("Multivariate")
```

ili

```
update.views("Multivariate")
```

**1.2.1.3 Instalacija paketa putem lokalne datoteke** Paketi se mogu instalirati iako još nisu provjereni i postavljeni na nekim od spremišta, iz lokalne datoteke (engl. *installing from source*), tj. istom naredbom *install.packages()*, ali dajući argumente *repos=NULL*, *type="source"*.

```
install.packages("ime_paketa", repos=NULL, type="source")
```

gdje je datoteka formata tar.gz ili

```
install.packages("rgdal_0.9-1.zip", repos = NULL, type ="source")
```

gdje je datoteka formata zip.

Ili sljedećim kôdom na primjeru paketa gdal

```
download.file("http://cran.r-project.org/src/contrib/Archive/rgdal/",
              "rgdal_0.9-1.tar.gz")
list.files(getwd(), pattern="gz")
system("R CMD INSTALL rgdal_0.9-1.tar.gz")
```

**1.2.1.4 Instalacija paketa s osobnih Git/GitHub spremišta** U posljednjih nekoliko godina, postoji trend pohrane pripremljenih funkcija i cjelovitih paketa na osobnim GitHub repozitorijima. Za razliku od paketa pohranjenih na repozitorijima poput CRAN-a, gdje je referentni priručnik obavezna dokumentacija paketa koja nam kasnije omogućava traženje informacije o



pojedinoj funkciji (otvaranje informacijske kartice funkcije), na GitHub repozitorijima pojedinih programera/statističara ovakvu konzistentnu dokumentaciju možda nećemo pronaći. Ipak, najčešće se neki oblik dokumentacije paketa nalazi u datotekama indikativnih naziva, primjerice *readme.txt* ili slično. Ovakav pohrana paketa događa se najčešće unutar područja veoma intenzivnog razvoja i želje autora da zajednici ponude svoja programska rješenja i prije no što uspiju zadovoljiti sve zahtjeve spremišta. Velika većina paketa koje je zajednica prihvatiла i počela koristiti u konačnici dospiju i na spremište poput CRAN-a ali GitHub ostaje mjesto pohrane, posebno verzija u izradi ii fazi testiranja. Ukoliko želite instalirati neki od paketa pohranjenih na GitHub repozitorijima programera, najprije je unutar sustava R potrebno instalirati paket *devtools*.

Primjer instalacije paketa *sf* za rad s prostornim bazama podataka s Git spremišta naziva *r-spatial*:

```
library(devtools)  
install_github("r-spatial/sf")
```

##Traženje pomoći

##Informacije o paketima i funkcijama

R je sustav u kojem je traženje pomoći veoma važno radi količine dostupnih funkcionalnosti. Kao što je već spomenuto, funkcionalnosti nekih paketa uneseni su u sustav samom instalacijom sustava R. Takav je paket *utils* koji u sebi nosi ugrađene funkcije za traženje pomoći kao što funkcije s kojima ćemo s upoznati.

Ukoliko radite u RStudiju, najlakše je krenuti putem padajućeg izbornika *Help* te potom izbora *R help*. S ove stranice vode poveznice na velik broj materijala, tečajeva i proručnika o sustavu R.

### 1.2.2 Pretraživanje lokalno instaliranog sustava (engl. *offline*)

Osnovnu stranicu za pomoć možemo pozvati na način, generalna pomoć u radu:

```
help.start()
```

Kao i kod svake druge funkcije, možemo pogledati njezinu dokumentaciju kako bismo bili sigurni što nam omogućava.

```
?help.start
```

Kako bi dobili uvid u sve funkcije koje sadrži neki paket dovoljno je napisati ovu naredbu:

```
library(help = "base") #pomoć za paket base
```

Po unosu naredbe [Enter] otvorit će nam se informacijska kartica za traženi paket. Primjer za pomoć oko paketa *base*:

```
#help(ime_paketa)  
help(base)
```

ili najjednostavnije

```
?ime_paketa
```

što nam daje informaciju o pravilnoj sintaksi za traženje informacije o funkcionalnostima nekog paketa.

Potpuni popis funkcionalnosti nekog paketa dobit ćemo sljedećim kôdom:

```
library(help = "base")
```

Funkcija *help()* može se zamijeniti upitnikom ? (funkcija *Question* iz paketa *utils*). Funkcija *Question()* omogućava pristup dokumentaciji. U primjeru koji slijedi funkcijom *Question* iz paketa *base* tražimo informacije o samom paketu *base*.

```
?base
```

Pomoć pri uporabi funkcije npr. *read.table()*:

```
help(read.table)  
#jednako kao i sljedeće  
?read.table
```

Prikaz primjera upotrebe funkcije *as.matrix()*:



```
example(as.matrix)
```

Samostalno proučite promjer korištenja funkcije `read.table()`.

Funkcija `help.search()` omogućuje pretraživanje u instaliranom sustavu dokumentacije podudaranjem određenog niza znakova u imenu datoteke, pseudonimu, naslovu, konceptu ili ključnim riječima (ili bilo koje njihove kombinacije).

Isprobajte slična pretraživanja i za druge pojmove s kojima ste se susreli u svojem dosadašnjem radu.

```
help.search("logistic")
```

jednako kao

```
??logistic
```

Iako veoma korisna i često korištena, funkcija `help()` i operator `?` korisni su ukoliko poznajemo ime funkcije koju tražimo. Unutar sustava R postoje funkcionalnosti koje nam omogućuju da, iako ne znamo ime željene funkcije i objekta kao što je funkcija `apropos()`. Popisi svih funkcija i objekata koje u sebi imaju riječ "raster":

```
#pretražujemo sve objekte (ime i funkcije) na putu pretraživanja
apropos("raster")
```

### 1.2.3 Pretraživanje globalnog sustava (engl. online)

Pretraživanje dokumentacije koja postoji na cijelokupnom webu, pretražujući obaveznu dokumentaciju (engl. *reference manuals*) i opcionalnu dokumentaciju (engl. *vignettes*) za pakete koji se nalaze na repozitoriju CRAN. Za razliku od funkcija `apropos()` i `help.search()` funkcija zahtjeva vezu prema internetu te pretražuje globalni sustav pomoći.

```
RSiteSearch("principal component analysis")
RSiteSearch("krige dimensions do not match")
```

Pretraživanje dokumentacije koja sadrži točan izraz, primjerice "regression kriging":

```
RSiteSearch("logistic regression")
```

**1.2.3.1 Traženje dokumentacije o instaliranim paketima** Osnovna stranica za traženje dokumentacije o uporabi sustava R i njegovih paketa u specifičnim analizama je <http://www.r-project.org/other-docs.html>.

**1.2.3.1.1 Obavezna dokumentacija paketa na CRAN-u** Svaki paket na repozitoriju CRAN mora sadržavati i osnovnu dokumentaciju o funkcionalnostima ([http://cran.r-project.org/web/packages/ime\\_paketa/index.html](http://cran.r-project.org/web/packages/ime_paketa/index.html)) koje nazivamo (engl. *reference manual*) s popisom svih funkcija i njihovih argumenata. Direktno iz sustava moramo tražiti `help(package=ime_paketa)`.

**1.2.3.1.2 Opcionalna dokumentacija paketa na CRAN-u** Vinjete su opcionalni dokumenti vezani za određeni korisnički paket. U njima su opisane funkcionalnosti paketa na nekom specifičnom problemu. One prižaju priču o tome što se s paketom može napraviti i ne rade ih nužno autori funkcija već vinjete ponekad napišu iskusni korisnici funkcionalnosti paketa. Prikaz svih dostupnih vinjeta instaliranih paketa možemo vidjeti ako unesemo sljedeće:

```
vignette()
```

Ukoliko želimo vinjete unutar određenog paketa:

```
#upoznavanje sa sintaksom funkcije
?vignette
vignette(package="ime_paketa")
#otvaramo vinjetu za koju znamo točan naziv
vignette("ime_vinjete")
```

Ukoliko želimo vidjeti popis svih vinjeta za sve učitane (engl *attached*) pakete:

```
vignette(all=FALSE)
```



Ukoliko želimo vidjeti popis svih vinjeta svih paketa koje imamo na instalirane na računalu:

```
vignette(all=TRUE)
```

#### ZA SAMOSTALAN RAD:

Potražite vinjetu paketa *sp* i paketa *akima* te prokomentirajte razliku. Ukoliko već nisu, instalirajte navedene pakete.

Funkcijom *browseVignettes()* otvaramo web-stranicu s vinjetama svih paketa koji su instalirani na našem sustavu ili samo određenog paketa kojeg smo definirali u naredbi. Vinjete je moguće preuzeti u formatima PDF, R i Latex.

```
browseVignettes (package="sp")
```

Osnovna stranica na kojoj je moguće pronaći ogroman broj priručnika s većeg broja spremišta paketa nalazi se na poveznici <https://www.rdocumentation.org/> dok se snovna stranica s priručnicima o R jeziku i paketima pohranjenim na CRAN-unalazi se na poveznici <https://cran.r-project.org/manuals.html>.

#### 1.2.3.2 Komunikacijske liste u sustavu R

Unutar sustava R postoje četiri glavne komunikacijske liste (engl. *Mailing Lists*) na kojima se nalaze informacije o sustavu:

- R-announce (<https://stat.ethz.ch/mailman/listinfo/r-announce>)

Na ovoj se listi objavljaju velike novosti u sustavu kao što je izdavanje nove verzije sustava R i promjene koje donosi.

- R-packages (<https://stat.ethz.ch/mailman/listinfo/r-packages>)

Ovdje se objavljaju novosti o nadogradnji postojećih i dostupnosti novih paketa - uglavnom se odnosi na spremište CRAN.

- R-help (<https://stat.ethz.ch/mailman/listinfo/r-help>)

Ovo je glavna obavijesna lista za sustav R, a namijenjena je raspravi o problemima i rješenjima pomoću sustava R, najavama koje nisu objavljene na prethodne dvije liste, obavijestima o dokumentaciji za sustav R, usporedbi i kompatibilnosti s jezikom S-plus te širenju dobrih primjera. R-announce i R-packages proslijeduju sve obavijesti na R-help.

- R-devel (<https://stat.ethz.ch/mailman/listinfo/r-devel>)

Ova je lista namijenjena pitanjima i raspravi vezanim za razvoj kôda programskog jezika R.

#### 1.2.3.3 Grupe specijalnog interesa

Unutar sustava došlo je do stvaranja zajednica korisnika sustava R s posebnim interesima (engl. *Special Interest Group - SIG*). Tako je moguće predbilježiti se za primanje obavijesti vezanih uz određeno područje, na primjer, u našem je to slučaju grupa koja koristi i razvija alate za analize prostornih podataka R-SIG-geo na mrežnim stranicama.

Ukoliko ste početnik i ne razumijete poruke koje Vam sustav javlja kao pogrešku, najbolji i najjednostavniji način je kopiranje teksta pogreške i unošenje u internetski pretraživač (na primjer Google). Na isti način moguće je pronaći i velik broj dokumenata i znanstvenih radova s temom koja nas zanima.

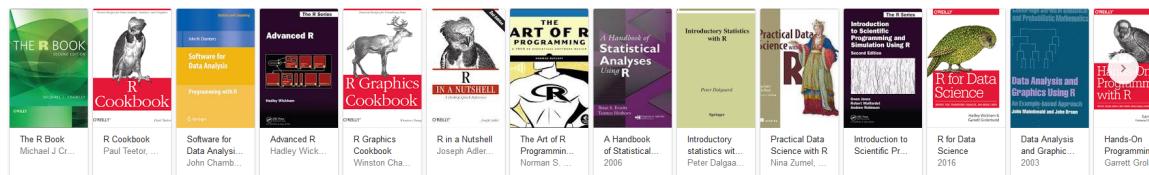
Ako na opisane načine niste uspjeli riješiti svoj problem i odlučite postaviti pitanje na neku od distribucijskih lista važno je:

- dati primjer koji će drugi korisnici moći reproducirati
- navesti koji ste rezultat očekivali, odnosno pokušali dobiti
- navesti što je prikazano umjesto očekivanog rezultata
- navesti verziju sustava R i paketa korištenih u analizi
- navesti naziv i inačicu operacijskog sustava kojeg koristite
- osmisiliti naslov poruke koji će na jasan način predstaviti problem.

Noviji komunikacijski kanali i mjesta za postavljanje pitanja su i osobne Git stranice osoba koje neki paket razvijaju te veoma popularna platforma *StackOverflow*.

Postoje i mnogi drugi kanali putem kojih R zajednica komunicira, tako postoje i grupe potpore u radu ženama koje se bave R-om (*RLadies*) ali i mnogo Google grupa. Lokalna otvorena Google grupa korisnika programskog jezika R zove se *Rtunut!*





Slika 3: Knjige o sustavu R iz različitih izvora

**1.2.3.4 Specijalizirane konferencije** Informacije o konferencijama o sustavu R mogu se pronaći na poveznici <http://www.r-project.org/conferences.html>. R-projekt aktivno podupire rad dva tipa konferencija koje redovno organizira R-zajednica:

- **useR!**  
konferencija-forum korisnika sustava R i
- **DSC (engl. Directions in Statistical Computing)**

konferencija koja okuplja programere statističkog softvera i istraživača u statističkom računalstvu koji su usredotočeni na sustav R, ali nije isključivo posvećena sustavu R. Cilj konferencije je osigurati platformu za razmjenu ideja o zbijanjima u statističkom računalstvu.

**1.2.3.5 Ostala literatura o sustavu R** Unutar sustava R postoji velika količina literature slobodno dostupne korisnicima. Osnovna grupa suradnika za razvoj sustava R (engl. *R Core Team*) zadužena je za održavanje i sadržaj mrežnih stranica s priručnicima.

Za operacijski sustav Windows literatura je dostupna putem poveznice <http://cran.r-project.org/manuals.html>. Putem ovih stranica moguće je doći do priručnika prevedenih na druge svjetske jezike (<http://cran.r-project.org/other-docs.html>).

Osim literature unutar sustava pomoći navodimo još neke od klasičnih tekstova koji olakšavaju početak rada u sustavu R:

- Chambers (2008) Software for data analysis, Springer
- Chambers (1998) Programming with data, Springer
- Venables & Ripley (2002) Modern Applied Statistics with S, Springer
- Venables & Ripley (2000) S Programming, Springer
- Pinheiro & Bates (2000) Mixed-Effect Models in S and S-PLUS, Springer
- Murrell (2005) R Graphics, Chapman & Hall / CRC Press
- Use R! Serija knjiga, Springer
- Serija knjiga izdavača Wiley
- ...
- The R Journal

*Open-access* časopis koji objavljuje kraće članke vezane uz projekt razvoja softvera u sustavu R koji bi mogli biti od interesa kako osobama koji sustav razvijaju tako i korisnicima, a može se pronaći na: <http://journal.r-project.org/current.html>.

Velik broj kniga o sustavu R moguće je pronaći na poveznici : <http://www.r-project.org/doc/bib/R-books.html> ali s naznakom da lista nije u potpunosti ažurirana.

Na hrvatskom jeziku, osim tečajeva Srca, moguće je pronaći relativno stare, dokumente:

*Uvod u korištenje R-a* (2004) Damir Kasum i Tarzan Legovic s Instituta Ruđer Bošković preveli su knjigu *An introduction to R* autorskog tima W.N. Venables, D.M. Smith i osnovna grupa suradnika za razvoj sustava R (<http://cran.r-project.org/doc/contrib/Kasum+Legovic-UvodUr.pdf>).

*Osnovne naredbe u R-u* Damir Kasum ([http://cran.r-project.org/doc/contrib/Kasum-QuickRefCard\\_ver.1.2.pdf](http://cran.r-project.org/doc/contrib/Kasum-QuickRefCard_ver.1.2.pdf)).

#### 1.2.4 Određivanje okoline za rad

Posao je okoline povezivanje skupa sa skupom vrijednosti. Naredbom `search()` bez dodatnih argumenata dobit ćemo popis redoslijeda spajanja vrijednosti u radnom prostoru. Ujedno iz njega vidimo i popis svih paketa učitanih u radnu okolinu kao i redolijed njihova učitavanja.

```
search()
```

```
## [1] ".GlobalEnv"      "package:latex2exp" "package:png"
## [4] "package:lattice"   "package:knitr"    "package:xlsx"
## [7] "package:xlsxjars"  "package:rJava"    "package:stats"
## [10] "package:graphics"  "package:grDevices" "package:utils"
## [13] "package:datasets"  "package:methods"  "Autoloads"
## [16] "package:base"
```

Naredbom `library()` bez dodatnih argumenata dobit ćemo popis svih instaliranih paketa u našem sustavu.

#### 1.2.5 Stvaranje individualiziranog sustava

Svaki korisnik u mogućnosti je prilagoditi neke funkcionalnosti na sustavu izmjenom datoteke `Rprofile.site` koja se nalazi unutar mape `etc`, u našem slučaju to je unutar ove putanje

```
paste("...\\R\\R-3.4.1\\etc")
```

#### 1.2.6 Skupovi podataka za vježbu

Unutar R sustava postoji određen broj skupova podataka na kojima možete isprobavati funkcije i pripremati analize. Velik broj skupova podata sakupljeno je i dostupno putem paketa `datasets` koji dolazi instalacijom sustava i obuhvaća raznolike strukture podataka s kojima možete vježbatи.

```
#skupovi podataka za vježbu unutar paketa "datasets"
data()
```

Ipak, paket `datasets` ne sadži podatke organizirana u specijalizirane interne R klase za specifične analitičke metode kao što su geografski referencirani podaci, prostorno vremenski podaci te mnogi drugi primjeri koje velik broj korisnika nikada i neće susresti u svojem radu. Ukoliko je potrebno za razumijevanje funkcionalnosti koje neki specifičan paket nudi, autori paketa, često u sklopu paketa dostavljaju adekvatne podatke na kojima je moguća vježba i upoznavanje s funkcionalnostima.

Prokomentirajte razliku između paketa raster i sp. Ukoliko je potrebno, napravite i instalaciju ova dva pakata te njihove funkcionalnosti učitajte u radni prostor funkcijama `library()` ili `require()`.

```
#install.packages("raster")
#install.packages('sp')
data(package = "raster")
data(package = "sp")
```

###Prijelaz na novu verziju sustava R

Najjednostavniji način prelaska na novu verziju sustava R je kopiranje i prebacivanje svih instaliranih paketa iz mape

```
...\\R\\R-ver1\\library
```

u mapu paketa verzije 2

```
...\\R\\R-ver2\\library
```

Na staroj verziji sustava R napravite sljedeće, uporabom primjera s verzijom R-3.4.1:

```
setwd("../S720")
packages <- installed.packages() [, "Package"]
save(packages, file="Rpackages")
#Na novoj verziji sustava R napravite sljedeće:
setwd("../C:/Temp/")
```



```
load("Rpackages")
for (p in setdiff(packages, installed.packages() [, "Package"]))
install.packages(p)
```

Kako je sustav R izuzetno fleksibilno okruženje, gotovo u svakoj prilici postoji mnogo načina na koje se može doći do željenog rezultata. Da bismo potkrijepili ovu tvrdnju isto ćemo napraviti putem već pripremljenog paketa *installr*; instaliranje i učitavanje najnovije verzije *installr* paketa.

```
install.packages("installr")
library(installr)
updateR()
```

Kako bismo bili sigurni da u svojem radu koristite najnoviju verziju nekog paketa moguće je napraviti ažuriranje uporabom naredbe *update.packages()*.

```
update.packages(checkBuilt=TRUE, ask=FALSE)
updateR()
```

Radi lakše izrade programa (skripti) i upravljanja kôdom sustava R, tijekom tečaja ćemo koristiti RStudio, integralnu platformu za razvoj (engl. *Integrated Development Environment - IDE*) pripremljenu upravo za sustav R. Ova platforma uključuje razne funkcionalnosti, kao što su automatsko detektiranje grešaka u kôdu (engl. *debug*) i pripremu vlastitih paketa, a integrirana je s dokumentacijom o funkcijama te omogućuje integraciju s alatima za automatsku izradu izvještaja kao što su *Sweave* (integracija s dokumentima programa Latex) i R Markdown (izrada HTML-dokumenata). Relativno noviji paket *rmarkdown* u znatnoj je mjeri olakšao reproducibilnu znanost i izvještavanje integrirajući veći broj softwera. Uz nove funkcionalnosti RStudija koje u RMarkdown dokumentima dopuštaju dijelove različitih programskih jezika doveo je pojам reproducibilnosti na posve novu razinu. Korisnik tako danas u jednom dokumentu ima mogućnost korištenja bilo kojeg programskog jezika u kojem može zna/može rješiti neki problem u jedinstvenom dokumentu (primjerice Tex, Python, R...) kombinirajući njih s pisanjem teksta rada ili izvještaja (engl. *iterate programming*).

Ukoliko u svojem radu koristite RStudio prelazak na novu verziju R-a je još jednostavniji. S CRAN repozitorija instalirajte željenu verziju R-a, otvorite RStudio i svi će paketi iz ranijih verzija biti dostupni i u novoj. RStudio će automatski prepoznati novu verziju i raditi s njom. Ukoliko to ne želite, pod opcijama programa namjestite (RStudio - Tools - Global Options - General) namjestite željenu verziju R-a za rad. RStudio, također, omogućava instaliranje i/ili ažuriranje pojedinih paketa interaktivno.

### 1.2.7 Početak rada

Program R pokreće se na neki od uobičajenih načina za operacijski sustav Windows:

- dvostrukim klikom lijevom tipkom miša na ikonu na radnoj površini (engl. *Desktop*)
- klikom lijeve tipke miša na ikonu na traci za brzo pokretanje programa (engl. *Quick Launch*)
- upisom R-3.4.1 u prompt prozor sustava Windows.

**1.2.7.1 Zadavanje naredbi** Većina akcija koje želite napraviti u sustavu radi se putem unosa naredbi koje obično izgledaju ovako: znak > ukazuje na spremnost sustava na prihvatanje nove naredbe (engl. *prompt*) i sustav R ga automatski stavlja na početak retka. Korisnik nikada ne upisuje znak za spremnost kao dio kôda.

Nakon pisanja naredbe potrebno je pritisnuti tipku [ENTER].

Postoji nekoliko načina da sustavu prenesete naredbe:

- izravan upis naredbe u prompt prozor na opisani način
- kopiranje i ljepljenje naredbe iz nekog drugog dokumenta (pojedini red, dio ili cijeli kôd) pritom treba paziti da se ne kopira znak spremnosti sustava (engl. *prompt*) > ili znak prekida naredbe +
- kopiranje i ljepljenje naredbe iz tekstualne datoteke s kôdom sustava R koji nosi nastavak .R; ovaj oblik datoteka može se otvoriti u tekstu editoru ili u editoru kôda kakav je Tinn-R ili RStudio
- pripremiti cjelokupnu skriptu u editoru kôda i poslati je na izvršenje na R konzolu.



Sustav R će prihvati naredbu koja ima ispravnu sintaksu. Ako to nije slučaj pojavit će se oznaka kojom sustav traži završetak sintakse u obliku znaka plus +. Ukoliko ne želite završavati naredbu, pritisnite tipku [esc].

Kako je za početnike rad s naredbama ponekad zastrašujući, nekoliko grupa razvija i grafička sučelja za rad o kojima u ovome tečaju neće biti govora. Najčešće rabljeno grafičko sučelje za rad u sustavu R je *R commander* dok primat u radu s web grafičkim sučeljima je Deducer.

Radi preglednosti kôda, od sada ćemo pripremljene programe otvarati u programu RStudiJU. Pokrenite RStudio. Ukratko ćemo se upoznati s izgledom programa.

**1.2.7.2 Postavljanje radnog direktorija** Za korisnike operacijskog sustava Windows važno je napomenuti da sustav R ne može koristiti notaciju putanje do određene datoteke na način kako se to radi unutar sustava Windows. Na primjer, sustav R neće moći pročitati

```
c:\mydocuments\file.txt
```

iz razloga što znak *backslash* prepoznaje kao tzv. engl. *escape character* koji sa sobom nosi alternativno tumačenje sljedećeg znaka u nizu znakova. Iz tog razloga putanje do datoteka moraju se promjeniti na jedan od ova dva načina:

1) ili dvije dvostrukе linija unatrag (engl. *backslash*)

```
"c:\mydocuments\file.txt"
```

2) ili jednostruka linija unaprijed (engl. *forwardslash*)

```
"c:/mydocuments/file.txt"
```

Prije početka rada potrebno je definirati radni direktorij. Informacija o trenutačnom radnom direktoriju dobiva se na ovaj način:

```
getwd()
```

Radni direktorij određuje se na sljedeći način:

```
setwd("putanja/moj_direktorij")
```

**1.2.7.3 Neke osnovne funkcije za rad u sustavu:** U nastavku slijedi nekoliko funkcija koje je dobro znati prilikom samog početka rada u R sustavu.

Za ispis svih objekata u radnom prostoru (engl. *workspace*) upišite:

```
ls()
```

Otvaranje informacijske kartice željene funkcije, primjer za funkciju *lm()*

```
?lm
```

Snimanje i učitavanje objekta i/ ili cjelokupnog radnog prostora:

```
save(ime_objekta, file="ime_datoteke.RData")
```

Snimanje liste objekata na tvrdi disk:

```
save(c(ime_objekta1, ime_objekta2), file="ime_datoteke.RData")
```

Snimanje cjelokupnog radnog prostora (svi objekata aktivne sesije) na tvrdi disk

```
save.image("ime_datoteke.RData")
```

Učitavanje cjelokupnog radnog prostora:

```
load("ime_datoteke.RData")
```

Snimanje povijesti (datoteke history) na tvrdi disk:

```
savehistory(file="ime.Rhistory")
```

Učitavanje povijesti (datoteke history) u radni prostor:



```
loadhistory(file="ime.Rhistory")
```

#### Napomena:

U konzolu sustava R upisujemo naredbe. Simbol <- je operator pridruživanja. Veoma slično, ipak ne identično, ponaša se i drugi simbol pridruživanja vrijednosti =.

Simbol # predstavlja početak komentara i sve napisano desno, uključujući i sam znak, biti će zanemareno prilikom izvođenja naredbi.

Program otvaramo R dvostrukim klikom lijevom tipkom miša na ikonu na radnoj površini. Kopirati je potrebno sve između znakova prompt > i komentara #.

Ne smeta ako je unesen i komentar - program R će ga zanemariti.

#### PITANJA ZA PONAVLJANJE:

- 1) Kojom funkcijom određujemo radni direktorij u R-u?
- 2) Što su to vinjete? Kojom funkcijom ih možete pregledati i otvoriti?
- 3) Što ćete dobiti ukoliko u konzoli izvršite naredbu `help()` bez zadanih argumenata? Prokomentirajte.
- 4) Što ćete dobiti ukoliko u konzoli izvršite naredbu `help(sd)` ili `help(base)` s dodanim argumentom? Prokomentirajte.
- 5) Pomoću funkcije `chooseCRANmirror()` odredite kao željenu stranicu s koje ćete uzimati pakete onu iz Italije.
- 6) Postoji li stranica CRAN-a u Hrvatskoj?
- 7) Odredite u sustavu radni direktorij `~/S720/user`.
- 8) Prokomentirajte poruku sustava kada ste zatražili skupove podataka iz paketa base.
- 9) Kojim znakom možete zamjeniti funkciju `help()`?
- 10) Kojim znakom možete zamjeniti funkciju `help.search()`?
- 11) Kojom biste funkcijom snimili cijelokupan radni prostor po završetku rada?
- 12) Razlikuje li R velika i mala slova?
- 13) Koja je razlika između lokalnog i globalnog pretraživanja? Navedite primjere funkcija.
- 14) Koja je razlika između funkcija `library` i `require` (ako postoji)?

#### ZADACI ZA SAMOSTALNI RAD:

#ZADATAK 1: Kojom biste funkcijom zatražili informaciju  
#o trenutnom radnom direktoriju?

#ZADATAK 2: Definirajte željeni radni direktorij kao direktorij  
#u kojem se nalaze skripte za ovaj tečaj.

#ZADATAK 3: Kreirajte objekt koji ćete nazvati razlika.  
#U taj objekt spremite razliku brojeva brojeva 19217 i 19113.  
#Što bi se dogodilo da ovu razliku nismo spremili u objekt?

#ZADATAK 4: Kreirajte objekt pod nazivom div.  
#Objekt treba biti broj 3384 podijeljeno s 51:

#ZADATAK 5: Instalirajte paket `ggplot2` i učitajte ga u radni prostor.  
#Ima li ovaj paket vinjete, ako da, koliko?



## 2 Osnovne vrste podataka u sustavu R

Unutar sustava R postoji šest osnovnih atomskeih (engl. *atomic*) vrsta podataka:

- znak / slovo / tekst (engl. *character, string*)
- realan broj / numerik (engl. *numeric*)
- cijeli broj (engl. *integer*)
- kompleksni broj (engl. *complex*)
- logički operator (engl. *logical*) - T (TRUE) / F (FALSE)
- sirovi (engl. *raw*).

Unutar sustava, definirani su i specijalni brojevi beskonačnost Inf (engl. *infinity*) primjerice  $1/0$  dok je  $1/\text{Inf} = 0$ .

```
#matematičke definicije
1/0
1/Inf
```

Sustav R je interaktivni kalkulator s određenim brojem ugrađenih funkcija i vrijednosti konstanti.

Generalno definirane, a nedostajuće vrijednosti u sustavu R su označene oznakom NA (engl. *not available*) dok se nedefinirane vrijednosti, kao što je slučaj  $0/0$  označavaju s NaN te predstavljaju nedefiniranu vrijednost (engl. *not a number*).

Primjer: Radimo vektor veličine 2 i ispisujemo ga na konzoli:

```
a<-c(1,2)
a
```

Selekcijom trežimo element koji ne postoji:

```
a[3]
```

Ukoliko čelimo ispitati pojedini element vektora, radimo specifične upite na svaki element vektora, primjerice spitujemo da li su pojedini elementi vektora NA vrijednosti:

```
#
#i
is.na(c(0/0,NA))

#ispitujemo da li su pojedini elementi vektora NAN vrijednosti
is.nan(c(0/0,NA))
```

Također su u ovom kontekstu ispitivanja svojstva brojeva korisne naredbe o provjeri pojedinog elementa vektora o njegovoj konačnosti, funkcija *is.finite()* i *is.infinite()*.

##Strukture podataka

R je objektno orientirani jezik. Neki od poznatijih objektno orientiranih jezika su i C++, Python, Smalltalk, PHP, Java, Perl, Ruby. Objektno orientirani jezici se temelje na konceptu da objekt posjeduje atribute koji ga opisuju te njemu pridružene procedure koje se još nazivaju metodama. Tako i u osnovi sustava R postoje klase i metode (engl. *methods*). Klasa (engl. *class*) je definicija objekta. Sustav klasa govori nam što pojedini elementi u sustavu R u biti jesu. Tipično je za klase da unutar sebe imaju definirane isječke (engl. *slot*) koji se koriste za čuvanje specifičnih informacija za tu klasu objekta. Metode su funkcije koje djeluju jedino na određenoj klasi objekata. Generičke funkcije u sebi nose neku generičku (temeljnu) funkcionalnost - koncept, tako funkcija *print()* ispisuje objekte, funkcija *plot()* ih crta, *predict()* radi projekciju na novom skupu podataka, itd. No svaka od tih funkcija ponaša se nešto drugačije u odnosu na klasu objekta na koji je primjenjena.

U osnovi, generička funkcija ništa ne radi/računa već prepoznaže klasu objekta i tada pronalazi odgovarajuću metodu koja odgovara tom tipu objekta i zove metodu. Metoda je ono što implementira generičke funkcije na objektu određene klase. Ako generička funkcija ne pronađe odgovarajuću metodu za traženu klasu objekta, primjenjuje unaprijed zadalu metodu (engl. *default*). Ako takva nije definirana, sustav javlja pogrešku. U ovome tečaju nećemo ići dublje u sustav metoda i klasa u sustavu R.

U ovom će dijelu biti prikazan pregled najvažnijih struktura podataka sustava R *base*. Kako je već i ranije spomenuto, paket *base* čini osnovu jezika R. Tablica u nastavku daje pregled struktura podataka u sustavu R na temelju njihove dimenzionalnosti i homogenosti podataka (svi sadržaji moraju biti iste vrste) ili heterogeni (sadržaj može biti različitih vrsta) kako je prikazano u tablici 1.



Tablica 1: Osnovne strukture podataka u sustavu R

Broj.dimenzija	Homogeni	Heterogeni
1	Vektori (engl. <i>Atomic Vector</i> )	liste (engl. <i>List</i> )
2	Matrice (engl. <i>Matrix</i> )	Skupovi podataka (engl. <i>Data Frame</i> )
N	Nizovi / polja (engl. <i>Array</i> )	

Primijetite jednu osobitost sustava R, a to je da nema skalar, odnosno vrstu podataka bez dimenzije. Svi skalari (alfanumerički znakovi) su u sustavu R definirani kao vektori duljine 1.

Gotovo svi drugi objekti u sustavu R su izgrađeni na ovim temeljnim strukturama. Alati unutar objektno orijentiranog jezika se također grade na ovim osnovama.

Ako želite pogledati strukturu nekog objekta u sustavu R, najvažnija je funkcija `str()` što je skraćenica engleske riječi *structure* koja nam na konzoli daje opis objekta na jednostavno čitljiv način.

#### ###Vektori

Osnovna vrsta podataka u sustavu R je vektor. Vektori dolaze u dva oblika: atomski (engl. *atomic*) i liste (engl. *list*). Ove dvije vrste razlikuju se po svom sadržaju. Dok podaci unutar atomskog vektora nužno moraju biti iste vrste, sadržaj liste to ne mora biti. Vektori u sustavu R imaju tri osnovna obilježja:

- 1) Što su? Koje su vrste?

Ispitivanje ovih svojstava vektora radimo specifičnim funkcijama paketa `base`.

`typeof()`

- 2) Koliko elemenata sadrži neki vektor, koja duljina vektora?

`length()`

- 3) Dodatni atributi, metapodaci o vektoru.

Jedini metapodaci koji možemo dodijeliti klasi objekta tipa vektor je atribut imenovanja svakog elementa vektora, `names()`.

`attributes()`

**2.0.0.1 Atomski vektori** Atomski vektori mogu biti logički (engl. *logical*), cijelobrojni (engl. *integer*) ili numerički (engl. *numeric*) te tekstualni (engl. *character* ili *string*). Rjeđe su kompleksni (engl. *complex*) ili sirovi binarni (engl. *raw*). O posljednje dvije vrste neće biti obrađene u ovome tečaju. Vektori se najčešće izrađuju funkcijom `c()` što je prvo slovo engleske riječi *combine*.

**Napomena:** Decimalni separator u sustavu R je decimalna točka.

#### Primjer:

Izrada tekstualnog vektora i pohrana u objekt naziva `tekstualni_vektor` te primjeri izrade numeričkog i eksplicitno cijelobrojnog vektora.

```
#izrada vektora, kombiniranje bilo kojih stringova u vektor
tekstualni_vektor <- c("tekst1", "tekst2", "tekst3", "tekst4", "tekst5")
```

```
#ispis
tekstualni_vektor #(engl. "character" vektor, dva teksta (engl. "strings")
```

```
[1] "tekst1" "tekst2" "tekst3" "tekst4" "tekst5"
```

#izrada numeričkog vektora

```
numericki_vektor <- c(1, 7.5, 6.5, 8.4)
```

```
numericki_vektor2 <- c(1,8,18,33)
```

Ukoliko želimo eksplicitno izraditi cijelobrojni vektor:



```
cjelobrojni_vektor <- c(1L, 6L, 10L)
```

Izrada sekvence cjelobrojnih vrijednosti:

```
cjelobrojni_vektor2 <- 1:20
```

```
cjelobrojni_vektor2
```

```
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

Ukoliko želimo kreiranje logičkog vektora i pohraniti ga u u objekt naziva *logički\_vektor*:

```
logički_vektor <- c(TRUE, FALSE, T, F)
```

```
logički_vektor
```

```
[1] TRUE FALSE TRUE FALSE
```

Objekt klase vektor možemo izraditi i funkcijom *vector()*.

```
logički_vektor_2 <- vector("logical", length=15)
```

```
logički_vektor_2
```

```
[1] FALSE  
[12] FALSE FALSE FALSE FALSE
```

Izrada vektora s dodatnim atributima - najčešći opcionalni atribut u sustavu R je *names()*.

```
vektor_1 <- c("Marko", 9, "sladoled", T)
```

Dodjeljujemo vektoru atribut *names* i ispisujemo objekt, vektor:

```
names(vektor_1) <- c("ime", "starost", "najdraža_hrana", "sestra_brat")
```

```
vektor_1
```

ime	starost	najdraža_hrana	sestra_brat
"Marko"	"9"	"sladoled"	"TRUE"

Određivanje tipa vektora radi se upitima, kako smo već naveli funkcijom *typeof()*, ili specifičnijim testovima kao što su: *is.character()* - funkcija ispituje da li su elementi vektora tipa karakter, *is.double()* - funkcija ispituje da li su elementi vektora realni brojevi, *is.integer()* - funkcija ispituje da li su elementi vektora cijeli brojevi, *is.logical()* - funkcija ispituje da li su elementi vektora logičkog tipa, ili općenitijima: *is.atomic()* - radi li se o atomskom tipu elemenata ili pak *is.na()* kojim ispitujemo da li je svaki pojedini element vektora NA vrijednost).

#### Napomena:

Tekst u sustavu R mora biti označen navodnicima. U suprotnom R smatra da se radi o objektu u njegovom prostoru ili imenu varijable.

**2.0.0.2 Indeksiranje i selekcija podskupa vektora** Sve primjere selekcije elemenata objakta koje ćemo pokazati na primjeru vektora, vrijede i na svim ostalim klasama objekata s kojima ćemo se upoznati.

**2.0.0.2.1 Selekcija putem indeksa (položaja elementa u objektu)** Selekcija elemenata vektora (engl. *subset*) provodi se operandom `[]`. Unutar zagrade indeksom rednog broja elementa u vektoru dobiti ćemo željenu vrijednost. Selekcija drugog elementa objekta vektor\_1:

```
vektor_1[2]
```

```
## starost  
##      "9"
```



Selekcija prvog i trećeg elementa vektor\_1:

```
vektor_1[c(1,3)]
```

```
    ime najdraža_hrana  
    "Marko"      "sladoled"
```

ili putem atributa, kod vektora moguć jedino atribut *names()*.

```
vektor_1["starost"]
```

```
starost  
"9"
```

Negativni indeksi se koriste u slučajevima kada želimo izbaciti određene elemente vektora kako je pokazano u sljedećem primjeru:

Izbacivanje drugog i četvrtog elementa vektora\_1:

```
vektor_1[-c(2,4)]
```

```
    ime najdraža_hrana  
    "Marko"      "sladoled"
```

**2.0.0.2.2 Selekcija logičkim uvjetima** Da bismo pokazali kako funkcioniра logička selekcija, dajemo jednostavan primjer na vektoru cijelih brojeva iako se do sada još nismo upoznali s logičkim operatorima u R-u.

```
#radimo vektor naziva x koji se sastoji od sekvence cijelih brojeva od 1 do 40  
x <- 1:40
```

Selektiramo svaki drugi element počevši od prvog, zadani obrazac:

```
x[c(T,F)]
```

```
[1] 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
```

Selektiramo svaki drugi element počevši od drugog, zadani obrazac:

```
x[c(F,T)]
```

```
[1] 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
```

Selektiramo sve elemente čije vrijednosti iznose 7 ili više. Prije no što napravimo zadatok pogledajmo što je rezultat upita:

```
x>=7
```

```
[1] FALSE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE  
[12] TRUE  
[23] TRUE  
[34] TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Ranije dobiveni logički vektor koristimo kao uvjet za selekt na prvočitnom elementu

```
x[x>=7]
```

```
FALSE [1] 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29  
FALSE [24] 30 31 32 33 34 35 36 37 38 39 40
```

Primje složene selekcije:

```
x[x<=30 | x==200]
```

```
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23  
[24] 24 25 26 27 28 29 30
```

Korištenjem negativnih indeksa možemo napraviti selekciju svih elemenata vektora. Tako ćemo napraviti selekciju svih elemenata vektora osim od drugog do četvrtog elementa na sljedeći način.

Izbacivanje drugog do četvrtog elementa vektora\_1



```
vektor_1[-(2:4)]
```

Primijetite da, ukoliko neprekinutu stvarate sekvencu cijelih brojeva, nije potrebno iste konbinirati korištenjem funkcije `c()`.

## 2.1 Opcionalni atributi objekata u sustavu R

Svi objekti u sustavu R mogu imati opcionalne attribute. Atributi pojedinog objekta mogu se pozvati i pregledati pojedinačno funkcijom `attr()` ili svi mogući atributi na zadanoj klasi objekta odjednom, korištenjem funkcije `attributes()`. Upoznajmo se s funkcijom `names()` otvaranjem informacijske kartice funkcije na način da u R konzolu upišemo: `?names`.

```
attributes(vektor_1)
```

```
## $names
## [1] "ime"           "starost"        "najdraža_hrana" "sestra_brat"
attr(vektor_1, "names")
## [1] "ime"           "starost"        "najdraža_hrana" "sestra_brat"
```

### 2.1.1 Opcionalni atributi na vektorima, klasa objekta vector.

Jedini atribut koji možemo dodijeliti klasi objekta vektor je atribut imenovanja, engl. `names`. Elemente vektora možemo imenovati na tri načina:

1) prilikom izrade vektora:

```
x <- c(var1 = 1, var2 = 2, var3 = 3)
x
```

```
var1 var2 var3
1     2     3
```

2) izmenjom postojećeg vektora:

```
#u prvom koraku samo napravimo željeni vektor, u ovome slučaju vektor x s tri elementa
x <- 1:3
x
```

```
[1] 1 2 3
#u drugoj iteraciji, dodjeljujemo ime svakom elementu vektora x
names(x) <- c("var1", "var2", "var3")
x
```

```
var1 var2 var3
1     2     3
```

3) izradom izmijenjenog vektora:

```
#upoznajmo se s funkcijom setNames tako što ćemo otvoriti informacijsku karticu funkcije
x <- setNames(1:3, c("var1", "var2", "var3"))
x
```

#### Napomena:

Sustav R prema konvenciji logičke operatore u suštini smatra numericima. Tako je T (TRUE) definirano i kao 1 dok F (FALSE) ima vrijednost 0.

#### PITANJA ZA PONAVLJANJE:

- 1) Nabrojite osnovne, atomske, strukture podataka u R-u.
- 2) Koja su tri osnovna obilježja vektora?
- 3) Kojom funkcijom biste ispitali od kakvog atomskog tipa podataka je kreiran zadani vektor?
- 4) Kojom funkcijom biste ispitali da li je atomska vektor numerički?
- 5) Kojom funkcijom biste ispitali da li je atomska vektor numerički, cjelobrojni?



- 6) Kojim funkcijama možemo kreirati vektor?
- 7) Kojom funkcijom biste ispitali atribute određenog vektora?
- 8) Na koji biste način izbacili posljednji element nekog vektora?
- 9) Što ispitujemo sljedećom linijom: `typeof(iris)[2]`?
- 10) Prokomentirajte dolje navedeni primjer i objasnite radi čega će nastati nevedene klase vektora:

```
v <- c(7.5, "b") # rezultat je tekstualni vektor (engl. "character")
v<- c(TRUE, 2) # rezultat je numerički vektor (typeof=double)
v<- c("a",TRUE) # rezultat je tekstualni vektor
```

### 2.1.2 Liste

Liste (engl. *list*), za razliku od atomskih vektora, mogu sadržavati i druge vrste vektora, uključujući i same liste. Radi ovog svojstva još ih nazivaju i rekurzivnim vektorima. Mogu se izraditi pomoću funkcije `list()`, a ne više uz pomoć funkcije `c()`.

```
#upoznajmo se s naredbom, otvorimo informacijsku karticu funkcije *list()**
?list
```

Prisjetimo se da ukoliko u konzolu samo unesemo ime objekta, R će objekt ispisati:

```
#izrada objekta klase list i stavljanje liste u objekt naziva lista_1
lista_1 <- list(10, "z", FALSE, matrix(1:20, ncol=4, nrow=5))
```

```
#ispis na konzoli
lista_1
```

```
[[1]]
[1] 10

[[2]]
[1] "z"

[[3]]
[1] FALSE

[[4]]
 [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
```

Liste su veoma važne pri izradi kompleksnih objekata u sustavu R. Prema konvenciji identificiranje elemenata liste radi se operatorom `[[ ]]`.

U sljedećem primjeru radimo selekt drugog elementa u listi:

```
lista_1[[2]]
```

```
[1] "z"
```

Važno je primijetiti pa stoga ponavljamo da se pojedini elementi liste označavaju dvostrukim uglatim zagradama `[[ ]]` dok se unutar svakog pojedinog elementa njihovi elementi označavaju jednostrukim uglatim zagradama `[ ]`.

Primjer selekcije drugog retka, trećeg stupca na četvrtom elementu liste radila bi se na ovaj način.

```
lista_1[[4]][2,3]
```

```
[1] 12
```

Dodjeljivanje atributa objektu klase `list` radi se, kako smo se već upoznali na vektorima, korištenjem funkcije `names()`.



```
#izrađujemo objekt klase *list*
lista_2 <- list("Pero", 1, matrix(runif(10)), 5.3)

#svakom elementu napravljenog objekta dodijelimo
names(lista_2) <- c("ime", "broj", "moja_matrica", "starost")
```

Do jednakog objekta mogli smo doći i na ovaj način, stvaranjem liste i dodjeljivanjem atributa u jednoj naredbi.

```
#izrada liste i dodavanje liste objektu naziva lista_2
lista_2 <- list(ime="Pero", broj=1, moja_matrica=matrix(runif(10)), starost=5.3)

#ispis objekta
lista_2
```

```
$ime
[1] "Pero"

$broj
[1] 1

$moja_matrica
[,1]
[1,] 0.683504646
[2,] 0.352969895
[3,] 0.199937873
[4,] 0.859475204
[5,] 0.632257934
[6,] 0.275409024
[7,] 0.289704392
[8,] 0.541028412
[9,] 0.009277279
[10,] 0.547607794
```

```
$starost
[1] 5.3
```

Generalno vrijedi da se pojedini dijelovi objekta mogu selektirati putem njihovog položaja u objektu (selekcija indeksom), putem atributa ali i kombinacijom ova dva načina.

Izrada liste koja u sebi nosi dvije liste.

```
lista_3<-list(grad="Zagreb",ulica="Ilica",k_br=17)

#koristenjem funkcije c()
spoj_lista_c <- c(lista_2,lista_3)

#koristenjem funkcije list()
spoj_lista_list <- list(lista_2,lista_3)
```

Usporedite rezultate dvije posljednje funkcije te prokomentirajte razliku u strukturama dobivenih objekata.

```
str(spoj_lista_c)

str(spoj_lista_list)
```

Traženje elementa određenog naziva u listi, selekcija elementa liste putem atributa *names()*:

```
lista_2[["grad"]]
```

Vraćanje liste natrag u atomski vektor radi se funkcijom *unlist()*.



```
unlist(spoj_lista_c)
```

Usporedite sa sljedećim kôdom:

```
unlist(spoj_lista_list)
vektor_povrat <- unlist(lista_2)
```

**2.1.2.1 Atributi objekata klase *list*** Kao što smo već naučili prilikom upoznavanja s vektorima, Svi objekti u sustavu R mogu imati opcionalne atributte. Na vektorima je to mogao jedino biti atribut pojedinog elementa, *names()*. Kako klasa objekta *list* ima mnogo karakteristika vektora, to je i u ovome slučaju ovaj atribut koji je jedino moguće dati.

```
#svi atributi na objektu
attributes(lista_2)

#ispitujemo baš atribut names()
attr(lista_2, "names")
```

Analogijom s vektorima, duljinu liste, broj elemenata liste, ispitujemo funkcijom *length()*.

```
length(lista_2)
```

#### ZADACI ZA SAMOSTALNI RAD:

- 1) Izradite proizvoljnu listu od 4 elementa na način da su dva elementa tekstualna a dva prema izboru.
- 2) Izradite listu naziva I koja će kao svaki element sadržavati jedan cijeli broj iz sekvence od 1:5.
- 3) Dodijelite svakom elementu lista I proizvoljno ime.
- 4) Selektirajte četvrti element liste I.

#### PITANJA ZA PONAVLJANJE:

- 1) Definirajte liste. Kojom biste funkcijom krenuli u izradu jedne liste?
- 2) Kojim operatorom radimo selekciju elemenata liste?
- 3) Kojim operatorom radimo selekciju unutar određenog elementa liste?
- 4) Da li je točna tvrdnja: U izrazu lista[[5]][2,2] radimo selekciju na trećem elementu liste?
- 5) Slažete li se s tvrdnjom: U prethodnom primjeru selekciju na objektu klase *list* napravili smo putem atributa objekta.

```
##Matrice (engl. matrix) i polja (engl. array)
```

Polja su relativno rijetko korištene strukture podataka. Mogu biti višedimenzionalne, a specijalni slučaj dvodimenzionalnog polja je matrica (engl. *matrix*) i one čine osnovu za velik dio statističkih analiza. Posjeduju atribut dimenzije, kao nadogradnja na duljinu *length()* vektora, koji se traži funkcijom *dim()*. Funkcija *dim()* ponaša se slično već ranije korištenim funkcijama koje u ovojnosti o sintaksi ili daju traženu informaciju (u ovome slučaju dimenziju objekta) ili postavljaju zadane postavke (dimenzije).

Kao i inače, prije korištenja bilo koje funkcije sustavu R, dobro je upoznati se s načinom njezina rada i sintaksu, čitajući njezinu informacijsku karticu.

```
#otvaranje informacijske kartice za funkciju
?matrix
```

Matrica se može izraditi uz argumente broja redaka i stupaca, a pritom treba obratiti pažnju na način "popunjavanja" matrice vektorom - brojevima od 1 do 10. Davanje dimenzija matrici može se napraviti:

- 1) prilikom izrade matrice:

```
#?matrix
#izrada matrice
matrica_1 <- matrix(1:20, ncol = 4, nrow = 25)

#ispis na konzolu
matrica_1
```

[,1]	[,2]	[,3]	[,4]	
[1,]	1	6	11	16



```
[2,]   2   7  12  17
[3,]   3   8  13  18
[4,]   4   9  14  19
[5,]   5  10  15  20
[6,]   6  11  16   1
[7,]   7  12  17   2
[8,]   8  13  18   3
[9,]   9  14  19   4
[10,]  10 15  20   5
[11,]  11 16   1   6
[12,]  12 17   2   7
[13,]  13 18   3   8
[14,]  14 19   4   9
[15,]  15 20   5  10
[16,]  16   1   6  11
[17,]  17   2   7  12
[18,]  18   3   8  13
[19,]  19   4   9  14
[20,]  20   5  10  15
[21,]   1   6  11  16
[22,]   2   7  12  17
[23,]   3   8  13  18
[24,]   4   9  14  19
[25,]   5  10  15  20
```

Jednakom naredbom možemo ispitati i/ili zadati dimenzije polja.

Jednako kao i kod matrice, dimenziju nekom polju možemo zadati u 1) samoj naredbi stvaranja polja jednim vektorom mogu se zadati sve dimenzije polja kao u primjeru:

```
#?array
#izrada polja
polje_a <- array(1:12, c(2, 3, 2))
```

```
#ispis na konzolu
polje_a
```

```
, , 1
 [,1] [,2] [,3]
[1,]   1   3   5
[2,]   2   4   6
```

```
, , 2
 [,1] [,2] [,3]
[1,]   7   9  11
[2,]   8  10  12
```

ili 2) modifikacijom već postojeće matrice/polja:

```
#njprije napravimo proizvoljni vektor
c <- 1:10
```

```
c
```

```
[1]  1  2  3  4  5  6  7  8  9 10
```

```
#ispitajmo klasu objekta koji je nastao
class(c)
```

```
[1] "integer"
```



```
#već napravljenom vektoru zadamo dvije dimenzije
dim(c) <- c(5, 2)
```

```
#ponovno ispitamo klasu nastalog objekta
class(c)
```

```
[1] "matrix"
```

Polja, kao mnogodimenzionalne strukture, teško je predviđati pa ponekad i pravilno indeksirati element koji se će izdvajati. Ako ponekad niste sigurni koji je indeks elementa kojeg tražite može Vam biti korisna funkcija `slice.index()`.

Primijetite kojim se redom matrica popunjava zadanim vrijednostima.

```
#popunjavanje matrice vektorom zadanih vrijednosti
matrica_2 <- matrix(1:20, ncol = 4, nrow = 26)
```

```
Warning in matrix(1:20, ncol = 4, nrow = 26): data length [20] is not a
sub-multiple or multiple of the number of rows [26]
```

```
#popunjavanje matrice predodređenom vrijednosti NA, zadajemo samo dimenzije matrice
matrica_3 <- matrix(ncol = 4, nrow = 26)
```

Pogledajte što kao rezultat daju funkcije:

```
length(matrica_1)
```

```
[1] 100
```

```
dim(matrica_1)
```

```
[1] 25 4
```

Osnovno svojstvo duljine vektora `length()` i elemenata `names()` sada dobivaju višedimenzionalnu nadogradnju kod matrica i polja na način da se funkcija `length()` nadograđuje funkcijama za specifične upite na broj redaka `nrow()` i broj kolona `ncol()` za matrice i polja, te funkcijom upita svih dimenzija `dim()` za polja. Istovjetno, funkcija `names()` se nadograđuje funkcijama `rownames()` i `colnames()` za matrice i polja te `dimnames()` za polja.

Postavljamo dimenzija objekta `matrica_1` u jednom koraku korištenjem funkcije `dim()`. Prije samog pisanja naredbe, upoznajte se sa sintaksom funkcije `dimnames`.

```
dimnames (matrica_1) <- list(paste("red", 1:nrow(matrica_1), sep = "_"),
                                paste("var", 1:ncol(matrica_1), sep = "_"))
```

```
#pogledajmo sada izgled objekta
matrica_1
```

	var_1	var_2	var_3	var_4
red_1	1	6	11	16
red_2	2	7	12	17
red_3	3	8	13	18
red_4	4	9	14	19
red_5	5	10	15	20
red_6	6	11	16	1
red_7	7	12	17	2
red_8	8	13	18	3
red_9	9	14	19	4
red_10	10	15	20	5
red_11	11	16	1	6
red_12	12	17	2	7
red_13	13	18	3	8
red_14	14	19	4	9
red_15	15	20	5	10
red_16	16	1	6	11
red_17	17	2	7	12



```
red_18    18     3     8     13
red_19    19     4     9     14
red_20    20     5    10     15
red_21     1     6    11     16
red_22     2     7    12     17
red_23     3     8    13     18
red_24     4     9    14     19
red_25     5    10    15     20
```

jednako kao i u dva koraka gdje svake dimenzije postavljamo zasebno

```
#pobrižimo postojeća redaka i kolona i postavimo ih ponovno
#u dva koraka - svaku dimenziju zasebno

#prisjetimo se brisanja
dimnames (matrica_1) <- NULL

#postavljamo redaka
rownames(matrica_1) <- paste("red", 1:nrow(matrica_1),sep="_")
#postavljamo kolona (stupaca)
colnames (matrica_1)<- paste("var", 1:ncol(matrica_1), sep="_")
```

Funkcija *c()* sada se proširuje na *cbind()* za spajanje po stupcima i *rbind()* spajanje po redcima za matrice. Matrice se transponiraju funkcijom *t()*, a polja funkcijom *abind()* koja nije dio paketa *base* već paketa *abind*.

### 2.1.3 Indeksiranje i selekcija podskupa matrice

Selekcija elemenata matrice (engl. *subset*) radi se operatorom `[]`. Unutar zagrada, indeksom retka i stupca željenog elementa u matrici dobiva se njegova vrijednost.

Važno je za zapamtiti da u sustavu R, prva dimenzija je dimenzija redaka dok se broj stupaca ili kolona nalazi na drugom mjestu.

Selekcija elementa u drugom retku i trećem stupcu matrice:

```
matrica_1[2,3]
```

```
[1] 12
```

ili putem atributa, drugi element u varijabli/objektu *var\_3*:

```
matrica_1[2, "var_3"]
```

```
## [1] 12
```

ili u cijelosti putem atributa, redak u varijabli *var\_3*

```
matrica_1["red_2", "var_3"]
```

```
[1] 12
```

Selekcija na vektorima, matricama, poljima i skupovima podataka može biti u uvjetna, tj. iz objekta će biti prepoznati samo oni elementi koji odgovaraju zadanom uvjetu:

```
#nova funkcija za generiranje sekvenci - seq
```

```
vektor_d <- seq(1,100, by=5)
```

```
vektor_d
```

```
[1]  1  6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96
```

```
vektor_d[c(T,F)]
```

```
[1]  1 11 21 31 41 51 61 71 81 91
```



```
vektor_d[c(T,F,T)]
[1] 1 11 16 26 31 41 46 56 61 71 76 86 91
#kao rezultat daje logički vektor označavajući sve elemente koji zadovoljavaju uvjet
selekt <- vektor_d>10
vektor_d[selekt]
[1] 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96
```

Ili korištenjem funkcije `which()` na način:

```
selekt2<-which(vektor_d>10, arr.in=TRUE)
vektor_d[selekt2]
## [1] 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96
```

Na sličan način moguće je i zamijeniti elemente nekom drugom vrijednosti.

Vektori nisu jedine 1D strukture. Moguće je imati matrice s jednim stupcem ili jednim retkom ili jednodimenzionalna polja. U ovome trenutku nam to nije važno, ali ponekad bismo se mogli iznenaditi rezultatom neke funkcije radi specifične strukture objekta.

Ako se atribut `dimension()` primjeni na matrici ili polju dobivaju se liste matrica (engl. *list-matrices*), odnosno liste polja (engl. *list-arrays*). Mnogi korisnici sustava R vrlo se rijetko susreću s ovom strukturom podataka.

Za sve navedene vrste objekata unutar sustava R postoje testovi kojima je moguće postaviti upit o specifičnom svojstvu nekog objekta. Uz najkorisniju funkciju `str()` (engl. *structure*) navodimo specifične testove koji kao odgovor daju T/TRUE ili F/FALSE.

```
is.matrix(vektor_1) #da li je objekt vektor_1 matrica?
is.integer(matrica_1) #da li su elementi objekta matrica_1 cijeli brojevi?
```

Na sličan način je moguće određeni objekt prebaciti iz jedne klase u drugu, ukoliko je takav prijelaz moguć i definiran:

```
matrica_iz_vektora <- as.matrix(vektor_1)
class(matrica_iz_vektora) #provjerimo klasu dobivenog objekta
```

#### Napomena:

Ponovimo da se selekcija elementa objekata (vektora, matrica, skupa podataka itd.) može se napraviti i logičkim vektorima, pri čemu vrijedi da je T/TRUE vrijednost koja će biti selektirana, a F/FALSE stoji za vrijednost koja neće biti uzeta u selekciju.

## 2.2 Skupovi podataka

Skupovi podataka (engl. *data frames*) su način čuvanja podataka u sustavu R. Preciznije rečeno, skup podataka je lista vektora jednakih duljina. Prema ovoj definiciji oni imaju dvije dimenzije i dijele svojstva i matrica i lista. To znači da skup podataka ima `names()`, `colnames()` i `rownames()` s naznakom da u ovom slučaju funkcije `names()` i `colnames()` su iste stvari.

Duljina skupa podataka je u biti duljina liste spojenih vektora kolona - varijabli pa daje isti odgovor kao i specifičnije napisan upit putem naredbe `ncol()` (engl. *number of columns*). Naredba `nrow()` (engl. *number of rows*) daje nam odgovor o broju redaka u skupu podataka.

Skup podataka stvaramo naredbom `data.frame()`. Kako je skup podataka prema svojoj definiciji lista vektora, tako je moguće da jedan skup podataka (`data.frame`) kao jednu kolonu ima lista. Navedeno početnicima može biti teško razumljivo ali na primjerima će sve postati znatno jasnije.

Objekte za koje je to definirano možemo prebaciti u neku drugu klasu naredbom `as()` paketa `methods`. U ovome slučaju, prebacivanje u skup podataka radi se naredbom `as.data.frame()` (što je gotovo identično s pisanjem `as(objekt, "data.frame")`) i to redom:

- vektor će rezultirati skupom podataka s jednim stupcom (broj redaka jednak duljini vektora)
- lista - za svaki element liste izraditi će se jedan stupac u skupu podataka. Ukoliko elementi popisa nisu jednako dugi, sustav R će javiti pogrešku.



- matrica će prijeći u skup podataka koji će imati jednak broj stupaca (i redaka) početnoj matrici.

**Napomena:** Predefinirano ponašenje funkcije `data.frame()` tekst prebacuje u faktore. Ukoliko to ne želimo moramo dati opciju `stringAsFactors = FALSE`. Svakako, kao i u radu s drugim funkcijama u sustavu R, najprije se treba upoznati sa sintaksom funkcije otvaranjem njezine informacijske kartice, u ovom slučaju unosom `?data.frame` u R konzolu.

Iako će o tipovima varijabli biti riječ nešto kasnije, na ovome mjestu želimo navesti da su faktori u sustavu R su način na koji R definira kategorisane varijable, varijable mjerene na nominalnoj skali. Varijable mjerene na ordinalnoj skali u sustavu R se definiraju kao posloženi (poredani) faktori (engl. *ordered factor*). Faktori se u sustavu R definiraju putem naredbe `gl()`, ali pretvaranjem već postojećeg vektora u faktor funkcijom `factor()`. Definiranje varijabli prema načinu njihova mjerjenja izuzetno je važno prilikom procesa analize podataka, a posebno grafičkih prikaza.

Promotrite razliku između sljedeća dva skupa podataka nastala korištenjem funkcije `data.frame()` s jednakim početnim vektorima. Najprije se upoznajmo s funkcijom i njezinom sintaksom otvarajući informacijsku karticu funkcije.

```
?data.frame
```

Primijetite na koji način sustav R sada daje informaciju o strukturi objekta klase `data.frame`, kolone naziva

```
#kreiranje skupa podataka i određivanje na koji se način ponaša prema znakovima:  
#da li da ih smatra character ili factor.  
df <- data.frame(brojevi = 20:29, slova = letters[1:10], stringsAsFactors = FALSE)  
df2 <- data.frame(brojevi = 20:29, slova = letters[1:10], stringsAsFactors = TRUE)
```

```
#pogledamo strukturu nastalih objekta  
str(df)
```

```
'data.frame': 10 obs. of 2 variables:  
 $ brojevi: int 20 21 22 23 24 25 26 27 28 29  
 $ slova : chr "a" "b" "c" "d" ...
```

```
str(df2)
```

```
'data.frame': 10 obs. of 2 variables:  
 $ brojevi: int 20 21 22 23 24 25 26 27 28 29  
 $ slova : Factor w/ 10 levels "a","b","c","d",...: 1 2 3 4 5 6 7 8 9 10
```

Poseban je slučaj podataka kodiranje klase (nominalne varijable) dodjeljivanjem broja kao identifikatora klase. Primjer za ovakvu varijablu može biti poštanski broj mjesta. Kroz primjere koji slijede, upoznat ćemo se s internim načinom čuvanja numeričkih vrijednosti koje su u svojoj suštini označe klase odnosno kategorija (faktori).

```
#prebacujemo numeričku iz skupa podataka df iz tipa character u tip factor.  
df$brojevi_f <- as.factor(df$brojevi)
```

```
#prebacujemo tek napravljanu faktorsku varijablu ponovno u varijablu tipa numeric  
#Prokomentirajte rezultat  
df$brojevi_povrat <- as.numeric(df$brojevi_f)
```

Ispravno prebacivanje faktora u numeričku varijablu:

```
#primijetite na koji način sustav bilježi varijable tipa factor  
as.numeric(levels(x))[x]  
#ispravno vraćanje faktorske varijable bilježene numeričkim znakovima  
as.numeric(as.character(x))
```

Kombiniranje (spajanje) dvaju skupa podataka putem stupaca provodi se naredbom koju smo već upoznali kod matrica `cbind()`. U tom slučaju broj redaka iz objekata koji se spajaju moraju biti jednaki. Funkcija će zanemariti ranije dodijeljena redaka `rownames()`. Napominjemo da ovaj oblik lijepljenja nema nikakve veze s relacijskim spajanjima podataka prema nekom ključu.

```
#spajanje po kolonama, ukoliko dimenzija rednaka objekata odgovara  
cbind(skup_podataka_1, data.frame(z = 4:1))
```

```
#spajanje dva objekta po redcima, ukoliko dimenzija kolona odgovara  
rbind(skup_podataka_1, data.frame(x = 5, y = "E"))
```



```
rbind(skup_podataka_1, data.frame(x = 5, y = "E"))
```

Primijetite da ovdje nije riječ o bilo kakvom relacijskom spajanju podataka (join, merge i slično) iz dva supa već se unutar funkcija `rbind()`/`cbind()`. podaci spajaju ukoliko odgovaraju prema eventualno odgovarajućim dimenzijama.

#### ZADACI ZA SAMOSTALNI RAD:

Pažljivo pročitajte zadatke koji slijede i riješite ih samostalno:

*#ZADATAK 1: Izradite logički vektor koji ćete nazvati logicki\_vektor, duljine 15 s 11 vrijednosti TRUE (T) i 4 vrijednosti FALSE (F).*

*#ZADATAK 2: Napravite sumu na vektoru l iz prethodnog zadatka (funkcija sum). #Prokomentirajte rezultat.*

*#ZADATAK 3: Izradite cjelobrojni vektor koji ćete nazvati r. #Vektor treba sadržati sve cjelobrojne vrijednosti od 1 do 100, osim sekvene od 5 do 10. Koliko elemenata ima napravljeni vektor? #Nova funkcija, seq() za generiranje sekvenci: #?seq #upoznavanje s funkcijom*

*#ZADATAK 4: Selektirajte prvih pet elemenata vektora iz predhodnog zadatka i vrijednosti sačuvajte kao objekt naziva selekt.*

*#ZADATAK 5: Selektirajte sve elemente veće od 20.*

*#ZADATAK 6: Što bismo dobili da smo samo napisali r>20 bez uglatih zagrada?*

*#ZADATAK 7: Učitajte skup podataka iris (pomoću naredbe data(iris) ili u novijim verzijama R-a samo unosom skupa podataka iris). #Odgovorite koje informacije o skupu podataka dobijete korištenjem sljedećih funkcija: #za svaku od navedenih funkcija možete otvoriti i njezinu informacijsku karticu.*

```
?class
class(iris)
?names
names(iris)
?dim
dim(iris)
dim(iris)[1]
dim(iris)[2]
?nrow
nrow(iris)
?ncol
ncol(iris)
```

*#ZADATAK 8: Napravite selekciju svakog trećeg elementa iz objekta vektor\_d*

### 2.3 Učitavanje postojećih podataka u R

R ima izuzetnu mogućnost izmjene podataka s drugim programima.

U sustav R podatke možemo učitati na mnogo načina (<https://cran.r-project.org/doc/manuals/r-devel/R-data.html>). Osnovni tipovi su učitavanje:

- **tabličnih podataka** funkcijom `read.table()` paketa `utils` unutar kojih je moguće definiranje velikog broja parametara ali i putem unaprijeđenih funkcija osnovnog sustava, primjerica funkcija `read.csv()`, `read.csv2()`, `read.delim()`, `read.delim2()` koje nisu ništa drugo no pojednostavljena funkcija `read.table()` s dijelom predodređenih parametara radi lakšeg korištenja; Google tablice funkcijom `gs_ls()` paketa `googlesheets()`



- **teksta** `readLines()` ili `scan()`
- **web stranice** funkcijom `getURL()` paketom `XML` za stvaranje veze te čitanje funkcijom `readLines()` ili funkcijom `download.file()` paketa `utils`
- **kôda sustava R** funkcijom `source()`
- **JavaScript** objekata funkcijom `fromJSON()` paketa `JSON`
- **objekata sustava R** funkcijama `load()` i `dget()` u ovisnosti kojim su funkcijama objekti snimani (`save()` ili `dput()`)
- **binarnih objekata** funkcijom `unserialize()` ali i za pristup različitim hijerarhijskim formatima podataka kao što su `hdf5`, `h5r`, `rhdf5`, `RNetCDF`, `ncdf` i `ncdf4`
- **radnog prostora sustava R** (engl. `workspace`) `load()`
- specifično razvijeni paketi i funkcije za učitavanje SAS-datoteka (paket `sas7bdat`), prostornih podataka (paketi `rgdal`, `shapefiles`, `maptools`, `maps` i mnogi drugi)
- i mnogi drugi.

Postoji niz razvijenih specifičnih paketa za rad npr. datotekama programa Excel (paketi `xlsx` i `openxlsx`) ili pak MODIS satelitskim snimkama (paketi `MODIS` i `MODISTools`). Velik broj formata moguće je učitati uz pomoć paketa `foreign`. SPSS, STATA i SAS datoteke se mogu učitavati i uz pomoć paketa `Hmisc`.

Izuzetno važan aspekt za svaki programski paket je njegova mogućnost pristupa podacima pohranjenim u različitim bazama podataka. Ključan paket za pristup bazama podataka iz R-a je paket `DBI` na kojeg se nadovezuju funkcionalnosti specifičnih načina povezivanja drugih paketa. Veoma važna je i mogućnost pristupa bazama podataka kao što su Oracle, Microsoft SQL Server, MS Access, PostgreSQL, MySQL, SQLite i druge putem ODBC pristupa bazama SQL upitima (paket `RODBC`, `RJDBC`) ali i specijaliziranih paketa za spajanja na specifične baze kao što je primjerice paket `ROracle` za spajanje specifično na Oracle baze podataka. Veoma dobro mjesto za upoznavanje s načinima povezivanja na različite baze podataka je poveznica RStudija <https://db.rstudio.com/>. Iz sustava R možete pristupiti i ne baš tako čestim ali specifičnim bazama podataka, primjerice bazi podataka specijaliziranoj za unos inf čcija specifična sintaksa omogućuje i ulančavanjem procesa analize (engl. `pipe`).

Kako je R zajednica postala izuzetno brojna tako se i funkcionalnosti sustava mijenjaju veoma brzo. Ovo treba uvijek imati na umu i pravo stanje sustava u nekom području dodatno pretražiti na Internetu, a posebno i na informacijskim kanalima.

### 2.3.1 Funkcije u sustavu R

R je tzv. funkcionalan program koji se fokusira na izradu i upravljanje funkcijama. Osnovno je pravilo da se unutar sustava R s funkcijama može napraviti sve što se može napraviti s vektorima - mogu se dodjeliti nekoj varijabli, spremiti na popis, biti argumenti nekih drugih funkcija. Čak je moguće kreirati bezimenu funkciju te funkciju unutar funkcije. Funkcije čine najmoćniji dio sustava R i brišu granice između programera sustava i korisnika. Korisnik može izraditi svoju funkciju uporabom naredbe `function()` i u osnovi ima ovaj oblik:

```
#function (arglist) {body}
f <- function(argumenti funkcije) {# funkcija nešto radi, tijelo funkcije}
```

Ukoliko nismo upoznati s funkcijom koju želimo koristiti trebamo otvoriti njezinu informacijsku karticu (ukoliko je funkcija dio paketa s repozitorija) ili zatražiti popis argumenata funkcije koja se nalazi u radnom prostoru funkcijom `args()`:

```
args(f)
```

U sljedećem primjeru pitamo koji su sve argumenti funkcije za izračun standardne devijacije uzorka, `sd()`:

```
args(sd)
```

```
function (x, na.rm = FALSE)
NULL
```

Kako smo već spomenuli, funkcije su također objekti unutar sustava, objekti s kojima se može raditi sve ostalo - objekti prve klase. Formalni argumenti su uključeni u samu definiciju funkcije (funkcija `formals()` dat će nam popis formalnih argumenata za traženu funkciju). Formalni argumenti koji su i imenovani mogu imati predodređene vrijednosti (engl. `default`).



### 2.3.2 Spajanje argumenata funkcija

Argumenti se spajaju (engl. *match*) na tri načina: prema imenu, prema djelomičnom imenu ili prema poziciji.

Primijetite na sljedećem primjeru da će sustav sve dolje napisane linije tretirati jednako. Koji način spajanja argumenata je primjenjen u kojem dijelu? Prokomentirajte.

```
#osiguravamo jednake podatke u procesu generiranja slučajne varijable
set.seed(1) #definiranje početne vrijednosti generatora slučajnih brojeva

#generiramo podatke
moji_podaci <- rnorm(150) #slučajna normalna varijabla sa 150 vrijednosti

#tražimo standardnu devijaciju (uzorak)
sd(moji_podaci)
```

[1] 0.9041799

Funkcija *sd()* računa standardnu devijaciju na zadanom argumentu (ako pogledamo pomoć za funkciju *sd()*, vidimo da ona ima dva argumenta *sd(x, na.rm = FALSE)* i prvi je *x* tj. vektor za koji računamo standardnu devijaciju. Stoga nije potrebno u funkciju pisati i ime argumenta, već samo ime objekta za izračun, iako i sljedeće vrijedi:

```
sd(x=moji_podaci)
```

ali i ovo:

```
sd (x=moji_podaci, na.rm=F)
```

U prethodnom retku eksplisitno smo rekli što želimo da funkcija *sd()* radi s nedostajućim vrijednostima iako je i automatski predodređena vrijednost (engl. *default*) definira jednako. Argumente koji imaju automatski definirane postavke važno je navesti tek onda kada predodređene vrijednosti želimo promjeniti. Svakako se treba, prije izvršenja bilo koje funkcije u sustavu R, upoznati se s algoritmom i predodređenim vrijednostima argumenata funkcije. Spajanje argumenata putem naziva:

```
sd(na.rm=FALSE, x=moji_podaci)
```

ili:

```
sd(na.rm=FALSE, moji_podaci)
```

**Napomena:** R razlikuje velika i mala slova.

Ako pogledamo način uporabe funkcije *lm()* iz paketa *stats*

```
?lm()
```

Iz informacija o funkciji vidljivo je da funkcija nema predefinirane vrijednosti za prvi pet argumenata, od kojih su neki obavezni argumenti funkcije, a neki tek opcionali. Da bi funkcija mogla raditi potrebno je definirati sve obavezne argumente bez predodređenih vrijednosti.

Specijalna vrsta argumenata '...' može sadržavati bilo koji broj dodatnih argumenata. Prije no što korisnik započne s pisanjem vlastitih funkcija dobro je upoznati se s načinima definicije argumenata, njihove evaluacije i drugome na stranicama <http://cran.r-project.org/doc/manuals/r-release/R-lang.html#Functions>. Osim što je prilagođen jednostavnom pisanju korisničkih funkcija, sustav R omogućava i jednostavnu pripremu dokumentacije o funkciji (<http://cran.r-project.org/doc/manuals/R-exts.html#Writing-R-documentation>).

Izuzetno dobar tekst o vrstama funkcija u sustavu R te njihovoj izradi i djelovanju možete pročitati ovdje: <http://adv-r.had.co.nz/Functional-programming.html>.

U nastavku su primjeri funkcija koje računaju elementarne statistike u sustavu R. Ovo su uglavnom tzv. primitivne funkcije koje u svojoj biti nemaju kôd sustava R, već izravno pristupaju kôdu C++:

```
#stvaranje podataka
```

```
x <- c(8,17,25,67,13,26,34, 74, 15)
y <- c(81,12,53,57,15,12,45, 100, 13)
```



```
#spajanjem dva vektora funkcijom cbind dobit ćemo objekt klase matrix  
X <- cbind(x,y)
```

Funkcije deskriptivne statistike:

```
#minimalna vrijednost u objektu  
min(x)
```

```
[1] 8
```

```
#maksimalna vrijednost  
max(x)
```

```
[1] 74
```

```
#suma elemenata  
sum(x)
```

```
[1] 279
```

```
#raspon vrijednosti  
range(x)
```

```
[1] 8 74
```

```
#kumulativna sume a elemenata  
cumsum(x)
```

```
[1] 8 25 50 117 130 156 190 264 279
```

```
#kumulativni umnožak  
cumprod(x)
```

```
[1] 8.000000e+00 1.360000e+02 3.400000e+03 2.278000e+05 2.961400e+06  
[6] 7.699640e+07 2.617878e+09 1.937229e+11 2.905844e+12
```

```
#razlika  
diff(x)
```

```
[1] 9 8 42 -54 13 8 40 -59
```

```
#sumarna statistika  
summary(x)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
8	15	25	31	34	74

```
#srednja vrijednost, aritmetička sredina  
mean(x)
```

```
[1] 31
```

```
#medijan  
median (x)
```

```
[1] 25
```

```
#standardna devijacija uzorka  
sd(x)
```

```
[1] 23.76973
```

```
#ista funkcija primjenjena na objektu klase matrix  
#prisjetite se što su to generičke funkcije  
sd(X)
```

```
[1] 28.46974
```



```
#varijanca uzorka  
var(X)
```

```
      x       y  
x 565 450.000  
y 450 1074.861
```

```
#kovarijanca  
cov(X)
```

```
      x       y  
x 565 450.000  
y 450 1074.861
```

```
#korelacijski koeficijenti  
cor(X)
```

```
      x       y  
x 1.000000 0.577447  
y 0.577447 1.000000
```

```
#kvantili za zadanu vjerojatnost  
quantile(x, 0.75)
```

```
75%  
34
```

```
#kvantili za zadanu vjerojatnost, predodređeni vektor vjerojatnosti  
quantile(x)
```

```
0% 25% 50% 75% 100%  
8   15   25   34   74
```

Dodatne funkcije neophodne u osnovnoj pripremi i analizi podataka:

```
#rangiranje vrijednosti  
rank(x)
```

```
[1] 1 4 5 8 2 6 7 9 3
```

```
#sortiranje vrijednosti  
sort(x)
```

```
[1] 8 13 15 17 25 26 34 67 74
```

```
#indeks poredka sortiranja vrijednosti  
order(x)
```

```
[1] 1 5 9 2 3 6 7 4 8
```

```
#kategorizacija kontinuirane varijable u zadane klase  
moja_razdioba <- cut(x, breaks=c(1,20,40,60,80))
```

### 2.3.3 Operatori u sustavu R

Operatori u sustavu R predstavljeni su na sljedeći način:

####Aritmetički operatori

Aritmetički operatori u R-u su operatori:

- – zbrajanja
- – oduzimanja
- – množenja



- / dijeljenja
- ^ ili \*\* eksponencije
- x %% y x sadržan u y, pripada u y.

#### ####Logički operatori

Logički operatori u R-u su operatori:

- <, <=, >, >=, == - manje od; manje ili jednako; veće, veće ili jednako, identično
- != - nije jednako
- !x - nije x
- x | y - x ili y
- x & y - x i y.

Unutar sustav postoje specijalizirane funkcije koje rade operacije na dva skupa podataka te rade uniju (*union()*), presjek (*intercest()*), razliku (*setdiff()*), testiraju jednakost (*setequal()* i *identical()*), pripadnost (*is.element()*).

Primjer korištenja spomenutih funkcija *base* paketa u radu s podacima. Koristit ćemo skup podataka o cijenama nekretnina u gradu Ames. Opis podataka može se pronaći na <https://ww2.amstat.org/publications/jse/v19n3/decock/DataDocumentation.txt>. Ovaj skup podataka veoma se često koristi prilikom upoznavanja s različitim segmentima analize podataka i metodama analize. Najprije ćemo pristupiti bazi podataka na web stranicama Openintro.org

```
#uzimamo podatke s wwb stranice i snimamo ih u radni direktorij
download.file("http://www.openintro.org/stat/data/ames.RData", destfile = "ames.RData")
```

Prisjetimo se funkcije *list.files()* kojom tražimo ispis svih datoteka u zadanim direktoriju, u ovom slučaju radnom direktoriju.

```
list.files(getwd(), pattern="RData")
```

Podaci se za sada se ne nalaze u radnom prostoru.

```
ls()
```

Učitajmo podatke u R konzolu:

```
load("ames.RData")
```

Pogledajmo sada popis objekata u radnom prostoru:

```
ls()
```

Pogledajmo strukturu/klasu/podatke/dimenzije/ varijabli objekta naziva ames:

```
#struktura
str(ames)

#klasa objekta
class(ames)

#prvih 6 podataka
head(ames)

#dimenziju podataka
dim(ames)
nrow(ames); ncol(ames)

# dimenzija (redaka/kolona)
# varijabli (kolona)
#na klasi objekta data.frame jednako kao i colnames(ames)
names(ames)
```

U sljedećih nekoliko koraka napravit ćemo drugi skup podataka, modifikacijom skupa ames te napraviti nove varijable ali i obrisati neke prvobitne varijable.



```
#radimo kopiju podataka
ames_2 <- ames

#računamo nove varijable kao kombinacije starih

#ames$SalePrice u milijunima $
ames$SalePrice_m <- ames$SalePrice / 1000000

#SalePrice/Gr.Liv.Area
ames_2$cijena_jed_pov <- ames$SalePrice / ames$Gr.Liv.Area

#razdjeljujem cijenu po jedinici površine na tri jednake kategorije
ames_2$cijena_jed_pov_cat1 <- cut(ames_2$cijena_jed_pov, breaks=c(3))

#razdjeljujem cijenu po jedinici površine na točno zadane kategorije
ames_2$cijena_jed_pov_cat2 <- cut(ames_2$cijena_jed_pov, breaks=c(50, 100, 200))

#razdjeljujem cijenu po jedinici površine prema kvantilnoj statistici,
#dvije kategorije, iznad i ispod medijana
ames_2$cijena_jed_pov_cat3 <- cut(ames_2$cijena_jed_pov,
                                    breaks=c(summary(ames_2$cijena_jed_pov)[1], #minimum
                                             summary(ames_2$cijena_jed_pov)[3], #median
                                             summary(ames_2$cijena_jed_pov)[6])) #maksimum
```

Sada ćemo pokušati funkcijama ispitati razlike između dva skupa podataka. Veoma često radeći analize podataka radimo s veoma velikim skupovima podataka i nije toliko jednostavno prepoznati razlike između dva skupa ali i generalno se snalaziti u različitim skupovima podataka.

```
#varijable koje se nalaze u oba skupa podataka
intersect(names(ames), names(ames_2))

#jednako kao i
setequal(names(ames), names(ames_2))

#koje varijable se nalaze u x a nema i u y
setdiff(names(ames), names(ames_2))

setdiff(names(ames_2), names(ames))

#postoji li varijabla SalePrices u skupu ames?
is.element("SalePrices", names(ames))

#postoji li varijabla SalePrice u skupu ames?
is.element("SalePrice", names(ames))

#koliko jedinstvenih varijabli imamo za pojedinu prodanu kuću u oba skupa podataka?
union(names(ames), names(ames_2))
```

#### ZADACI ZA SAMOSTALAN RAD:

```
#ZADATAK 1: U skupovima A i B
#Koliko se djece ponavlja u razredima A i B.
#Drugačije rečeno, pronađite zajedničke elemente skupova A i B.
A <- c("Petra", "Mirna", "Luka", "Karla", "Iva", "Tea", "Lucija",
      "Lea", "Sara", "Kristina", "Ivan", "Ivana", "Jana")

B <- c("Luka", "Sara", "Jure", "Kristijan", "Iva", "Igor",
      "Petra", "Mirna", "Ivan", "Katja", "Petar", "Karla",
      "Sven", "Jan", "Luka")
```



```
#ZADATAK 2: Koja se pojavljuju jedino u razredu A?
```

```
#ZADATAK 3: Koja se pojavljuju jedino u razredu B?
```

### 2.3.4 Distribucije vjerojatnosti u sustavu R

Sustav R posjeduje bogatu obitelj funkcija vjerojatnosnih distribucija kao što su na primjer normalna, beta, binomna, gama, hipergeometrijska itd.

Funkcija `distributions()` iz paketa stats dat će nam popis svi ugrađenih distribucija u sustavu R.

```
#upoznajmo se s osnovnom funkcijom koja će nam dati smjernice za dalje  
?distributions
```

**2.3.4.1 Funkcija gustoće `d()`** Za svaku od navedenih funkcija vjerojatnosne razdiobe postoje četiri funkcije koje čine sljedeće: Funkcija `d(x,...)` računa gustoću na x-u:

```
#upoznajmo se sa sintaksom funkcije  
#otvara nam informacijsku karticu za normanu razdiobu slučajne varijable  
?dnorm
```

```
dnorm(0) #gustoća na 0; standardna normalna distribucija
```

```
[1] 0.3989423
```

**2.3.4.2 Funkcija vjerojatnosti `p()`** Funkcija `p(q, ...)` računa kumulativnu vjerojatnost do q  $P(x \leq q)$ . Kolika je kumulativna vjerojatnost na zadanom kvantilu raspodjele. Primjer koji slijedi, također, je napravljen na standardnoj normalnoj distribuciju:

```
#kumulativna distribucija za dosezanje Z vrijednosti P(Z<1.96)  
pnorm(1.96)
```

```
[1] 0.9750021
```

**2.3.4.3 Funkcija kvantila `q()`** Funkcija `q(p, ...)` računa kvantil za zadanu kumulativnu vjerojatnost, zadani p

```
#na kojem kvantilu standardne normalne raspodjele će  
#kumulativna vjerojatnost doseći 0.89?  
qnorm(0.89) #89-ti percentil
```

```
[1] 1.226528
```

**2.3.4.4 Funkcija generiranja slučajne varijable prema zadanoj raspodjeli `r()`** Funkcija `r(n, ...)` generira slučajnu varijablu zadane veličine

```
#postavljamo generator slučajnih brojeva na jednaku vrijednost  
#kako bismo svi imali jednaku generirane vrijednosti  
set.seed(1)

#generiramo slučajnu varijablu iz standardne normalne distribucije  
#zadane veličine, 40  
rnorm(40)
```

```
[1] -0.62645381 0.18364332 -0.83562861 1.59528080 0.32950777
[6] -0.82046838 0.48742905 0.73832471 0.57578135 -0.30538839
[11] 1.51178117 0.38984324 -0.62124058 -2.21469989 1.12493092
[16] -0.04493361 -0.01619026 0.94383621 0.82122120 0.59390132
[21] 0.91897737 0.78213630 0.07456498 -1.98935170 0.61982575
```



```
[26] -0.05612874 -0.15579551 -1.47075238 -0.47815006  0.41794156
[31]  1.35867955 -0.10278773  0.38767161 -0.05380504 -1.37705956
[36] -0.41499456 -0.39428995 -0.05931340  1.10002537  0.76317575
```

Pokazan je primjer za normalnu distribuciju (norm), ali vrijedi za sve ranije nabrojane distribucije u sustavu.

###Osnovno o petljama u sustavu R

Unutar sustava R postoji nekoliko načina petlji koje su veoma učinkovite. Osim standardnih for i while petlji, sustav R ima razvijene i nestandardne petlje koje su izražene kako bi radile ponavljajuće stvari u specifičnim slučajevima. To je tzv. apply obitelj petlji, a osim funkcije *apply()* sadrži i funkcije *lapply()*, *sapply()* i *tapply()* te još nekoliko o kojima na ovome tečaju neće biti riječi. Slijedi vrlo kratak osvrt na spomenute petlje, specifične za sustav R.

####Funkcija *apply()*

Ovo je funkcija koja provodi drugu zadanu funkciju na zadanim marginama (redak ili stupac) matrice ili polja. Prisjetimo se načina indeksiranja u R-u. Prvi indeks je indeks redka (margina 1), drugi indeks je indeks stupca/kolone (margina 2).

Osnovni oblik petlje. Funkcija koju primjenjujemo može biti bilo koja ugrađena funkcija u R-u ili korisnička funkcija koja odgovara podacima koje petlji dajemo.

**apply(X, MARGIN, FUN, ...)**

**Primjer:**

U sljedećem primjeru koristimo četiri numeričke varijable iz skupa podataka *iris* koji dolazi instalacijom R-a.

```
#izdvajamo numeričke varijable iz skupa podataka
moji_podaci2 <- iris[,1:4]
```

Pogledajmo strukturu i prvih nekoliko podataka:

```
str(moji_podaci2)
```

```
'data.frame': 150 obs. of 4 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
```

```
head(moji_podaci2)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5.0	3.6	1.4	0.2
6	5.4	3.9	1.7	0.4

Računamo željene statistike, ovdje sredine, po željenoj margini, u našem primjeru redcima:

```
#funkcija se provodi po redcima, margina 1
apply(moji_podaci2, 1, max)
```

```
[1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4
[18] 5.1 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4 5.2 5.5
[35] 4.9 5.0 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.6 5.3 5.0 7.0
[52] 6.4 6.9 5.5 6.5 5.7 6.3 4.9 6.6 5.2 5.0 5.9 6.0 6.1 5.6 6.7 5.6 5.8
[69] 6.2 5.6 5.9 6.1 6.3 6.1 6.4 6.6 6.8 6.7 6.0 5.7 5.5 5.5 5.8 6.0 5.4
[86] 6.0 6.7 6.3 5.6 5.5 6.1 5.8 5.0 5.6 5.7 5.7 6.2 5.1 5.7 6.3 5.8
[103] 7.1 6.3 6.5 7.6 4.9 7.3 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7
[120] 6.0 6.9 5.6 7.7 6.3 6.7 7.2 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7
[137] 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8 6.7 6.7 6.3 6.5 6.2 5.9
```

Računamo željene statistike, ovdje mjere deskriptivne statistika, funkcija *summary()*, po željenoj margini, u našem primjeru redcima:



```
#funkcija se provodi po stupcima, margina 2
apply(moji_podaci2, 2, summary)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
Min.	4.300000	2.000000	1.000	0.100000
1st Qu.	5.100000	2.800000	1.600	0.300000
Median	5.800000	3.000000	4.350	1.300000
Mean	5.843333	3.057333	3.758	1.199333
3rd Qu.	6.400000	3.300000	5.100	1.800000
Max.	7.900000	4.400000	6.900	2.500000

####Funkcija *lapply()*

Ovo je funkcija koja primjenjuje zadanu funkciju na svaki element liste. Rezultat funkcije je lista.

Kako bismo demonstrirali rad funkcije radimo listu. Funkcija koju želimo primijeniti na sve elemente mora odgovarati tipu objekata u listi. Kako u primjeru koji slijedi primjenjujemo funkcije koje je logično i jedino moguće primjenjivati na numeričkim podacima tako nam svi elementi liste moraju biti numerički.

```
#izrada liste
```

```
numerička_lista <- list(var1 = 1:5, var2 = rnorm(10))
```

```
#pogledajmo strukturu objekta
```

```
str(numerička_lista)
```

```
List of 2
$ var1: int [1:5] 1 2 3 4 5
$ var2: num [1:10] -0.165 -0.253 0.697 0.557 -0.689 ...
```

Primjenjujemo željenu funkciju na svaki element liste:

```
#tražimo najveću vrijednost u svakom pojedinom elementu liste
lapply(numerička_lista, max)
```

```
$var1
[1] 5
```

```
$var2
[1] 0.8811077
```

```
#isprobajte različite funkcije, primjerice summary
lapply(numerička_lista, summary)
```

```
$var1
  Min. 1st Qu. Median   Mean 3rd Qu.   Max.
  1       2       3       3       4       5
```

```
$var2
  Min. 1st Qu. Median   Mean 3rd Qu.   Max.
-0.7075 -0.2312  0.1261  0.1341  0.6619  0.8811
```

####Funkcija *sapply()*

Ova je funkcija slična funkciji *lapply()* s razlikom da rezultat pokušava strukturirati u jednostavniji oblik ako je to moguće.

####Funkcija *tapply()*

Ovo je funkcija koja radi određeni proces na selekciji vektora te na poljima s promjenjivim duljinama. Grupiranje unutar petlje *tapply* definira se faktorskom varijablom.

```
#osiguravamo reproducibilan primjer
set.seed(1)
```

```
#radimo podatke koji imaju jednu faktorsku varijablu (funkcija gl)
podaci <- data.frame(osoba = 1:100, tlak = rnorm(100, mean = 120, sd = 20),
```



```
tretman = gl(2, 50, labels = c("Tretman", "Kontrola"))
```

#pogledajmo strukturu nastalih podataka  
summary(podaci)

osoba	tlak	tretman
Min. : 1.00	Min. : 75.71	Tretman :50
1st Qu.: 25.75	1st Qu.:110.12	Kontrola:50
Median : 50.50	Median :122.28	
Mean : 50.50	Mean :122.18	
3rd Qu.: 75.25	3rd Qu.:133.83	
Max. :100.00	Max. :168.03	

#pogledajmo stupaca/kolona/varijabli u skupu podataka  
names(podaci)

```
[1] "osoba"    "tlak"      "tretman"
```

Primijenimo funkciju *summary()* za varijablu *tlak* posebno za svaki nivo faktorske varijable *tretman*. Primjer koji slijedi je primjer s dva nivoa faktora ali mogli bismo imati skup podataka sa znatno većim brojem nivoa, primjerice rezultati testa matematike državne mature s informacijom o školi ili razredu.

```
tapply(podaci$tlak, podaci$tretman, summary)
```

\$Tretman

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
75.71	112.56	122.58	122.01	134.56	151.91

\$Kontrola

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
83.9	108.6	122.3	122.3	132.1	168.0

**2.3.4.5 Dodatne funkcije nužne za početak rada** U nastavku dajemo skup naredbi koje je dobro poznavati prilikom početka rada u sustavu R.

Pregled dostupnih mogućnosti sustava:

```
#ispis dostupnih mogućnosti sustava - prvih 10
help(options)
```

Pregled trenutnih postavki mogućnosti sustava:

```
options()[1:10]
```

```
$add.smooth
[1] TRUE
```

```
$browserNLdisabled
[1] FALSE
```

```
$CBoundsCheck
[1] FALSE
```

```
$check.bounds
[1] FALSE
```

```
$citation.bibtex.max
[1] 1
```

```
$continue
[1] "+"
```



```
$contrasts
  unordered          ordered
"contr.treatment"    "contr.poly"

$defaultPackages
[1] "datasets"   "utils"      "grDevices"  "graphics"   "stats"       "methods"

$demo.ask
[1] "default"

$deparse.cutoff
[1] 60
```

Namještanje formata brojeva pri ispisu - tri decimalne:

```
options(digits=3)
```

Listanje svih datoteka u radnom direktoriju:

```
dir()
```

ili

```
list.files()
```

Pregled posljednjih 25 naredbi:

```
history()
```

Prisjećanje zadanih naredbi moguće je i uz pomoć strelica na tipkovnici računala.

Pohranjivanje tijeka naredbi u datoteku (format .Rhistory):

```
#spremamo na disk cijelu povijest rada tijekom jedne sesije
#otvaranjem ove datoteke u novoj sesiji omogućit će nam nastavak rada
#i primjerice, korištenje funkcije history() po otvaranju sesije
savehistory(file="moja_povijest.Rhistory")
```

Unos tijeka naredbi u sustav (format .Rhistory):

```
#učitavamo povijest naredbi
loadhistory(file="moja_povijest.Rhistory")
```

Spremanje cjelokupnog radnog prostora (format .Rdata):

```
save.image("moj_radni_prostor.RData")
```

Učitavanje radnog prostora u sustav:

```
load("moj_radni_prostor.RData")
```

Spremanje određenog objekta u datoteku (na primjer tekstualnu datoteku):

```
write.table(objekt, file = "ime_datoteke")
```

```
write.table(mydata, "c:/podaci.txt", sep="\t") #za windows definiranje kraja redka
```

Kada su podaci nepoznatog formata ili za učitavanje s ekrana korisna je funkcija `scan()`.

#### Napomena:

Ako nije navedena putanja u koju se datoteka snima, sustav R prepostavlja da je tražen radni direktorij.

```
save(object=list,file="moj_radni_prostor_dio.RData")
```

Brisanje objekta iz radnog prostora:



```
rm(ime objekta)
```

Brisanje svih objekata iz radnog prostora:

```
rm(list=ls())
```

Brisanje određenog objekta ili njegovog dijela:

```
ime objekta <- NULL
```

#### Napomena:

Ako nije navedena putanja u koju se datoteka snima, sustav R prepostavlja da je tražen radni direktorij.

```
load("moja_datoteka.RData")
```

Izlaz iz konzole sustava R (korisnik je u tom trenutku upozoren na snimanje radnog prostora):

```
q()
```

ili

Izborom na grafičkom sučelju (GUI) File - Exit

ili

uobičajenim zatvaranjem programa operacijskog sustava Windows, klikom lijevom tipkom miša na znak x u gornjem desnom kutu.

### 2.3.5 Osnove o datumima u sustavu R

Radi važnosti pravilnog definiranja datuma prilikom vizualizacije i analize podataka ovdje navodimo neophodne osnove.

Datumi se unutar klase *Date* čuvaju (internu unutar sustava R) kao broj dana od 1.1.1970. godine. Ako je potrebno čuvati i vrijeme, koriste se POSIX klase koje se razlikuju po internom načinu čuvanja. Klasa *POSIXct* čuva vrijeme kao broj sekundi od 1.1.1970. *POSIXlt* čuva vrijeme kao objekt klase *list* (sat, minuta, sekunda, engl. *hour, min, sec, mon,..*) čime se lakše dohvaćaju pojedini elementi, primjerice pojedini član zapisa, mjesec (engl. *mon*).n.

Ovaj način definiranja formata datuma omogućava različite izračune o protjeku vremena između dva datuma. Funkcija *as.Date()* prebacuje datum zapisan kao tekst (character) u datum klase *Date*.

**2.3.5.1 Format zapisa datumskih varijabli** Ukoliko unosimo podatke ili ih stvaramo u sustavu R, predefinirani oblik karkternog zapisa koji sustav očekuje kako bi pravilno karakter konvertirao u varijablu klase *Date* je u obliku YYYY-MM-DD. Često imamo informaciju o datumu zabilježenu u nekom drugom poretku i formatu. U takvom slučaju moramo sustavu jasno reći koji je ulazni oblik karakterne varijable koju želimo prebaciti u datumsku varijablu.

Ukoliko otvorimo informacijsku karticu za funkciju *strftime()* kojom varijable tipa *string / character* prebacujemo u varijable klase *POSIXlt* i *POSIXct* na načina da im točno definiramo strukturu ulaznog formata tipa *string / character*.

Uz nekoliko promjera ćemo se upoznati s načinom rada s datumskim varijablama:

```
#primjer datuma napisanog kao string te ga pretvaramo u klasu Date  
dat<- as.Date("1990-10-01")  
class(dat)
```

```
[1] "Date"
```

```
#specificirajmo format datuma
```

```
#definirajmo što znači koji dio karkternog zapisa  
as.Date("08/30/2016", format = "%m/%d/%Y")
```

```
[1] "2016-08-30"
```



```
#ukoliko radimo unos podataka iz nekog drugog softwera, trebamo specificirati i origin  
#u primjeru koji slijedi, primjer je unosa iz excela koji za origin ima 01.01.1900.  
as.Date(41149, origin = "1900-01-01")
```

```
[1] "2012-08-30"
```

```
#datumi su prave numeričke varijable te se s njima može računati  
#koliko je dana proteklo od određenog datuma do danas  
Sys.Date() - as.Date("2016-07-01")
```

Time difference of 577 days

```
#prema želji odredimo i drugačiju mjeru jedinicu koristeći funkciju difftime()  
difftime(Sys.Date(), as.Date("2016-07-01"), units = "hours")
```

Time difference of 13848 hours

### 2.3.5.2 Klase za čuvanje datumskih varijabli unutar sustava R

Sustav R ima poseban način prezentacije vremena:

*POSIXct* klasa za prikaz vremena - u pozadini velika cijelobrojna vrijednost, format koristan kada informaciju o datumu želite spremiti u objekt sličan skupu podataka.

```
Sys.time()
```

```
[1] "2018-01-29 10:24:06 CET"
```

```
class(Sys.time())
```

```
[1] "POSIXct" "POSIXt"
```

Klasa za prikaz datuma *POSIXlt* u obliku liste koja nosi velik broj dodatnih informacija kao što su dan u tjednu, mjesec, dan u mjesecu.

```
obj.lt <- as.POSIXlt (Sys.time())
```

```
#trenutno vrijeme, klasa POSIXct, sekunde protekle od 01.01.1990.  
unclass(Sys.time())
```

```
[1] 1.52e+09
```

```
# kao lista POSIXlt  
#jednostavniji dohvati velikog broja parametara  
unclass(as.POSIXlt(Sys.time()))
```

```
$sec
```

```
[1] 6.87
```

```
$min
```

```
[1] 24
```

```
$hour
```

```
[1] 10
```

```
$mday
```

```
[1] 29
```

```
$mon
```

```
[1] 0
```

```
$year
```

```
[1] 118
```

```
$wday
```



```
[1] 1  
  
$yday  
[1] 28  
  
$isdst  
[1] 0  
  
$zone  
[1] "CET"  
  
$gmtoff  
[1] 3600  
  
attr(, "tzone")  
[1] ""      "CET"   "CEST"
```

Neke korisne funkcije za rad s datumima:

```
weekdays(dat) #01.10.1990. je bio ponedjeljak  
[1] "ponedjeljak"  
months(dat) #R prepoznaje regionalne postavke računala  
[1] "listopad"  
quarters(dat) #spada u posljednji kvartal u godini  
[1] "Q4"  
#pazite na dobiveni format podatka  
unclass(dat)  
[1] 7578
```

Ponekad postoji potreba da za svoja opažanja znate koji dan u godini su se dogodila. Ovaj oblik varijable veoma je važan u slučajevima kada, na primjer, želite odvojiti svoja opažanja po sezonom ili radite predikcije u tzv. cirkularnim modelima (<http://tgmstat.wordpress.com/2013/10/16/latent-gaussian-models-inla/>) ili Markovljevim lancima gdje zaključivanje o jednom stanju ovisi o stanju sustava trenutak prije. Informaciju o rednom danu u godini možemo dobiti uporabom funkcije za vremena klase POSIX, *strptime()* na način da pod argument format stavimo samo godinu:

```
dat  
[1] "1990-10-01"  
strptime(dat, format = "%j") #01.10.1970. je 274 dan u godini  
[1] "274"
```

Funkcija *strptime()* omogućava stvaranje vremena klase POSIX iz tekstualnog formata na način:

```
mojPOSIX <- strptime("2014-07-19 20:26:22", format = "%Y-%m-%d %H:%M:%S")
```

Prvi paket koji se bavio analitikom dnevnih podataka (financijskih) i koji je olakšao rad s vremenski referenciranim podacima je paket *zoo* s vlastitom klasom podataka istog. Objekti klase *zoo* naknadno su poboljšani paketom *xts* (engl. *extensible time series*) za analize vremenskih serija koji je nadogradio strukturu i metode objekata *zoo*.

Veoma popularan paket za rad s datumskim varijablama je i paket *lubridate*.

Tijekom ovog tečaja nemamo vremena detaljnije ulaziti u strukture i primjenu podataka o datumima prilikom statističke obrade podataka.

## ZA SAMOSTALNI RAD

Samostalno rješite zadatke u nastavku.



```

#ZADATAK 1: Gledajući informacijsku karticu funkcije strptime na pravilan
#način unesite karakterni, alfanumerički zapis u datumski tip varijable i
#dodijelite rezultat varijabli datum_1
#a) Tue, 23 Mar 2010 14:36:38 -0400
#b) "2006-01-08 10:07:52", "2006-08-07 19:33:02", vremenska zona EST5EDT

datum_1 <- strptime("Tue, 23 Mar 2010 14:36:38 -0400", "%a, %d %b %Y %H:%M:%S %z")

datum_2 <- strptime(c("2006-01-08 10:07:52",
                     "2006-08-07 19:33:02"), "%Y-%m-%d %H:%M:%S", tz = "EST5EDT"))

#ZADATAK 2: Izračunajte vrijeme proteklo u danima i sekundama od prethodna dva datuma.

#ZADATAK 3: Adekvatnom funkcijom ispitajte sljedeće za 13.04.2006.
#a) Koliko je dana proteklo do 15.07.2009.?
#b) Koji je to bio dan u godini?
#c) Ukoliko pripremate kvartalne izvještaje, u konačnici sa svim danima 2006. godine,
#u koji će kvartalni izvještaj ovaj datum biti uključen?

```

### 3 Uvod u grafičke sustave u R-u

Statistička vizualizacija podataka i rezultata analize jedan su od najvažnijih elemenata cijelokupnog procesa analize podataka. R je programski jezik otvorenoga kôda za statističku analizu i grafiku. R ima iscrpne i moćne grafičke mogućnosti koje su usko povezane s njegovim analitičkim mogućnostima. Iz tog razloga nužno je upoznati se s osnovama grafičkih sustava koje omogućava R.

Unutar sustava R postoji nekoliko gotovo potpuno odvojenih sustava grafičke prezentacije objekata. Tijekom ovog tečaja upoznat ćemo se s osnovama rada u dva sustava.

#### 3.1 Grafika osnovnog sustava (engl. base)

Grafika osnovnog (engl. *base*) sustava nalazi se automatski u sustavu po njegovoj instalaciji. Funkcije ovog paketa nalaze se u osnovnom paketu *graphics*, a uključuju funkcije kao što su *plot()*, *hist()* za izradu histograma, *boxplot()* za Box-Whiskers plot i još mnoge druge.

```
help(package=graphics)
```

Ako zovemo funkciju paketa *base* *plot()* i nemamo otvoren grafički prozor (engl. *plotting device*), ona ga automatski otvara. Ako želimo imati veći broj otvorenih grafičkih prozora možemo sami otvoriti novi grafički prozor funkcijom *dev.new()*. Ovo nije slučaj ukoliko radite s RStudiom.

Ako u konačnici izrađujemo neku grafiku radi prezentacije u nekom dokumentu bolje je otvoriti datotečni uređaj (engl. *file device*) nego ekran (engl. *screen device*). Grafički sustavi u sustavu R uglavnom se ne mogu miješati iako se posljednjih godina intenzivno razvijaju paketi koji omogućavaju integracije različitih sustava. Izrada grafike u sustavu *base* uglavnom se može podijeliti na nekoliko segmenta: - određujemo područje crtanja - dodajemo (funkcija *add()*) različite dijelove grafike (točke, crte, anotaciju, legendu, naslove te grafičke osi - ova funkcija uglavnom prati misaoni proces stvaranja slike).

```

#osiguravamo svima jednak rezultat
set.seed(1000)

#stvaramo dvije slučajne varijable, x i y
#izvlačimo po 500 brojeva iz normalne raspodjele
x<-rnorm(500)
y <- rnorm(500)

#generička grafička funkcija visoke razine
#x predstavlja vrijednosti x koordinate, a y - y koordinate

```



```
#pozivamo grafičku funkciju visoke razine koje inicira dijagram
plot(x,y)

#nadopunjavamo postojeći dijagram naslovom korištenjem funkcije niske razine
title(main="Dijagram raspršenja dvaju generiranih slučajnih varijabli x i y")
```

Prilikom poziva neke grafičke funkcije visoke razine otvara se i grafički prozor u sustavu R. Pažljivo pogledajte nekoliko slijedećih linija i prokomentirajte način korištenja grafičkih funkcija niske razine prilikom izrade dijagrama.

```
#radimo sekvencu koju ćemo crtati
z <- seq(min(x),max(x),length=500)

#primjer druge grafičke funkcije visoke razine
#specijalizirane za izrade histograma
hist(x, main="Gustoća varijable generirane prema slučajnoj raspodjeli", freq=F)

#uspoređujemo histogram s normalnom distribucijom pa trebamo visine na svakom x-u
y <- dnorm(z)

#nadopuna dijagrama prema želji
legend( 1.7,90, c("dnorm x"),
        text.col = "green4", lty =2 ,
        bg = "gray80")

points(-1, 0.8, col = 3)
abline(1, 1)
lines(z, y, col="blue", lwd=2)
```

S narednih nekoliko primjera i naredbi ćemo se upoznati s osnovama grafičkog sustava koji dolazi instalacijom R-a, tzv. *base* grafikom, najčešćim grafičkim parametrima, grafičkim funkcijama te korištenjem boja u sustavu R. Ovo je doista veoma bazičan uvod u način rada s najvažnijim grafičkim funkcijama i uređajima u sustavu.

Osnovni (engl. *base*) grafički sustav omogučava određivanje velikog broja parametara na grafu. Popis svih aktivnih parametara može se pronaći putem upita *?par()* ili *help(par)*.

Globalni grafički parametri utjecat će na izglede svih grafova koji se izrađuju tijekom jedne seanse, a određuju se funkcijom *par()*, uz izbor najkorisnijih argumenata u nastavku.

U samom pozivu nekog grafa moguće je neku od ovih mogućnosti promijeniti:

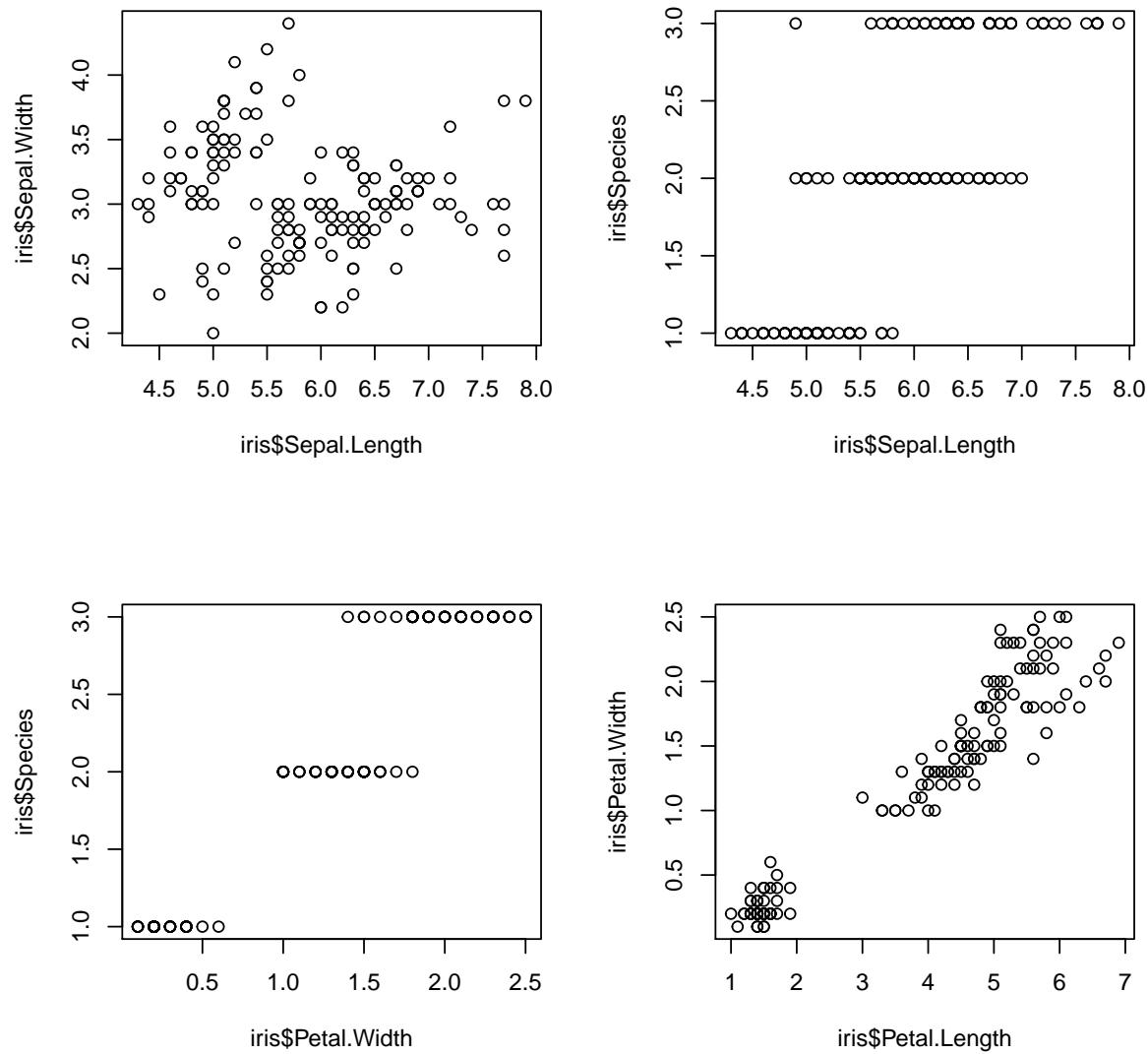
- **pch** - vrsta simbola automatski crta otvoreni krug
- **lty** - vrsta crte
- **lwd** - debљina crte
- **col** - boje korištene u crtanju (mogu se definirati uz pomoć teksta (engl. string), broja ili heksadecimalnog broja).
- **las** - orientacija osi na grafikonu
- **bg** - boja pozadine
- **mar** - određuje širinu marginu brojem linija teksta kao vektor četiri vrijednosti redom: donja, lijeva, gornja i desna margina. Alternativa je mai koja definira širinu marginu u inčima
- **oma** - vanjske marge; važno jedino kod višestrukih plotova (alternativno mai u inčima)
- **mfrow\*\*** - broj grafike u jednom retku (popunjava se unutar retka)
- **mfcoll** - broj grafike u jednom stupcu (popunjava se unutar stupca).

Provjerite zadane vrijednosti unutar sustava za neke od navedenih grafičkih parametara:

```
#lista svih grafičkih parametara i njihovih vrijednosti
par()

#vrijednost parametra u sustavu
par("lty")
par("pch")
```





Slika 4: Organizacija većeg broja grafikona, dva reda i dva stupca

```
#proizvoljni podaci za crtanje - iris

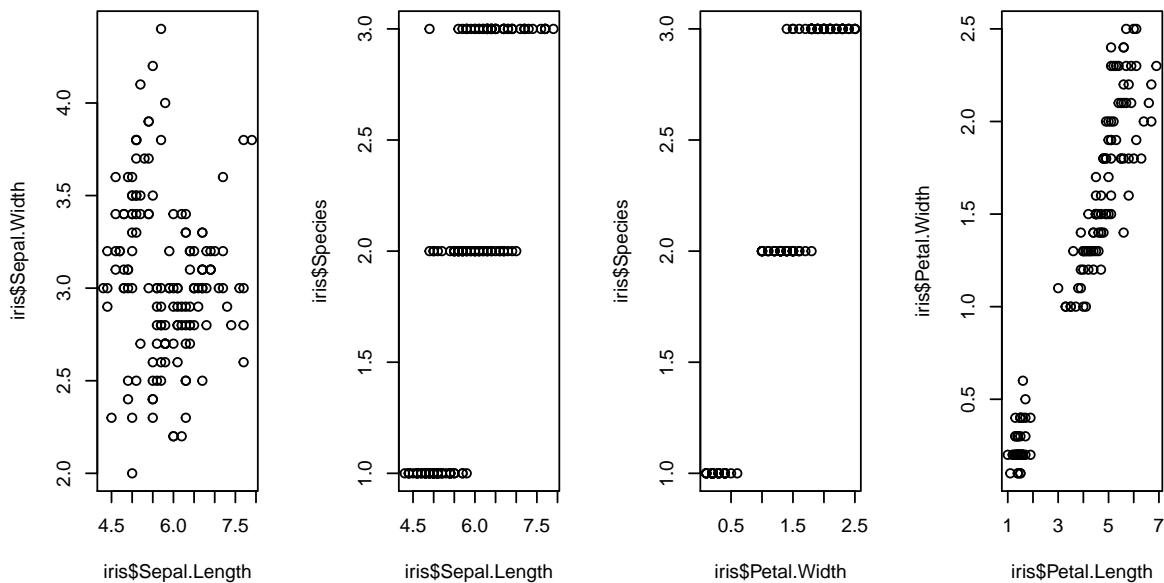
#ukoliko želimo 4 grafika, po dva u dva retka
par(mfrow=c(2,2))

#četiri proizvoljna grafikona
plot(iris$Sepal.Length, iris$Sepal.Width)
plot(iris$Sepal.Length, iris$Species)
plot(iris$Petal.Width, iris$Species)
plot(iris$Petal.Length, iris$Petal.Width)

#ukoliko želimo 4 grafika, po dva u dva retka
par(mfrow=c(1,4))

#četiri proizvoljna grafikona
plot(iris$Sepal.Length, iris$Sepal.Width)
plot(iris$Sepal.Length, iris$Species)
```





Slika 5: Organizacija većeg broja grafikona, jedan redak i četiri stupca

```
plot(iris$Petal.Width, iris$Species)
plot(iris$Petal.Length, iris$Petal.Width)
```

### 3.1.1 Grafički uređaji

Kolokvijalno rečeno, grafički uređaj je medij na kojem želimo prikazati neku grafiku. Primjeri uređaja su grafički prozor programa (grafički prozor RStudija - RStudioGD), *JPG*, *PNG*, *PDF*, *TIFF* i mnogi drugi.

Ukoliko radimo na Windows platformi i pozovemo neku od funkcija koja stvara neku grafiku, grafički prozor će se automatski otvoriti. Ukoliko želimo otvoriti novi grafički prozor bez pozivanja takve funkcije moramo koristiti funkciju `windows()`. U Rstudiju rad s višestrukim grafikonima nešto je drugačiji i s njime ćemo se upoznati tijekom rada u programu.

U primjeru koji slijedi dan je generalni način pohrane ili eksporta grafičkih prikaza na željeni grafički uređaj. Pokazujemo primjer za grafički uređaj *JPG* ali proces je jednak i za bilo koji drugi željeni uređaj. Svakako, prije korištenja novih funkcija, potrebno je upoznati se s njihovom sintaksom otvaranjem informacijske kartice za uređaj `?jpg`.

Generalno, proces ima tri faze:

- otvaranje željenog grafičkog uređaja s postavkama
- izrada drafa (base, lattice, ggplot grafika...)
- zatvaranje grafičkog uređaja

```
#otvaramo željeni medij - jpg
jpeg('moj_graf.jpg')

#crtamo na uređaj
plot()

#zatvaramo uređaj
dev.off()
```

Najvažnije funkcije unutar osnovnog sustava grafike su sljedeće:

- `plot()` - generička funkcija radi dijagram raspršenja (engl. *scatterplot*) ili druge vrste grafikona u ovisnosti o vrsti objekta na kojem je pokrenuta. To je grafička funkcija visoke razine čiji rezultat kasnije možemo, prema željama upotpunjavati



funkcijama niske razine kao što su primjerice:

- `lines()`\*\* već izrađenom grafikonu dodaje crte - ako je funkcija pokrenuta na dvodimenzionalnoj matrici ova funkcija jednostavno povezuje točke
- `points()` - dodaje točke na postojeći grafikon
- `text()` - dodaje oznake na zadane koordinate
- `title()` - dodaje naslove i podnaslove
- `mtext()` - dodaje proizvoljan tekst na margine grafa
- `axis()` - dodaje oznake na osima, itd.

U ovome dijelu ćemo se upoznati s najčešće korištenim grafičkim parametrima.

```
#pomoć oko parametara
#help(par)
#trenutne postavke grafičkih parametara
#koje su nam trenutne postavke grafičkih parametara - prvih 5 u listi
par() [1:5]
```

```
FALSE $xlog
FALSE [1] FALSE
FALSE
FALSE $ylog
FALSE [1] FALSE
FALSE
FALSE $adj
FALSE [1] 0.5
FALSE
FALSE $ann
FALSE [1] TRUE
FALSE
FALSE $ask
FALSE [1] FALSE
```

Često je korisno napraviti kopiju starih postavki kako bismo ih kasnije mogli pozvati bez zatvaranja sesije te na taj način povratka na predodređene vrijednosti.

```
opar <- par()

# definiramo željenu boju za ispis labels na x i y osi
# ova promjena odnosit će se na sve grafičke prikaze dok postavke ne promijenimo
par("col.main"="darkred")

par("col.main")
```

```
FALSE [1] "darkred"
```

Ukoliko želimo dobiti informaciju informaciju o specifičnom grafičkom parametru njegov naziv stavimo kao argument funkcije `par()`.

```
#trenutna postavka vrijednost parametra, tip linije i tip crtanja točaka
par("lty")
```

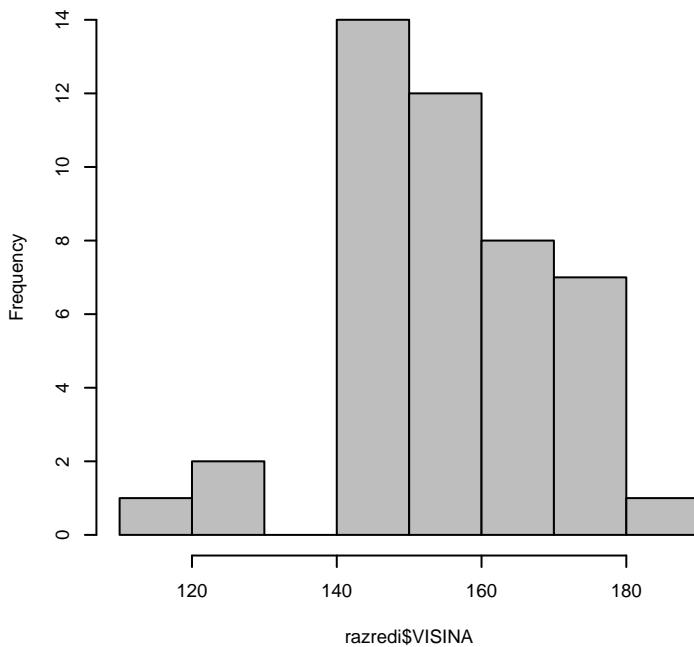
```
[1] "solid"
par("pch")
```

```
[1] 1
```

Postojeći grafikon možemo doraditi funkcijama niske razine u zasebnoj naradbi. U primjeru koji slijedi grafički parametar će biti promijenjen samo za izradu ovog grafikona i neće se primjenjivati na ostale grafikone izrađene u sesiji kao što je slučaj kada grafičke postavke mijenjemo putem funkcije `par()`.

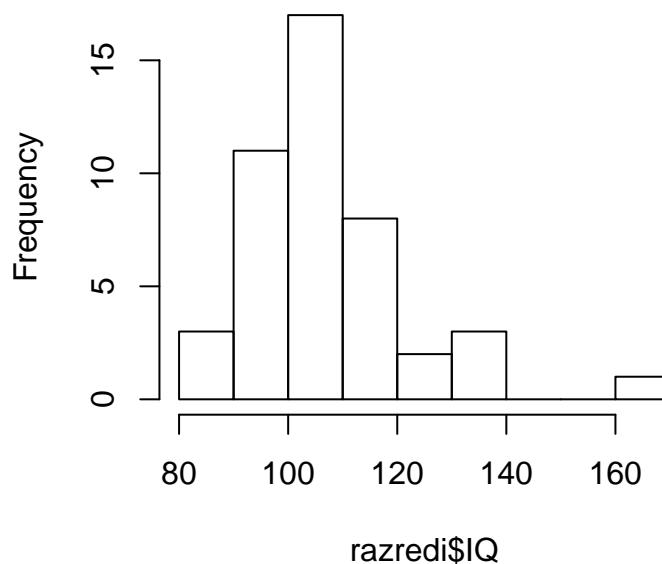


**Histogramski prikaz visina,  
nove postavke koje vrijede za sve grafikone do opoziva**



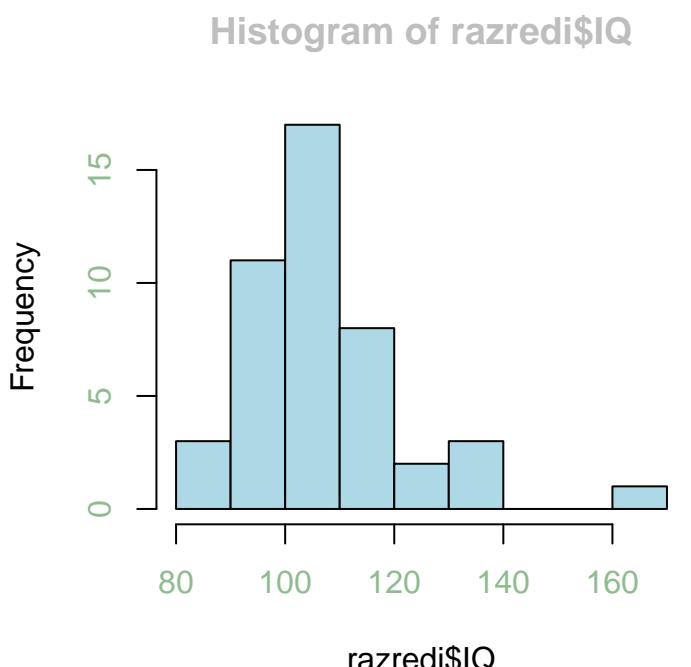
Slika 6: Grafički prikaz s promijenjenim postavkama funkcijom par()

**Histogramski prikaz IQ, stare postavke**



Slika 7: Grafički prikaz s vraćenim starim postavkama





Slika 8: Crtanje s promijenjenim postavkama grafičkih parametara unutar grafičke funkcije visoke razine

```
# promjena grafičkog parametra u pozivu funkcije, funkcija
hist(razredi$IQ,
  col.axis="darkseagreen",
  col.main="gray",
  col="lightblue",
  cex=2.5)
```

Na grafikama veoma često želimo kombinirati matematičke simbole i tekst. Slijedi primjer, slika 9, kombinacije proizvoljnog teksta i matematičkih oznaka kao labelama osi grafikona. Mnoge osobe koje rade analitiku podataka barem su u nekoj mjeri upoznate s jezikom *Tex* u kojem je pisanje kompleksih matematičkih izraza znatno jednostavnije i čije su mogućnosti u matematičkoj notaciji ogromne. Ukoliko želite neki od *Tex* izraza prebaciti u format R-a, izraz, engl. *expression* upoznajte se s paketom *latex2exp*.

```
#primjer na skupu podataka paketa DAAG
library(DAAG)
data(ais)

#nova funkcija expression
plot(hg ~ rcc, data=ais, pch=19, col="darkseagreen",
      xlab=expression("Eritrociti (* 10^12 * italic(l)^{-1} * ")),
      ylab=expression("Hemoglobin (* g*dot(" ")* daL^{-1} * ")"),
      main="Primjer kombiniranja teksta \ni matematičkih simbola")
```

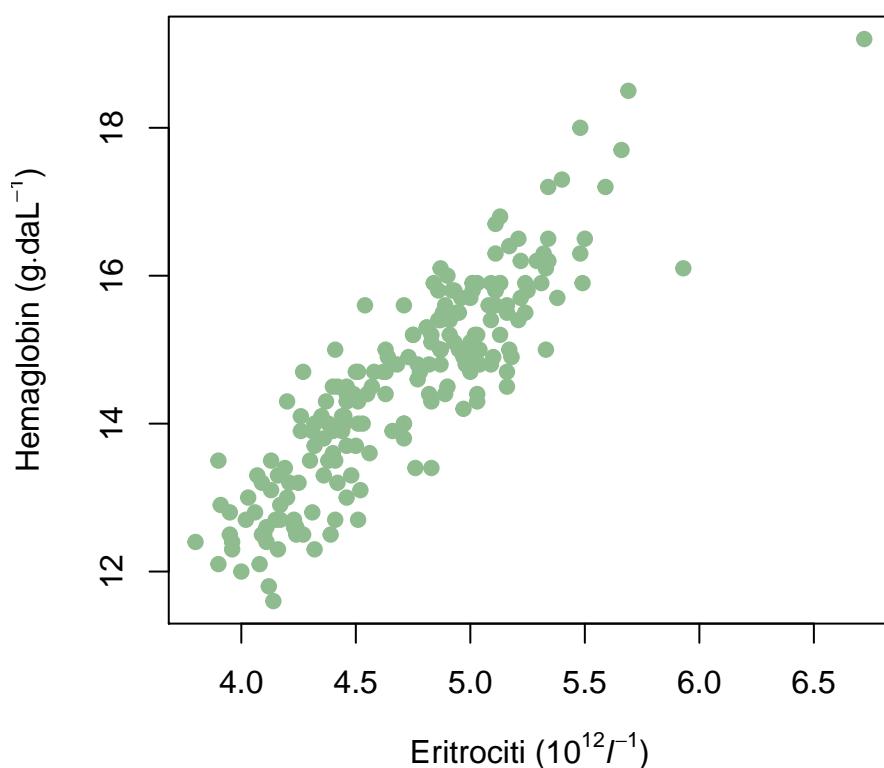
Za one koji žele znati kako se u sustavu koriste ostale matematičke notacije dodatne informacije mogu se pronaći na poveznici <http://vis.supstat.com/2013/04/mathematical-annotation-in-r/>.

Najčešće grafičke funkcije base grafičke: *plot()*, *boxplot()*, *barplot()*, *stars()*, *pairs()*, *hist()*, *image()*, *contour()*. S nekim prikazima ćemo se upoznati kasnije u tečaju prilikom upoznavanja s vizualizacijom distribucije, raspodjele, jedne slučajne varijable.

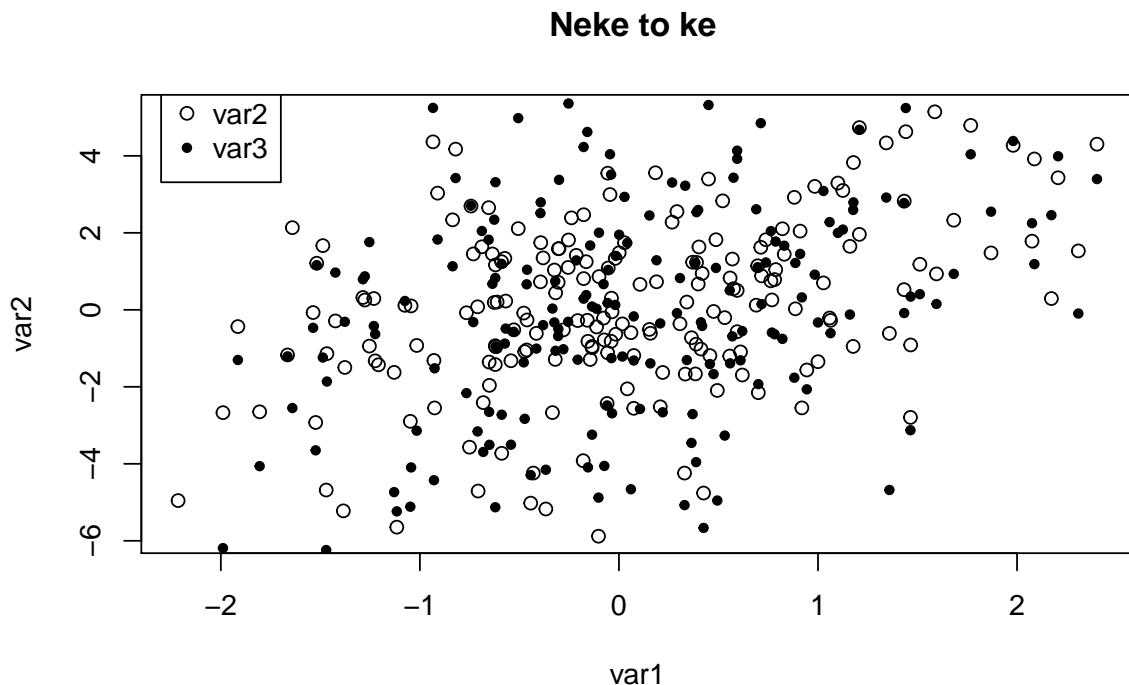
Funkcija *plot()* je najosnovnija grafička funkcija. To je generička funkcija i izgled grafičke koju daje ovisi o tipu argumenta zadanih funkciji.



### Primjer kombiniranja teksta i matematičkih simbola



Slika 9: Kombinacija teksta i matematičkih simbola



Slika 10: Crtanje dijagrama raspršenja

```
#podaci
set.seed(1) #reproducibilan primjer
var1 <- rnorm(200)
var2 <- var1 + 2 * rnorm(200)

#crtamo funkcijom plot dviye kontinuirane varijable
#dijagram raspršenja engl. scatterplot

plot(var1, var2)
var3 <- var2 + 2 * rnorm(100)

#nadodajemo točke na postojeći graf, funkcija points()
points(var1, var3, pch = 20)

#nadodajemo legendu na postojeći graf, funkcija legend
legend(-2.3, 6, c("var2", "var3"), pch= c(1, 20))

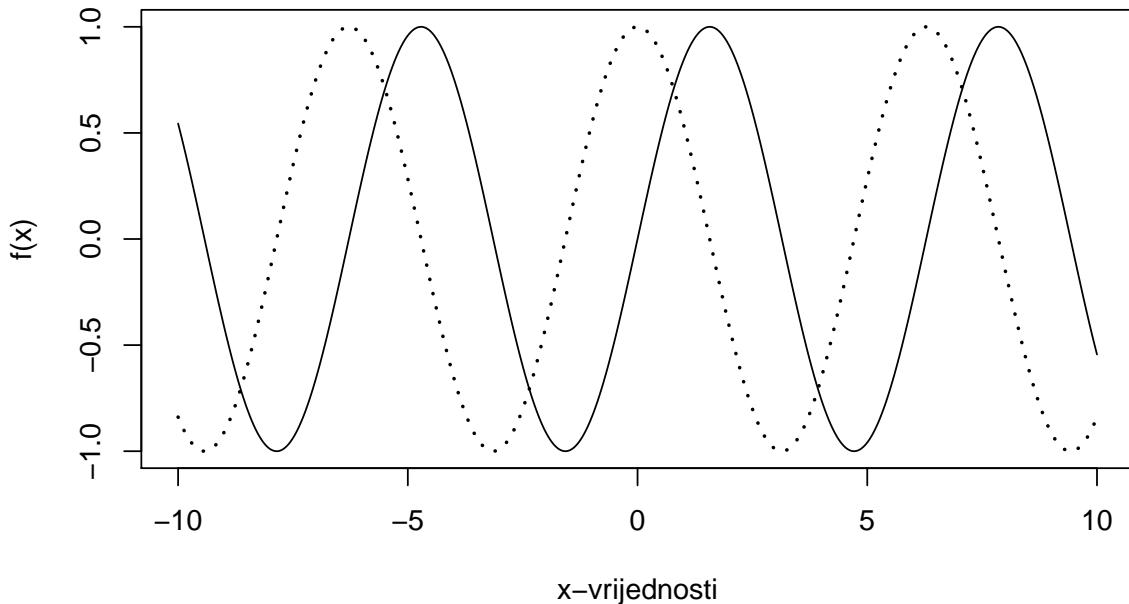
#natodajemo naslov
title("Neke točke")

#podaci
#funkcija plot - crtanje trigonometrijskih funkcija
x <- seq(-10, 10, length = 1000)

#crtanje, pozivamo funkciju visoke razine, stvaranje novog grafikona
plot(x,
      sin(x),
      xlab = "x-vrijednosti",
      ylab = "f(x)",
      type = "l")
```



## Trigonometrijske funkcije $\sin(x)$ i $\cos(x)$



Slika 11: Crtanje trigonometrijskih funkcija

```
#dorada grafikona funkcijama niske razine, dodavanje linija
lines(x,
      cos(x),
      lty = 3,
      lwd=2)
```

```
#dorada grafikona funkcijama niske razine, dodavanje naslova
title("Trigonometrijske funkcije sin(x) i cos(x)")
```

Unutar funkcije `plot()` moguće je definirati točke za koje će funkcija izračunati gustoće - pozivanje `densCols()` funkcije iz paketa `grDevices`. U sljedećem primjeru unutar funkcije `plot()` moguće je dodati faktor prozirnosti putem parametra `col` - četvrti broj, prva tri `rgb` boja.

```
plot(podaci$X,
      podaci$Y,
      col=rgb(0.3,0.5,0.3,0.3),
      pch=19,
      main="Gustoća točaka prozirno")

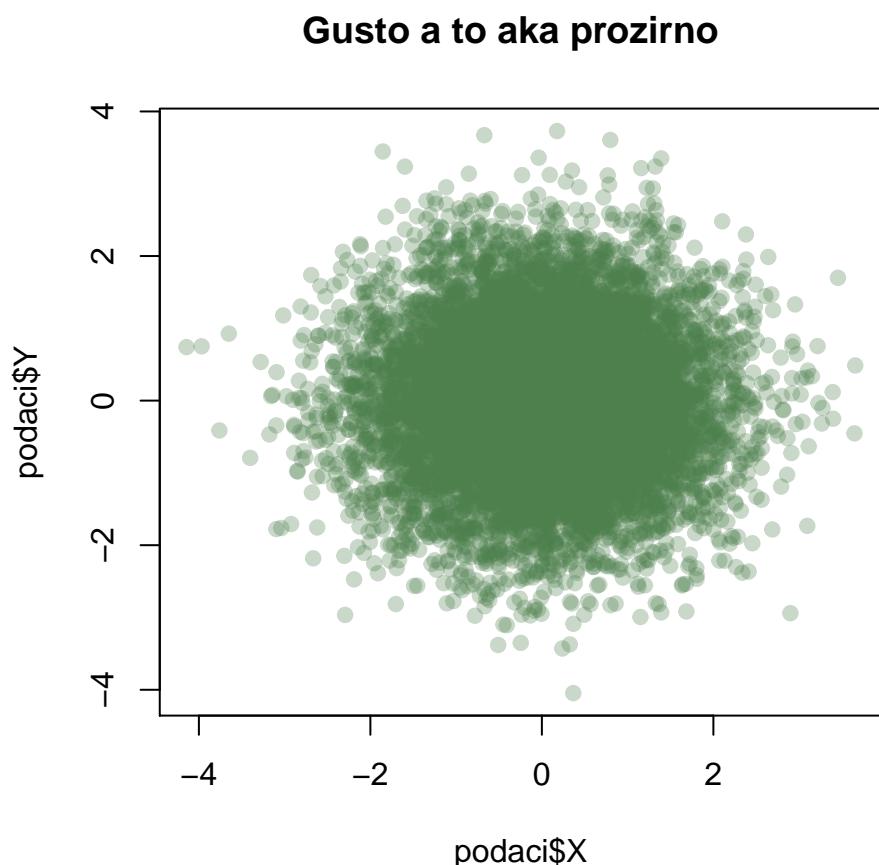
str(razredi)

'data.frame': 45 obs. of 4 variables:
 $ IQ      : num 107 100 101 110 99 111 110 108 104 102 ...
 $ RAZRED: Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
 $ VISINA: num 170 168 149 156 176 ...
 $ SPOL   : Factor w/ 2 levels "M","Ž": 2 1 2 1 1 1 2 2 2 2 ...
```

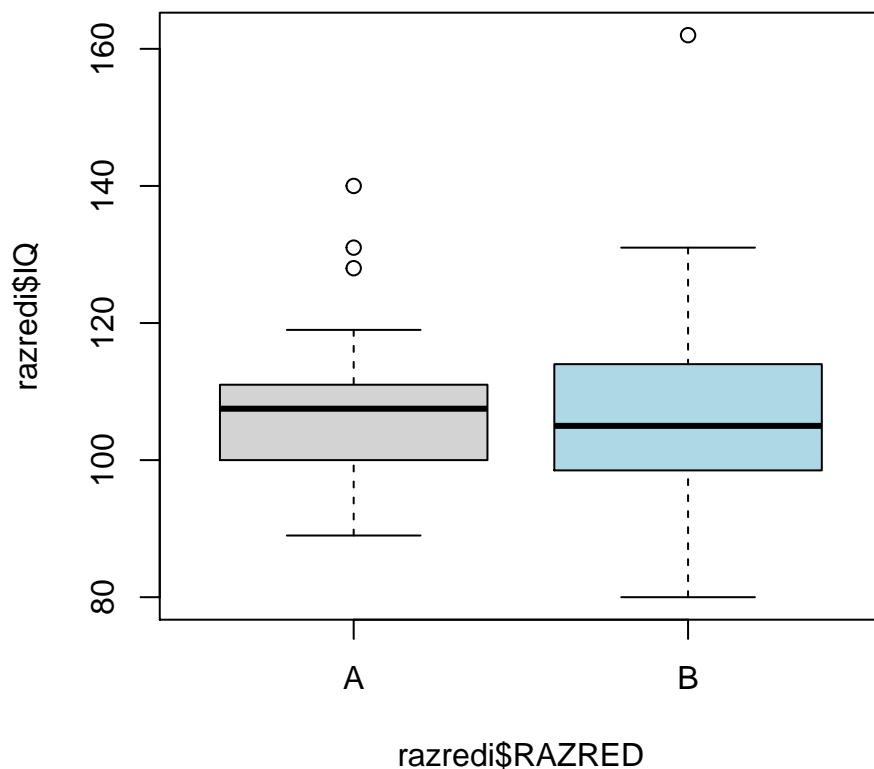
Kada funkciji `plot()` kao argumente zadamo jednu kontinuiranu i jednu kategoriju (factor) varijablu, dobijemo box-plot prikaz. Ukoliko želimo boxplot za samo jednu varijablu, bez razdjeljivanja:

Generička funkcija `plot()` koja daje boxplot prikaz ukoliko smo kao argumente funkciji dali jednu kontinuiranu i jednu kategoriju varijablu.





Slika 12: Primjer izrade dijagrama raspršenja s velikim brojem točaka, prilagodna prozirnosti boje



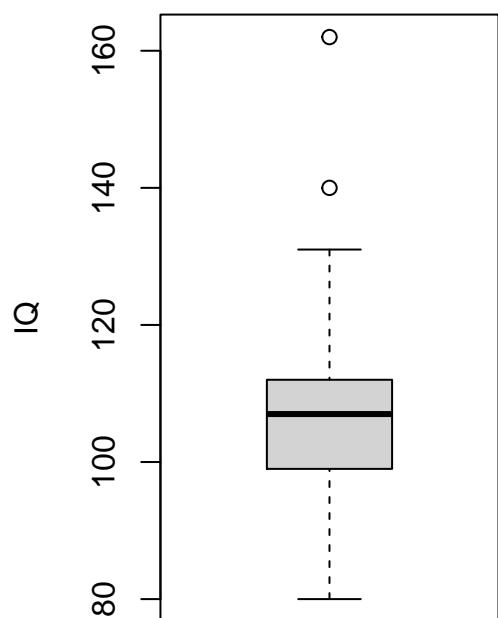
Slika 13: Generička funkcija plot, boxplot

```
plot(razredi$IQ ~ razredi$RAZRED,
      main="Box-plot prikaz napravljen funkcijom plot()",  
      col=c("lightgray", "lightblue"))
```

Ipak, za Box-Whisker prikaz postoji i specijalizirana funkcija:

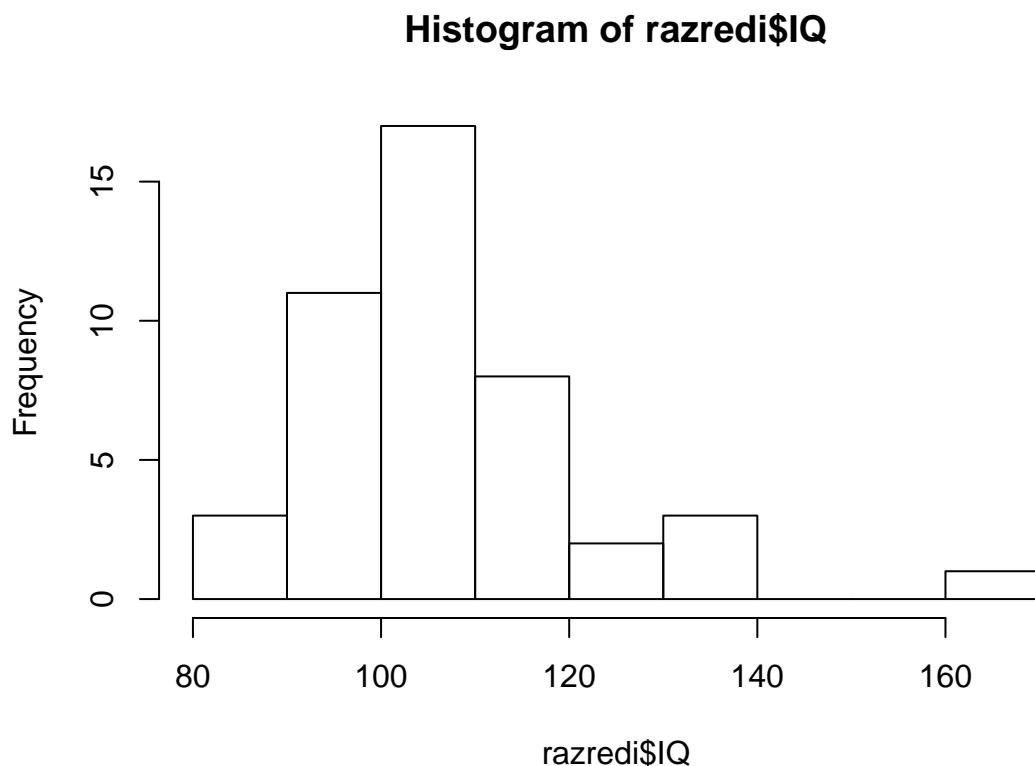
```
boxplot(razredi$IQ,  
        col="lightgray",  
        main="Box-plot prikaz varijable IQ",  
        xlab="RAZRED",  
        ylab="IQ")
```

### Box–plot prikaz varijable IQ



RAZRED

Slika 14: Generička funkcija plot, jedinstvena varijabla



Slika 15: Histogramski prikaz, predodređene postavke

Funkcija `hist()` prikazuje histogramski prikaz kontinuiranog skupa podataka:

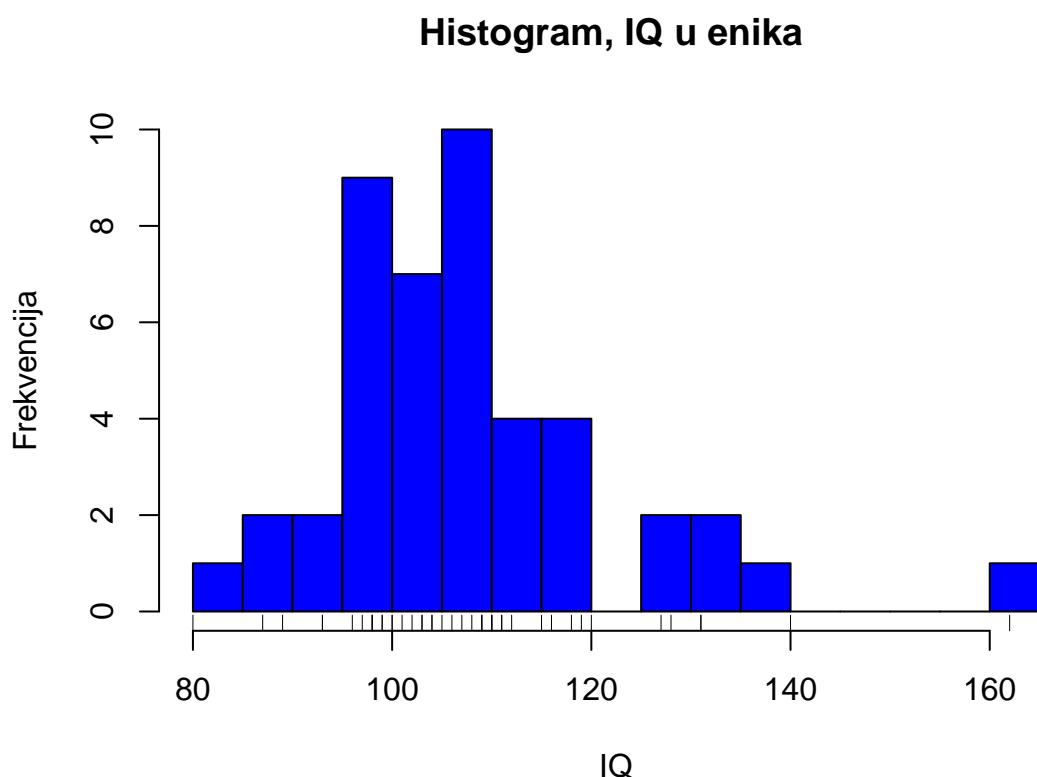
```
hist(razredi$IQ) #osnovni graf
```

Promjena grafičkih parametara unutar funkcije visoke razine nadopuna grafikona funkcijom niske razine, `main()`.

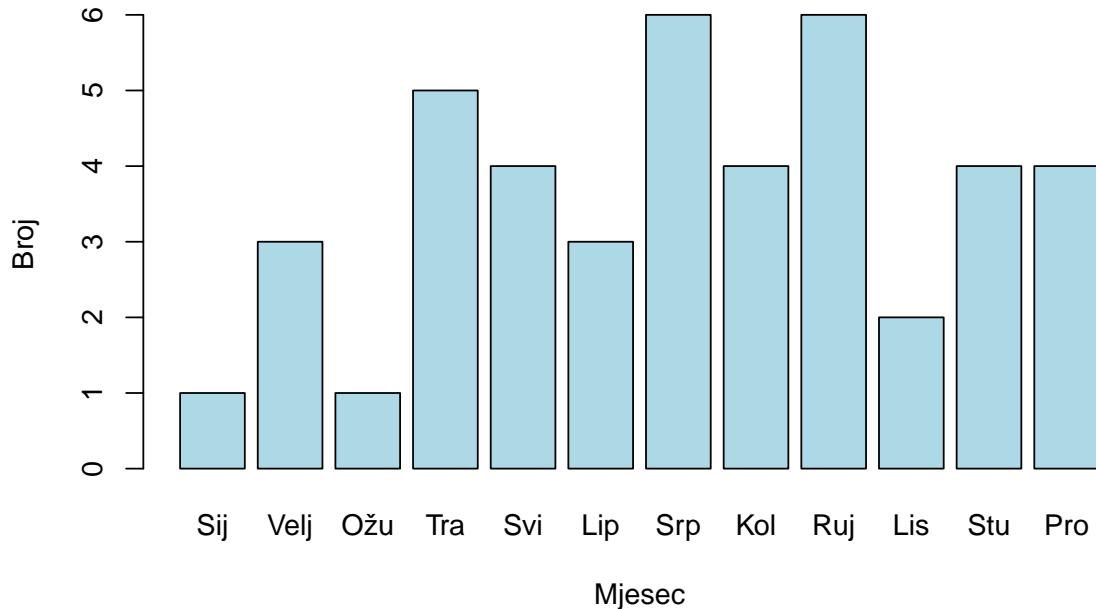
```
hist(razredi$IQ,
  br=12,
  col="blue",
  xlab="IQ",
  ylab="Frekvencija",
  freq=TRUE,
  main="Histogram, IQ učenika")
```

```
#nova funkcija rug nadodaje vrijednosti na graf
#funkcija niske razine
rug(razredi$IQ)
```





Slika 16: Histogramski prikaz, promjena grafičkih postavki koje vrijede samo na ovom crtežu s dodavanjem naslova



Slika 17: Primjer barplot prikaza za diskrete podatke

Za vizualizaciju kategoriskih podataka prikladan je grafički prikaz stupičasti dijagram, funkcija *barchart()*.

```
#barplot, diskretni podaci
#upoznajmo se s funkcijom
?barplot
```

```
#skup diskretnih vrijednosti
x <- c(3,2,6,8,7,7,5,9,4,2,10,6,4,12,11,
      12, 11, 5,5,9,11, 12, 12,4,4,4,10,
      7,8,8,6,9,2,5,7,9,1,11,9,7,7,8,9)
```

Prije vizualizacije kategoriskih podataka potrebno je izračunati frekvencije pojedine vrijednosti.

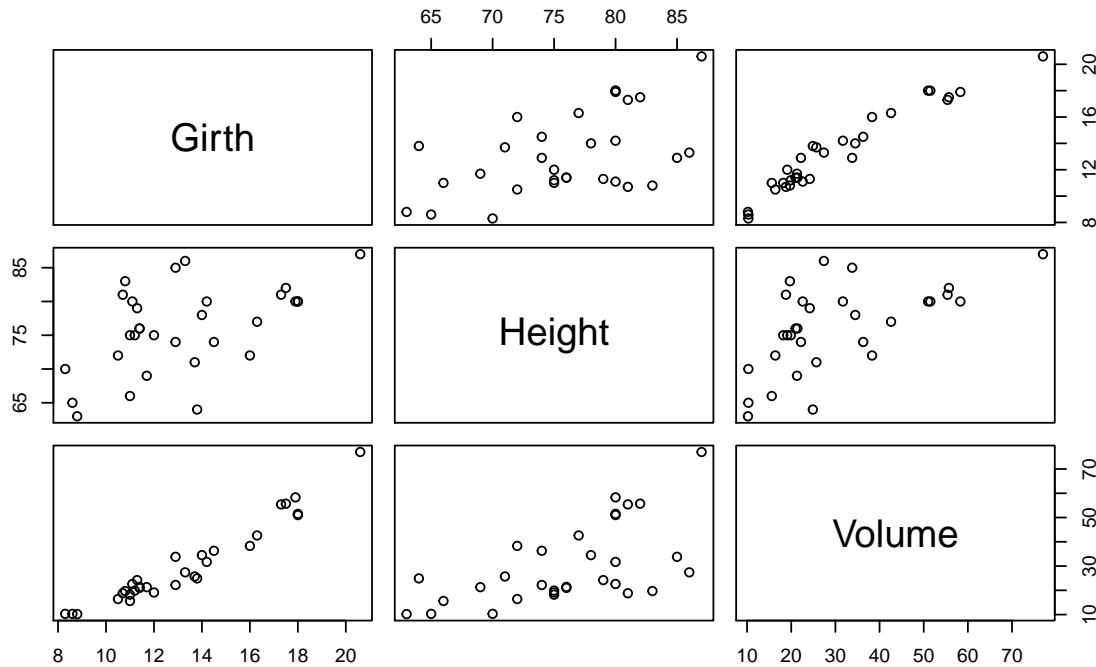
```
#nova funkcija table
?table

x_crtam <- table(x)
```

Ili s većim brojem parametara:

```
#crtamo diskrete
barplot(x_crtam, border="black",
        col ="lightblue",
        names.arg=c("Sij","Velj","Ožu","Tra",
                  "Svi", "Lip", "Srp", "Kol", "Ruj", "Lis", "Stu", "Pro"),
        angle=90,
        xlab="Mjesec",
        ylab="Broj") #kraj slike
```





Slika 18: Primjer funkcije pairs

U nastavku slijede primjer vizualizacije parova varijabli korištenjem funkcije `pairs()`. Funkcija daje grafički prikaz dvije po dvije varijable, kombinacije svih parova varijabli u skupu podataka. Demonstrirat ćemo funkciju na skupu podataka koji dolazi instalacijom sustava R, `trees`.

```
#skup podataka trees
data(trees)

#prvih nekoliko podataka
head(trees)
```

	Girth	Height	Volume
1	8.3	70	10.3
2	8.6	65	10.3
3	8.8	63	10.2
4	10.5	72	16.4
5	10.7	81	18.8
6	10.8	83	19.7

```
#funkcija pairs koja vizualizira veze parova varijabli iz objekta
pairs(trees)
```

##Boje u R-u

Pravilnom uporabom boja moguće je u znatnoj mjeri predočiti vezu unutar podataka koje crtamo. Sustav boja unutar sustava R je izuzetno dobro razvijen. U okviru ovog tečaja dat ćemo samo kratak uvod u način dodjeljivanja boja podacima te izboru i definiranju odgovarajućih paleta u ovisnosti o vrsti podataka za koje radimo vizualizaciju. Boje koje su predodređene u sustavu R nisu zadovoljavale korisnike pri izradi raznolikih, kvalitetnih prikaza podataka. Zbog toga, u posljednjih se desetak godina sustavno radio na razvoju, upravljanju i specifikaciji boja na grafikonima. Danas postoji niz kontribuiranih paketa koji rad s bojama čine vrlo jednostavnim i praktičnim.

Standardan način uporabe boja u sustavu R, prilikom poziva grafa radi se na način da sustav najprije koristi prvu boju u standardnoj paleti boja, zatim nastavlja s drugom bojom u standardnom nizu i tako dalje, koliko je boja potrebno u određenoj grafici. Prva boja u nizu, boja 1, u sustavu R je crna (`col=1`), boja 2 je crvena, boja 3 je zelena itd.



Često u pripremi vlastite grafike sami želimo odrediti paletu boja kojom ćemo prikazati naše podatke. Sustav R u ovom je segmentu izuzetno fleksibilan i dozvoljava veliku kreativnost. Za vizualizaciju različitih tipova podataka potrebne su i različite palete, generalno tako postoje tri vrste paleta:

- sekvenčne za kontinuirane varijable
- divergentne za varijable koje divergiraju nekoj vrijednosti ili imaju raspon od negativnih prema pozitivnim vrijednostima te
- kvalitativne za varijable koje su mjerene na nominalnoj skali.

Upoznat ćemo se s ograničenim brojem funkcija paketa *grDevices*. Funkcija *colours()* ili *colors()* daje nam vektor imenovanih boja unutar sustava R.

```
#prije provođenja - upoznajmo se sa svakom funkcijom
?colors
```

Ukoliko želimo imena svih imenovanih boja u sustavu:

```
colors()[1:5] #ispis samo za prvih pet
```

```
[1] "white"      "aliceblue"    "antiquewhite" "antiquewhite1"
[5] "antiquewhite2"
```

Kada želimo vidjeti koja je trenutno aktivna paleta na našem sustavu:

```
palette()
```

```
[1] "black"     "red"       "green3"    "blue"      "cyan"      "magenta"  "yellow"
[8] "gray"
```

Istom funkcijom s danim argumentima možemo definirati novu paletu sustava:

```
palette(rainbow(6))
```

Povratak postavki na prvočitnu paletu:

```
palette("default")
palette()
```

```
[1] "black"     "red"       "green3"    "blue"      "cyan"      "magenta"  "yellow"
[8] "gray"
```

Izrađujemo paletu od 16 sivih nijansi :

```
#interpolacija između graničnih vrijednosti
gray(0:15/15)
```

```
par(mfrow=c(1,2), mar=c(0,0,0,0))
#vizualizacija paleta
#unaprijed pripremljene funkcije u sustavu
#terrain.colors - uzimamo 15 boja
#grafički prikaz pite - samo za vizualizaciju paleta!
pie(rep(1,15), col=terrain.colors(15))
pie(rep(1,15), col=gray(0:15/15))
```

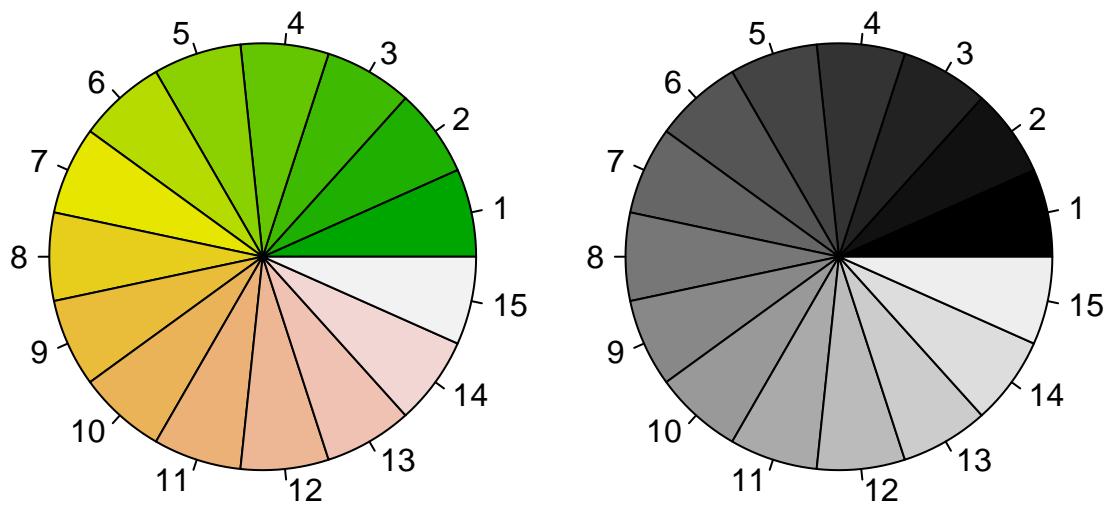
U nastavku ćemo se upoznati s najčešćim načinima izrade vlastitih shemama boja, vlastitih paleta unutar sustava R.

```
#želimo izraditi svoju paletu boja
#radimo paletu između graničnih boja crvene (neg. "red") i plave (eng. "blue")
#preciznije - radimo funkciju koja će računati interpolaciju
#na traženim lokacijama unutar zadanih rubnih boja
pal <- colorRamp(c("red", "blue"))

#rezultat funkcije colorRamp je funkcija koju smo proizvoljno nazvali pal
pal
```

```
function (x)
```





Slika 19: Vizualizacija palete `terrain.colors()` sustava R te korisnički napravljene palete sivih boja s 15, obje palete od 15 nijansi

```
roundcolor(cbind(palette[[1L]](x), palette[[2L]](x), palette[[3L]](x),
  if (alpha) palette[[4L]](x))) * 255
<bytecode: 0x0000000051f1c4f8>
<environment: 0x0000000051f1dde0>
```

Pogledajmo vrijednosti paleta na njezinim rubovima:

```
#vrijednost funkcije na početku
#maksimum od 255 na crvenoj komponenti, ostalo 0 =crvena
pal(0)
```

```
[,1] [,2] [,3]
[1,] 255    0    0
#vrijednost funkcije na kraju raspona
##kolone predstavljaju RGB komponente boje
pal(1)
```

```
[,1] [,2] [,3]
[1,] 0    0  255
#vrijednost funkcije (boja) na 1/10 razdaljine između granica
pal(0.1)
```

```
[,1] [,2] [,3]
[1,] 230    0 25.5
#ukoliko želimo npr. paletu od 15 boja zatražit ćemo sljedeće:
#izradimo sekvencu vrijednosti na kojima želimo izračunati vrijednost boje
col <- pal(seq(0,1, len=15))

#ispisemo dobiveni objekt, pogledamo njegovu strukturu
col
```

```
[,1] [,2] [,3]
[1,] 255.0  0   0.0
```



```
[2,] 236.8   0 18.2
[3,] 218.6   0 36.4
[4,] 200.4   0 54.6
[5,] 182.1   0 72.9
[6,] 163.9   0 91.1
[7,] 145.7   0 109.3
[8,] 127.5   0 127.5
[9,] 109.3   0 145.7
[10,] 91.1    0 163.9
[11,] 72.9    0 182.1
[12,] 54.6    0 200.4
[13,] 36.4    0 218.6
[14,] 18.2    0 236.8
[15,] 0.0     0 255.0
```

Veoma važna i korisna funkcija `colorRampPalette()` koja kao rezultat daje novu funkciju kojom ćemo interpolirati boje između zadanih graničnih vrijednosti.

```
pal2 <- colorRampPalette (c("red", "yellow"))
```

Sada s pripremljenom funkcijom `pal2` možemo tražiti interpolaciju želenog broja boja unutar zadanih granica. Na primjeru koji slijedi tražimo 15 boja koje se ravnomjerno nalaze između granica crvene i žute.

```
col2 <- pal2(15)
```

```
col2
```

```
[1] "#FF0000" "#FF1200" "#FF2400" "#FF3600" "#FF4800" "#FF5B00" "#FF6D00"
[8] "#FF7F00" "#FF9100" "#FFA300" "#FFB600" "#FFC800" "#FFDA00" "#FFEC00"
[15] "#FFFF00"
```

```
pal3 <- colorRampPalette (c("red4", "darkgray"))
col3 <- pal3(15)
```

```
pal4 <- colorRampPalette (c("darkseagreen", "lightblue"))
col4 <- pal4(15)
```

Iako ih nikako ne preporučamo kao prikaz podataka, strukturni dijagrami, popularne pite, veoma su korisne za vizualizacije pripremljenih paleta boja.

```
par(mfrow=c(1,3), mar=c(0,0,0,0))
#prva
pie(rep(1,15), col=col2)

#druga
pie(rep(1,15), col=col3)

#treća
pie(rep(1,15), col=col4)
```

Primjer vizualizacije numeričke matrice, koristimo skup podataka iz sustava R, `volcano` (<https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/volcano.html>). Koristit ćemo jedan od trenutno najpopularnijih paketa za izradu vlastitih paleta, paket `RColorBrewer`. Ukoliko nemate instaliran paket, prije učitavanja u radni prostor paket je potrebno instalirati s repozitorija CRAN. Najprije, pogledajmo podatke:

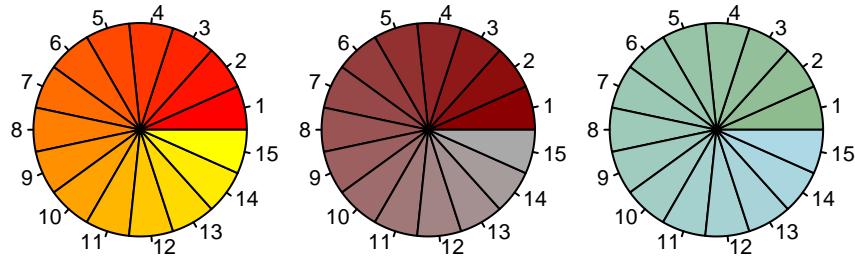
```
#pozivamo podatke
#u novijim verzijama R-a dovoljno samo napisati ime objekta
data(volcano)

#upoznajmo se s podacima
str(volcano)
```

---

```
num [1:87, 1:61] 100 101 102 103 104 105 105 106 107 108 ...
```





Slika 20: Vizualizacija paleta interpoliranih boja između zadanih granica

```
#provjerimo klasu objekta koji ćemo vizualizirati
class(volcano)
```

```
[1] "matrix"
```

Pogledat ćemo sve palete koje nam nudi paket *RColorBrewer*:

```
#instalacija s CRAN-a
#install.packages("RColorBrewer")
#učitamo funkcionalnosti paketa u radni prostor
library(RColorBrewer)
```

```
#prikaz postojećih paleta paketa
display.brewer.all()
```

Pripremamo željenu paletu:

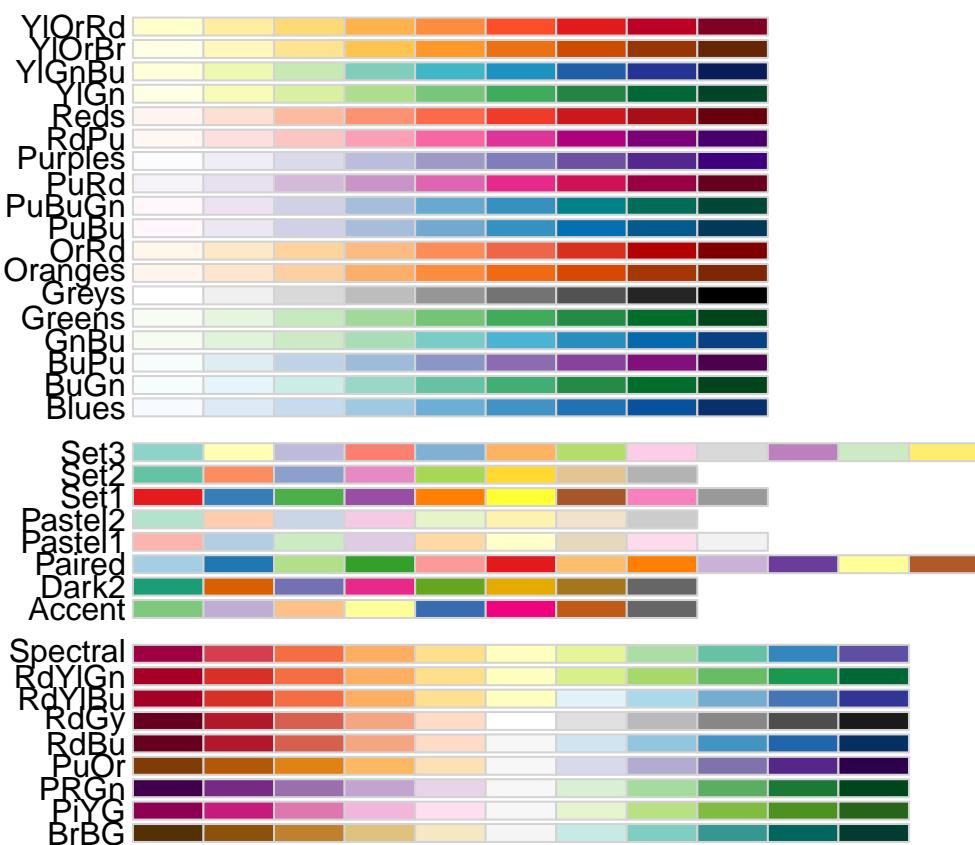
```
#biramo pet boja iz paleta Blues, plave boje
boje <- brewer.pal(5,"Blues")

#izrađujemo vlastitu paletu od 20 nijansi plavih boja
#koje se temelje na bojama izvučenim iz paleta Blues paketa RColorBrewer
boje20<-colorRampPalette(brewer.pal(5,"Blues"))(20)

#ili ovako
boje20<-colorRampPalette(boje)(20)

boje20
```





Slika 21: Palete paketa RColorBrewer



```
[1] "#EFF3FF" "#E4EDF9" "#D9E7F4" "#CFE1EF" "#C4DBEA" "#B8D4E6" "#A7CCE2"
[8] "#96C3DE" "#84BADB" "#73B2D7" "#64A9D3" "#58A0CE" "#4C96C8" "#408DC3"
[15] "#3484BE" "#2A7AB7" "#216FB0" "#1965A9" "#105BA2" "#08519C"
```

Vizualiziramo matricu s numeričkim vrijednostima, mjerjenjima nadmorske visine u na čeliji koristeći grafičku funkciju visoke razine `image()` te usporedite prikaz korištenjem inverzne palete, funkcija `rev()`.

```
par(mfrow=c(2,1))
#?image

#prvi prikaz
image(volcano, col=boje20,
      main="Primjer vizualizacije matrice, paleta od 20 boja")
#drugi prikaz
image(volcano,
      col=rev(boje20),
      main="Primjer vizualizacije matrice, inverzne boje")
```

Često prilikom vizualizacije multivarijatnih podataka želimo različite kategorije neke varijable obojati različitim bojama te na taj način unjeti dodatnu informaciju na grafički prikaz. U primjeru koji slijedi, slika 23, želimo svaku pojedinu jedinku, stablo, obojati drugom bojom kako bi bili prepoznatljivi na prikazu. Skup podataka koji koristimo, *Orange*, dolazi instalacijom sustava R, a sastoji se od kvantitativnih varijabli koje mjere starost i opseg drveta te identifikatora svakog pojedinog stabla, varijabla *Tree*.

```
data(Orange)
plot(Orange$circumference, Orange$age, col=Orange$Tree,
      pch=19, cex=1.3,
      main="Dijagram raspršenja s dodatnom \ninformacijom o nivoima faktora varijable Tree")
```

Palete koje napravimo ili standardne palete, u sustavu R slobodno koristimo u bilo kojem našem grafikonu. S većim brojem prikaza upoznat ćemo se kada ćemo govoriti o deskriptivnoj statistici u drugome dijelu tečaja.

#### Pitanja za ponavljanje

- 1) Poznavanjem svega što ste naučili odgovorite koja je boja pridruženo kojem stablu (*Orange\$Tree*)?
- 2) Na koji način su izabrane boje na prikazu?

## 3.2 Lattice (osnova paket *grid*)

*Lattice* sustav u potpunosti je neovisan neovisan od sustava osnovne grafike. Mnogi ga smatraju naprednjim i fleksibilnijim grafičkim sustavom od sustava base radi moguće vizualizacije više varijabli i njihovih zavisnosti istovremeno. Uključuje funkcije kao što su `xypplot()`, `levelplot()`. Grafički sustavi u jeziku R ne mogu se mijesati, iako napredniji sustavi u sebi nose ugrađene neke dijelove sustava base. Korisnici uglavnom započinju rad s osnovnom grafikom, a kasnije u ovisnosti o vrsti analiza i podataka kojima barataju koriste dijelove ostalih grafičkih sustava. Paket lattice čini osnovu ovog grafičkog sustava. Važan paket je paket iz osnovne instalacije sustava R, *grDevices*, koji u sebi nosi mogućnost izvoza grafike iz sustava R u velik broj grafičkih formata kao što su X11, PDF, PostScript, PNG i mnogi drugi. Sustav *lattice* radi na način da se svi detalji neke grafike daju u obliku jedne jedine naredbe. Sustav R automatski izračunava veličinu slova i slično, tamo gdje je potrebna prilagodba.

Osnovna struktura funkcije u grafici lattice ima ove argumente:

`_formula_ (npr. y ~ x | factor1*factor2)_`

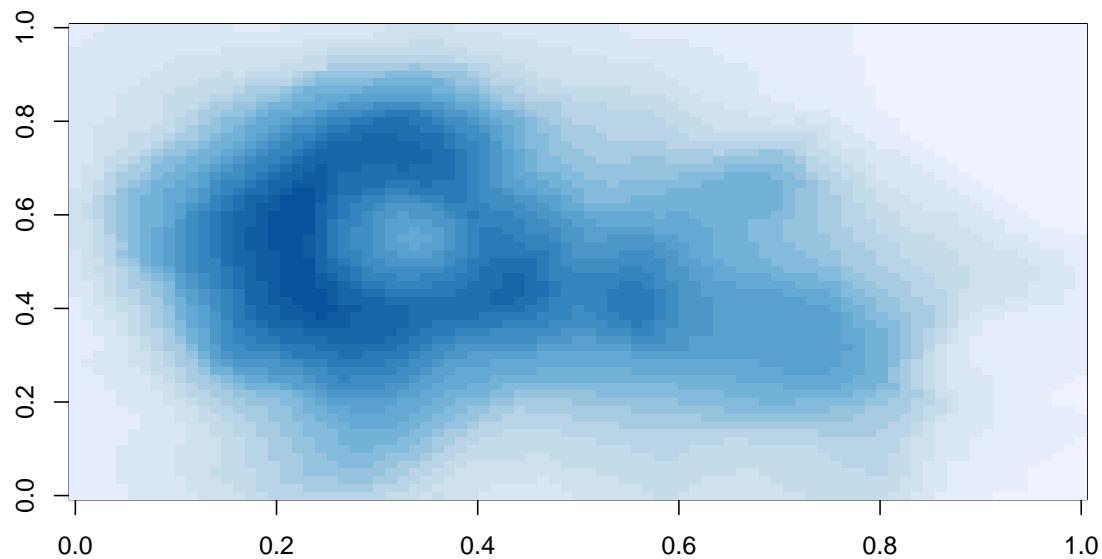
Grafika *lattice*, za razliku od *base* sustava, sve što smo zadali u naredbi pohranjuje u objekt klase *trellis* i u kasnije crta pripremljeni objekt. Znači, nakon izvršenja funkcije može se snimiti cijeli objekt, no ipak je bolje snimiti kôd koji je proizveo graf, jednako kao i u grafici base.

*Lattice* prikaz se sastoji od različitih elemenata koji su koordinirani različitim parametrima kako bi dali smislene rezultate. Ključni elementi *lattice* prikaza su:

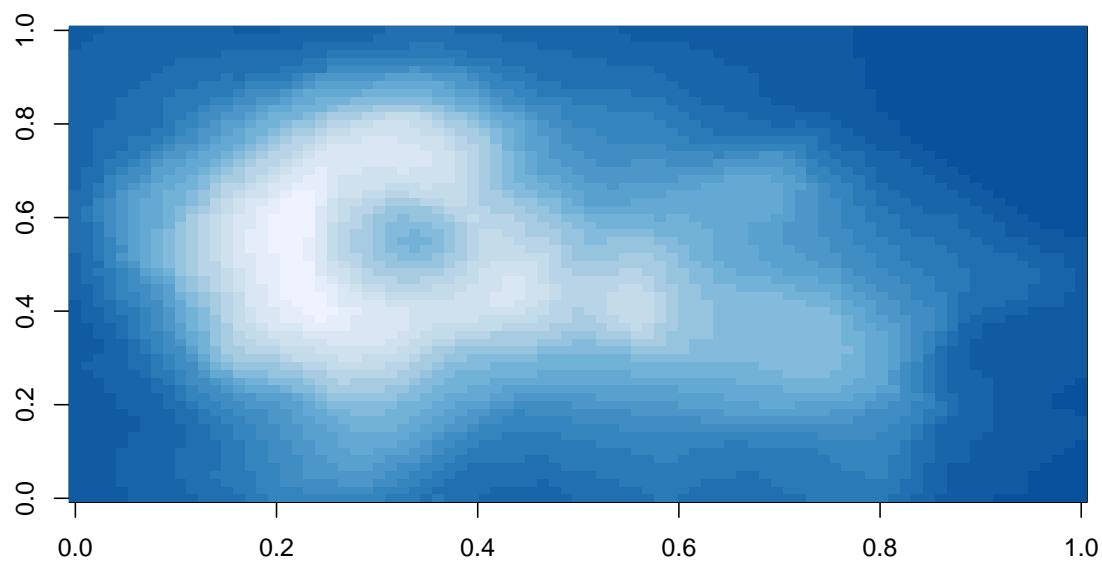
- **primarni prikaz** - panel
- **oznaka osi**



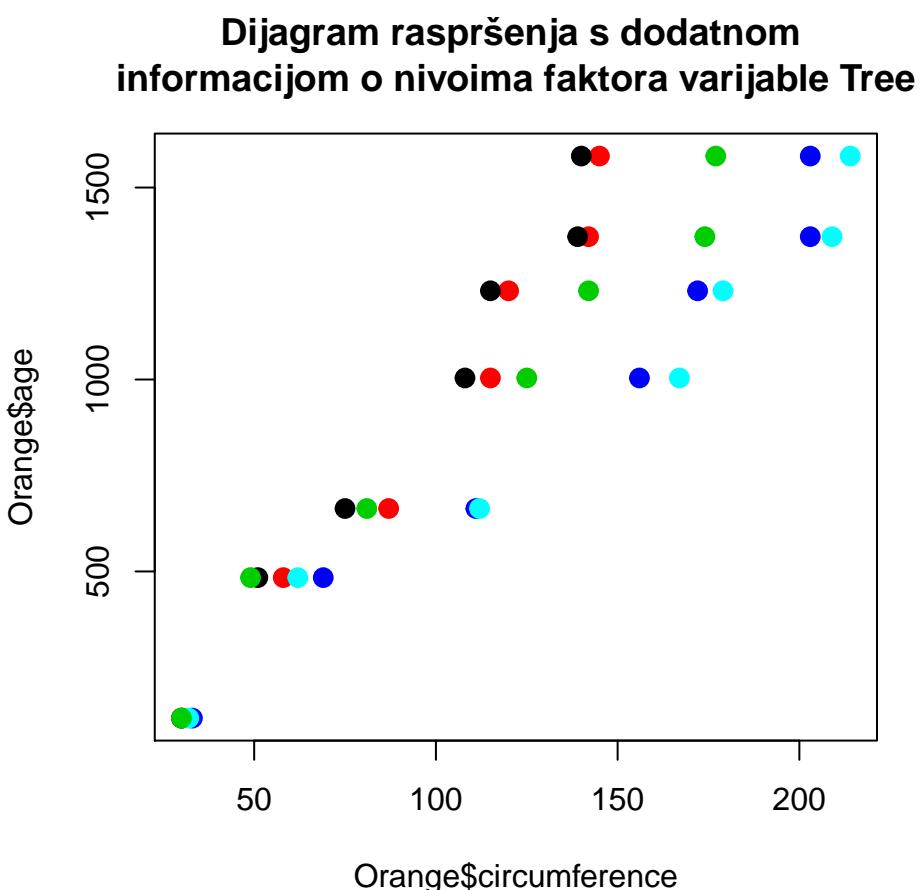
**Primjer vizualizacije matrice, paleta od 20 boja**



**Primjer vizualizacije matrice, inverzne boje**



Slika 22: Primjer vizualizacije matrice s a) prvočitnom i b) inverznom paletom



Slika 23: Primjer odavanja informacije o faktorskoj varijabli

- **oznaka trake** (opisuje proces uvjetovanja)
- **legende** (tipično opisuju proces grupiranja).

Korisnik može kontrolirati svaki element kao je to potrebno. Postoji mogućnost pružanja dodatnih argumenata pozivanjima funkcija visoke razine kako bi se aktivirale zadane vrijednosti ili putem kreiranja funkcija koje korisnik proizvoljno definira.

*Trellis* prikazi su definirani vrstom grafike i ulogom koju različite varijabe u njoj igraju. Svaka vrsta prikaza povezana je s odgovarajućom funkcijom visoke razine (histogram, dijagram gustoće itd.). Moguće uloge ovise o vrsti prikaza ali, su tipične:

- **primarne varijable**: one koje definiraju primarni prikaz
- **uvjetujuće varijable**: dijele podatke na podskupine, svaka od njih predstavlja različite panele
- **grupirajuće varijable**: podskupine se porede unutar panela tako da se prikazuju u odgovarajućim prikazima.

Generalan izgled funkcije u *lattice*-u:

```
graph_type(formula, data=)
```

gdje je `graph_type` odabran iz prethodno navedenih. Formula precizira varijablu(e) koje će se prikazati i bilo koje uvjetujuće varijable. Na primjer:

- `~x|A` - prikaz numeričke varijable x za svaku razinu faktora A
- `_y~x | A*B` - prikaz odnosa između numeričkih varijabli y i x odvojeno za svaku kombinaciju razine faktora A i B
- `~x` znači prikaz same numeričke varijable x.

Nacrtat ćemo pripremljeni skup podataka ali se najprije s podacima moramo upoznati:

```
str(OrchardSprays)
```

```
'data.frame': 64 obs. of 4 variables:
 $ decrease : num 57 95 8 69 92 90 15 2 84 6 ...
 $ rowpos   : num 1 2 3 4 5 6 7 8 1 2 ...
 $ colpos   : num 1 1 1 1 1 1 1 1 2 2 ...
 $ treatment: Factor w/ 8 levels "A","B","C","D",...: 4 5 2 8 7 6 3 1 3 2 ...
```

```
summary(OrchardSprays$treatment)
```

```
A B C D E F G H
8 8 8 8 8 8 8 8
```

Pogledajmo prije crtanja koliko nivoa ima varijabla `treatment`, funkcija `nlevels()` te koji su to nivoi funkcija `levels()` na faktoraskoj varijabli:

```
nlevels(OrchardSprays$treatment)
```

```
[1] 8
```

```
#eksplicitan upit o nivoima faktora
levels(OrchardSprays$treatment)
```

```
[1] "A" "B" "C" "D" "E" "F" "G" "H"
```

Najopćenitija grafička funkcija visoke razine iz paketa *lattice* je funkcija `xypplot()` kojom možemo napraviti velik broj prikaza:

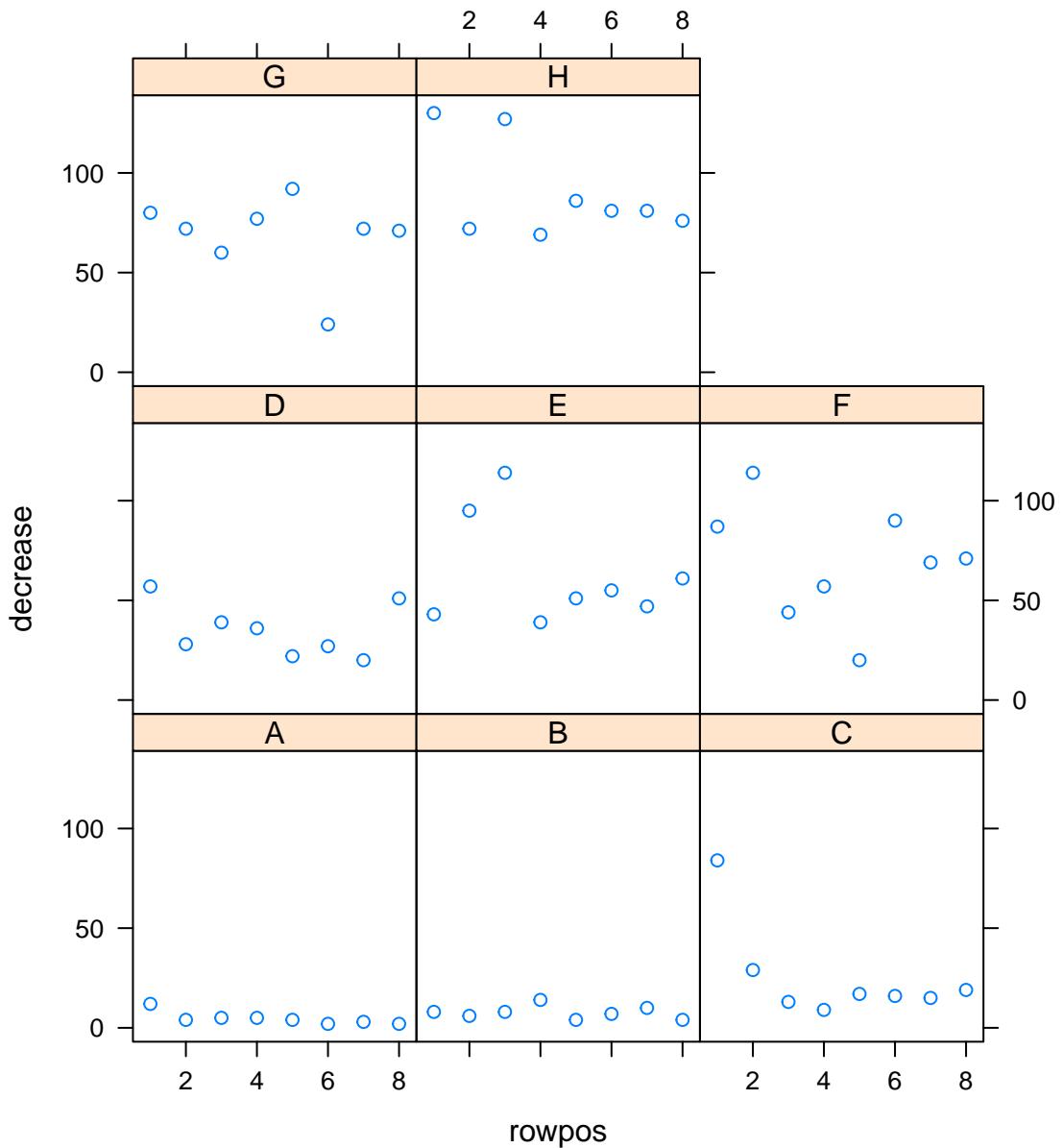
```
xypplot(decrease ~ rowpos | treatment, OrchardSprays,
        main="Primjer podjele multivarijatnih podataka \n korištenjem grafičkih mogućnosti paketa lattice",
        cex.main=0.8)
```

Malo prilagodbe grafikona:

```
library(lattice)
xypplot(decrease ~ rowpos | treatment, OrchardSprays,
        panel = function(x,y,...){
          panel.xyplot(x,y,...)
          #panel.abline(h=median(y), lty=3)
```



### Primjer podjele multivarijatnih podataka korištenjem grafi kih mogunosti paketa lattice



Slika 24: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable

```
panel.lmline(x,y, col=1)
})
```

Kao argumente funkcije `panel()` definiramo jedino one varijable koje želimo crtati na panelima i dodajemo dodatne uvjete. Za mogućnosti unutar ove, jedne od najvažnijih funkcija u paketu `lattice`, potrebno je u konzolu unijeti sljedeći upit `?xyplot`. Općenito, ukoliko želimo specificirati određenu funkciju (npr. `xyplot()`) iz paketa `lattice` potrebno je unijeti sljedeće: `lattice::xyplot`.

Ukratko, često korištene funkcije u paketu `lattice` mogu se podijeliti prema vrsti podataka i to na način da imamo funkcije za vizualizaciju:

- jedne varijable: `histogram()`, `densityplot()`, `bwplot()`, `stripplot()`
- kvantili za jednu ili više varijabli: `qqmath()`, `qq()`
- grafika za imenovane podatke: `barchart()`, `dotplot()`
- vizualizacija dvodimenzionalnih varijabli: `xyplot()` te mnoge druge.

Kroz nekoliko primjera ćemo vidjeti prednosti `lattice` grafike.

```
#Učitavanje ranije pripremljenog R objekta
load("podaci_lattice_grafika.Rdata")
```

Histogrami

```
#histogram cjelokupne varijable
histogram(~urod, podaci_lattice_grafika, main="Varijabla urod ukupno")

#razdijeljeno na panele prema nivoima varijable varijetet
histogram(~urod | varijetet, podaci_lattice_grafika)
```

Box-Whisker graf

```
bwplot(ploha ~ urod, podaci_lattice_grafika)

bwplot(ploha ~ urod | godina, podaci_lattice_grafika)
```

Gustoće

Na jednak način funkcioniraju i drugi tipovi prikaza varijabli, u ovome dijelu neparametarsko zaglađivanje (engl. *Kernel Density Plots*). Nakon što ste se upoznali sa sintaksom funkcije te posebno argumenta `plot.points` isprobajte opciju `TRUE` te prokomentirajte razliku u prikazu.

```
densityplot(~urod, podaci_lattice_grafika, plot.points = FALSE)
```

Pokušajte reći na koji biste način u `base` grafici napravili ovakav prikaz *Kernel Density*, neparametarsko zaglađivanje podataka?

Stupčasti dijagram

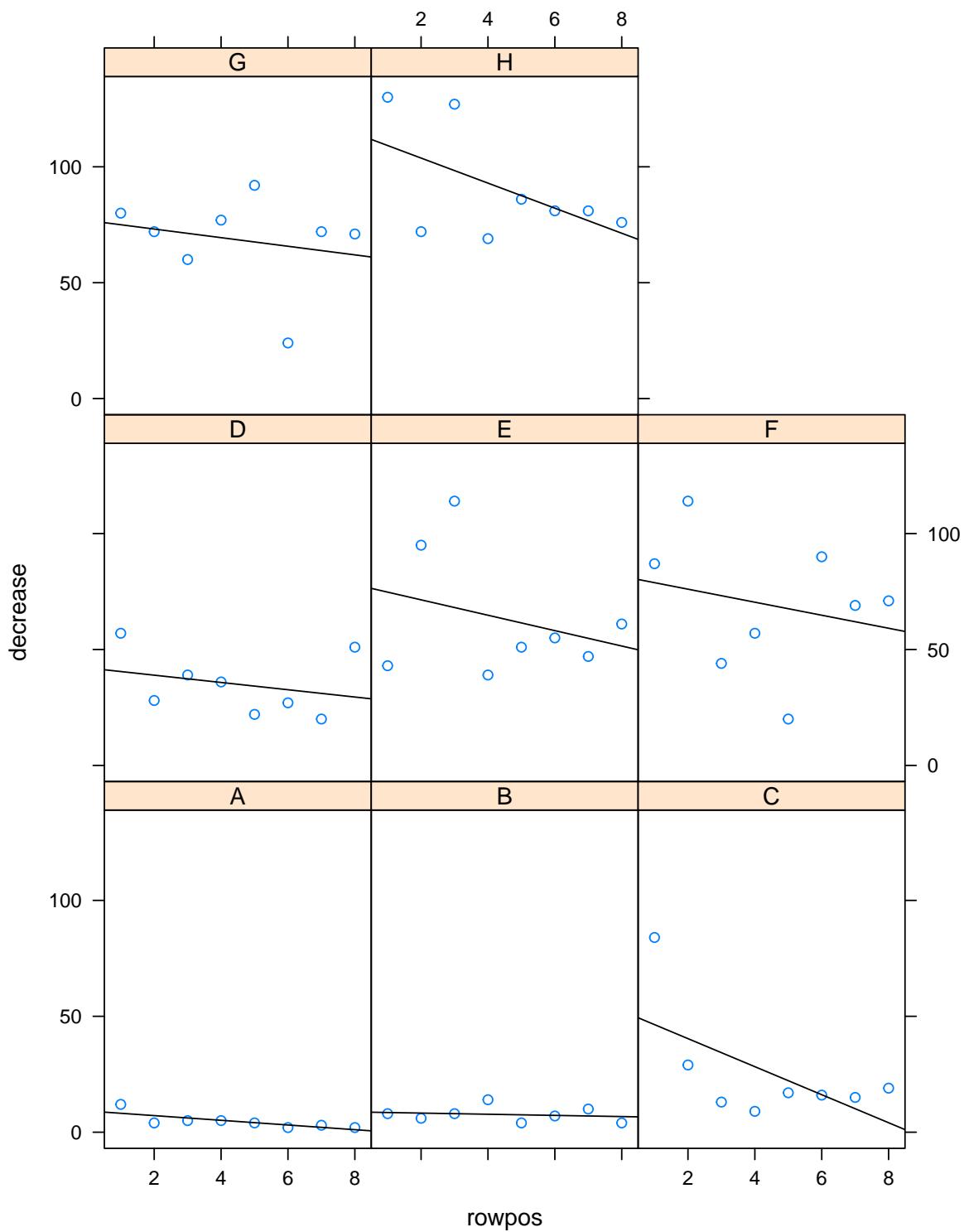
Na primjeru stupčastih dijagrama pokazat ćemo podatke razdijeljene po plohama dok će varijeteti biti različito obojani. Koji argument kontrolira obojanost prema varijabli `varijetet`?

```
barchart(urod ~ varijetet | ploha, data = podaci_lattice_grafika,
          groups = godina, layout = c(1,6), stack = TRUE,
          auto.key = list(space = "right"),
          ylab = "Urod u tonama",
          scales = list(x = list(rot = 90)))
```

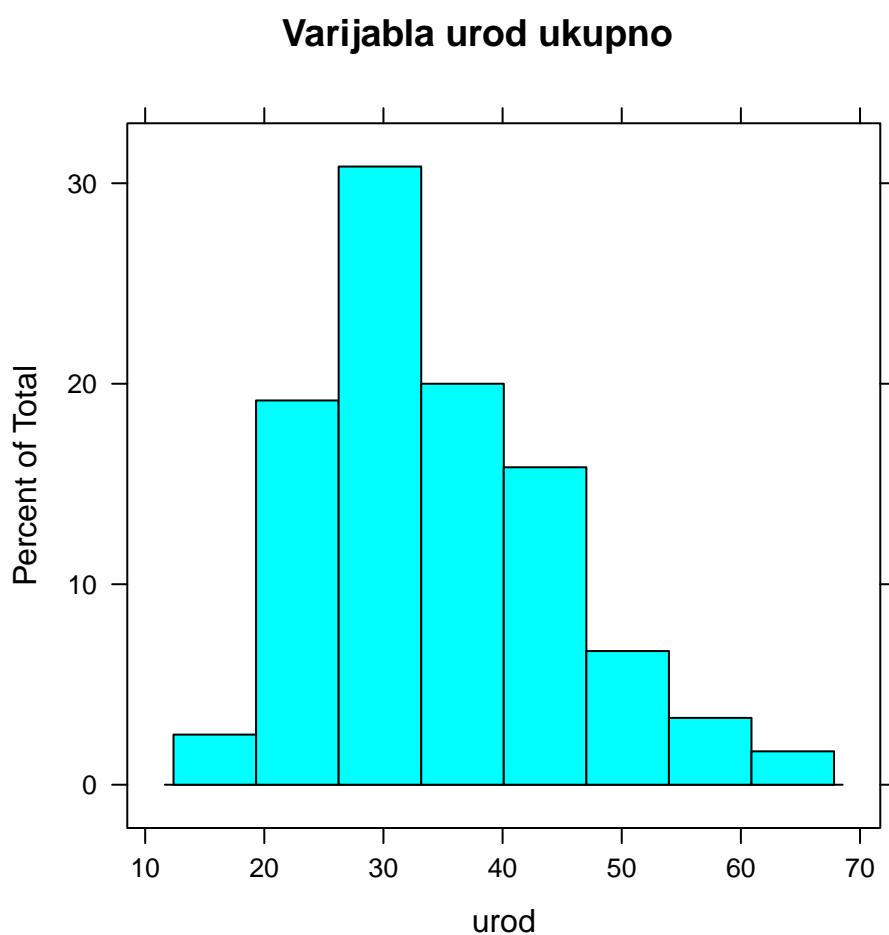
```
barchart(urod ~ varijetet | godina, data = podaci_lattice_grafika,
          groups = ploha, layout = c(1,2), stack = TRUE,
          auto.key = list(space = "right"),
          ylab = "Urod u tonama",
          scales = list(x = list(rot = 90)))
```

```
barchart(urod ~ varijetet, data = podaci_lattice_grafika,
          groups = ploha, layout = c(1,1), stack = TRUE,
          auto.key = list(space = "right"),
```

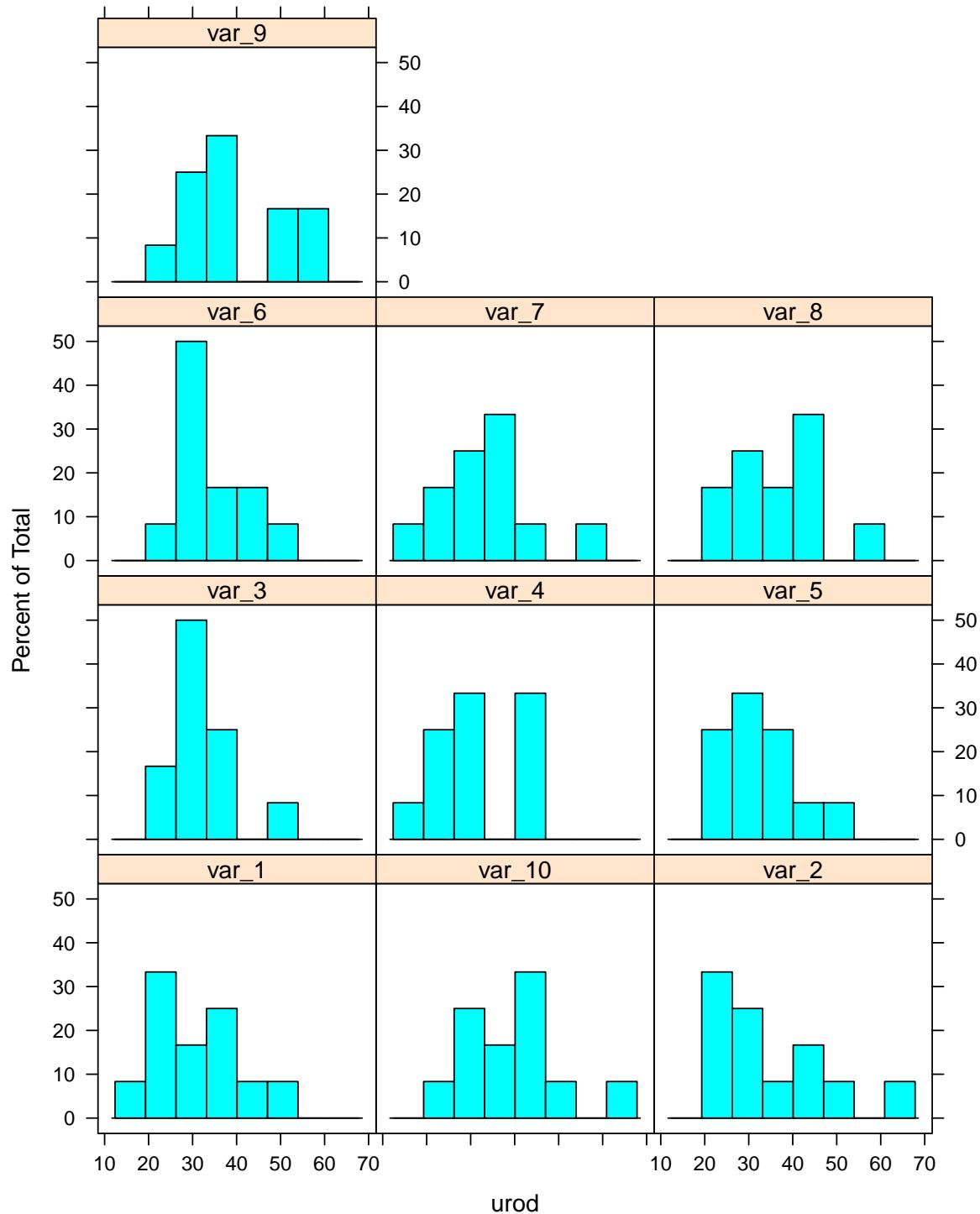




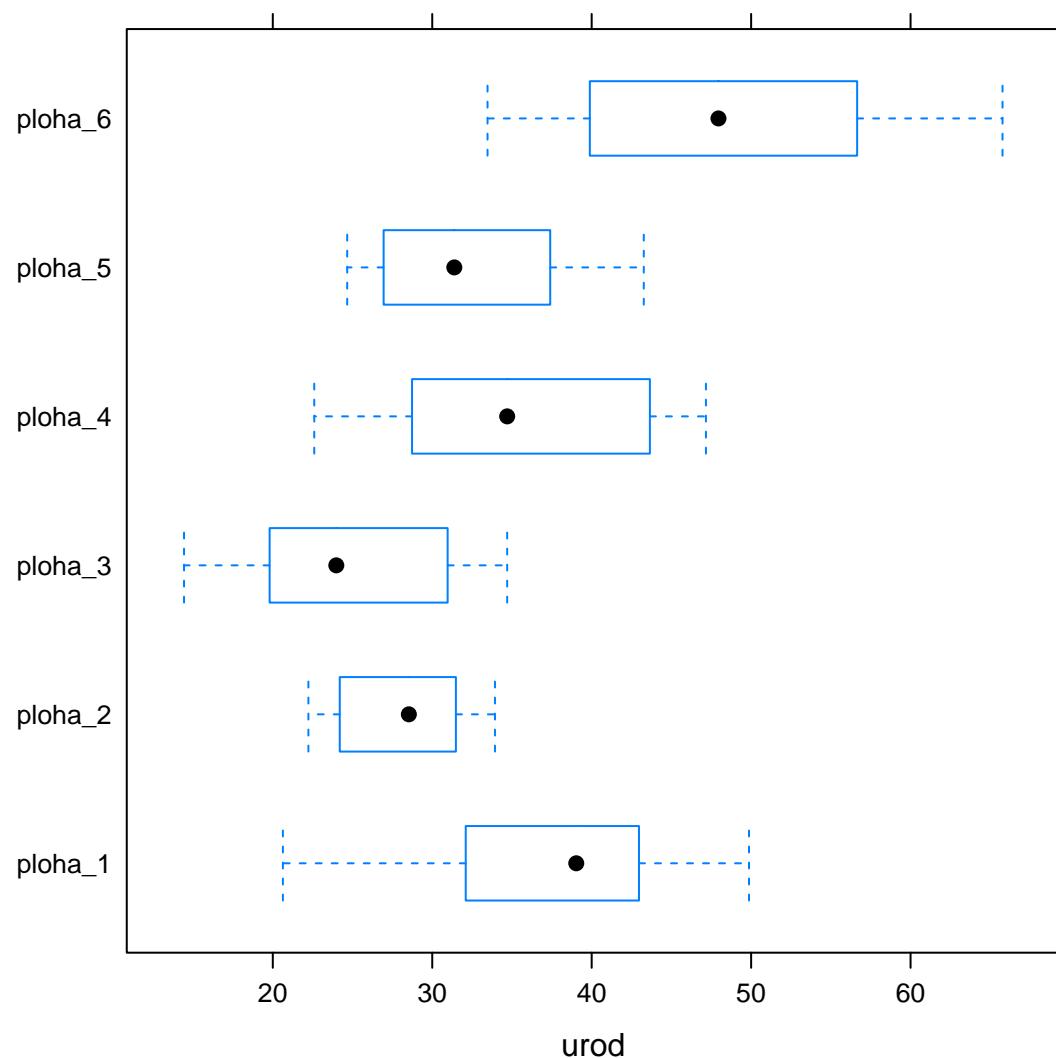
Slika 25: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable



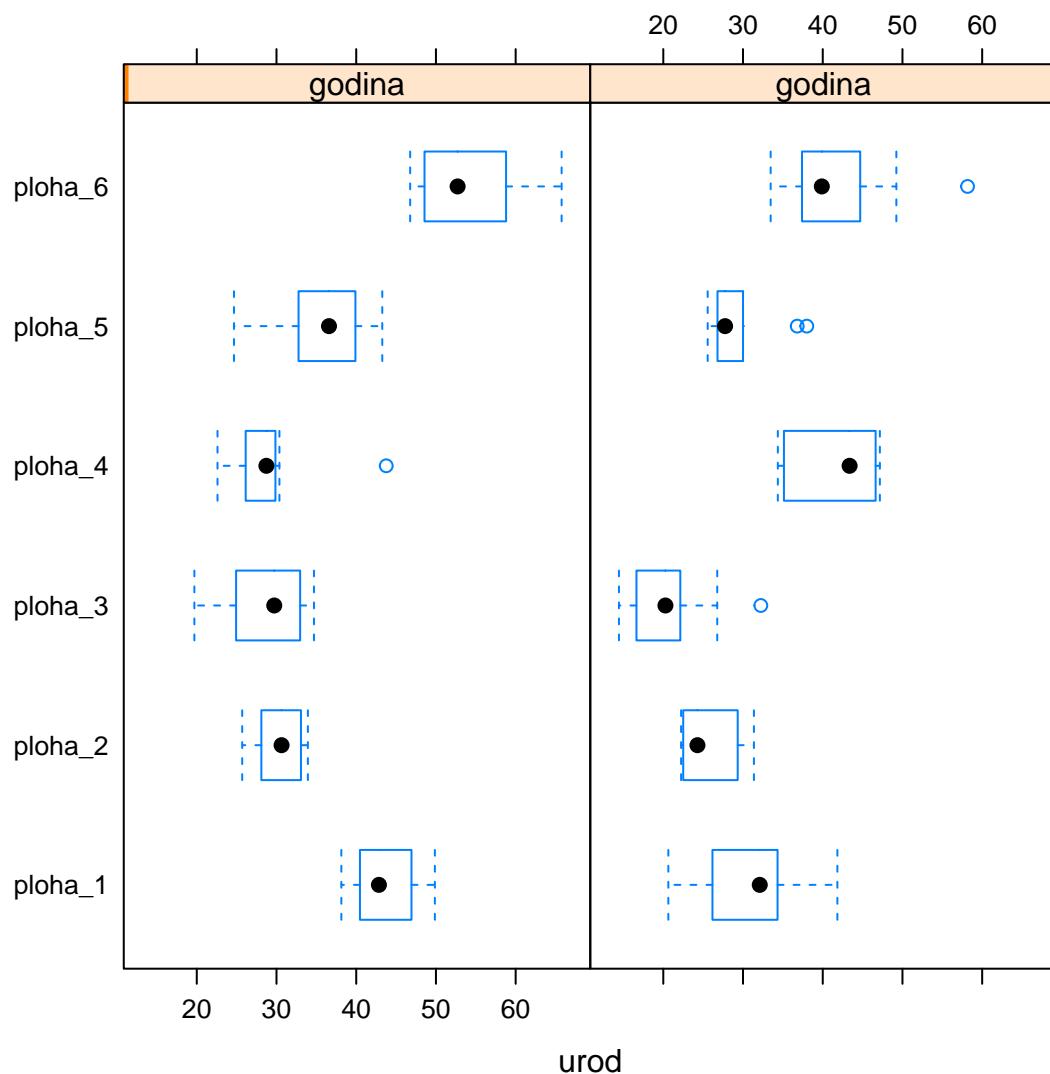
Slika 26: Histogramski prikaz varijable cjelokupno



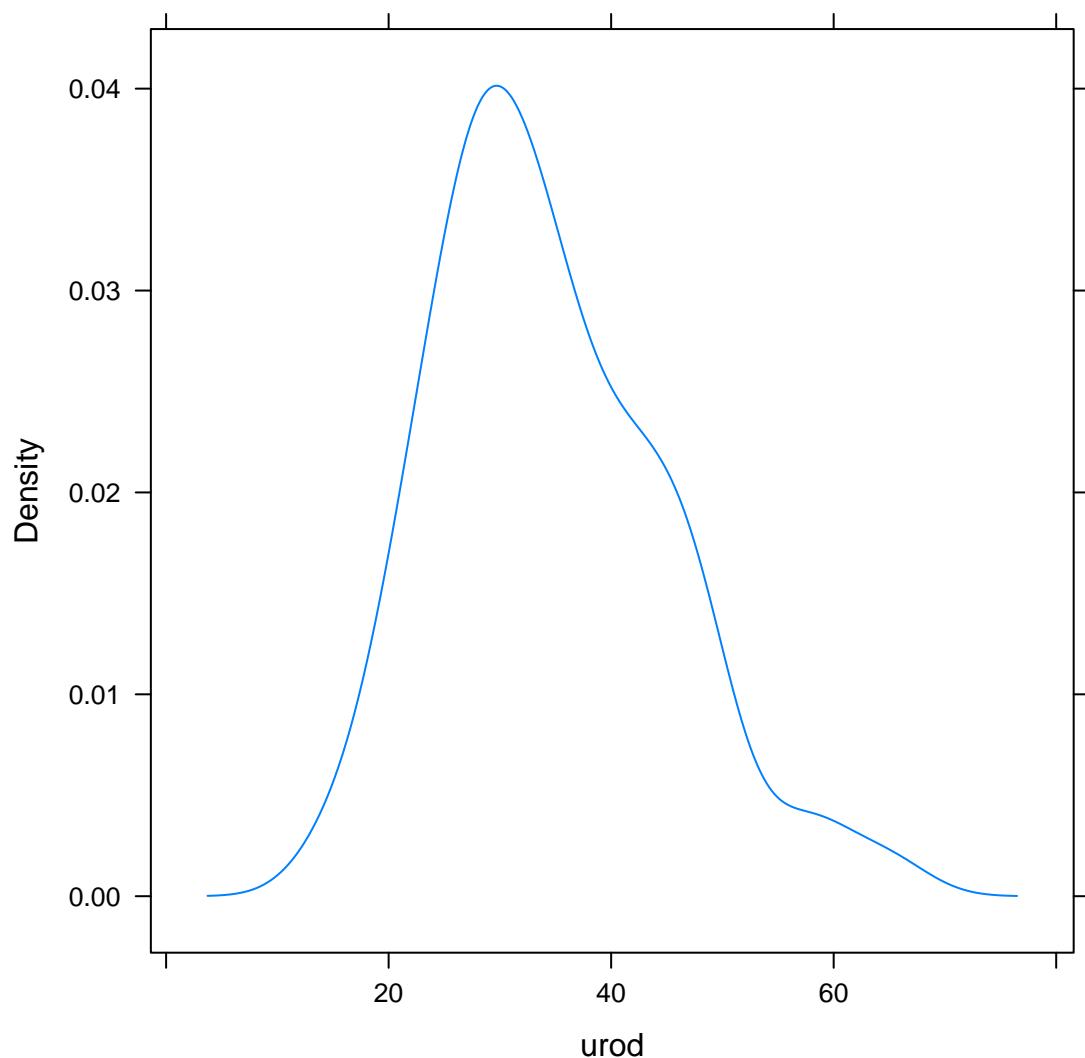
Slika 27: Histogramski prikaz varijable razdijeljene prema nivoima faktorske varijable



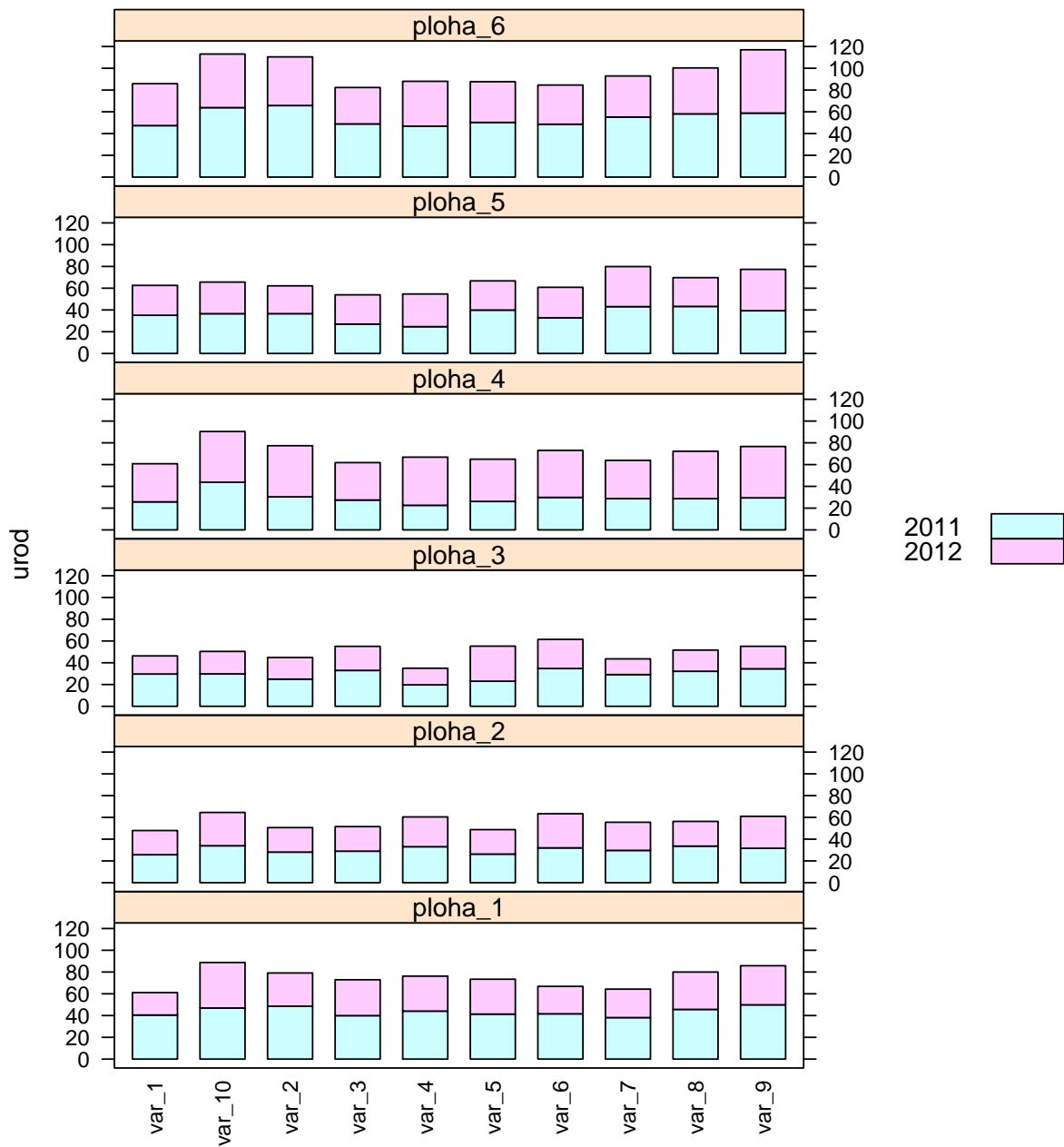
Slika 28: Usporedni Box-Whiskers prikaz varijable urod prema ploham



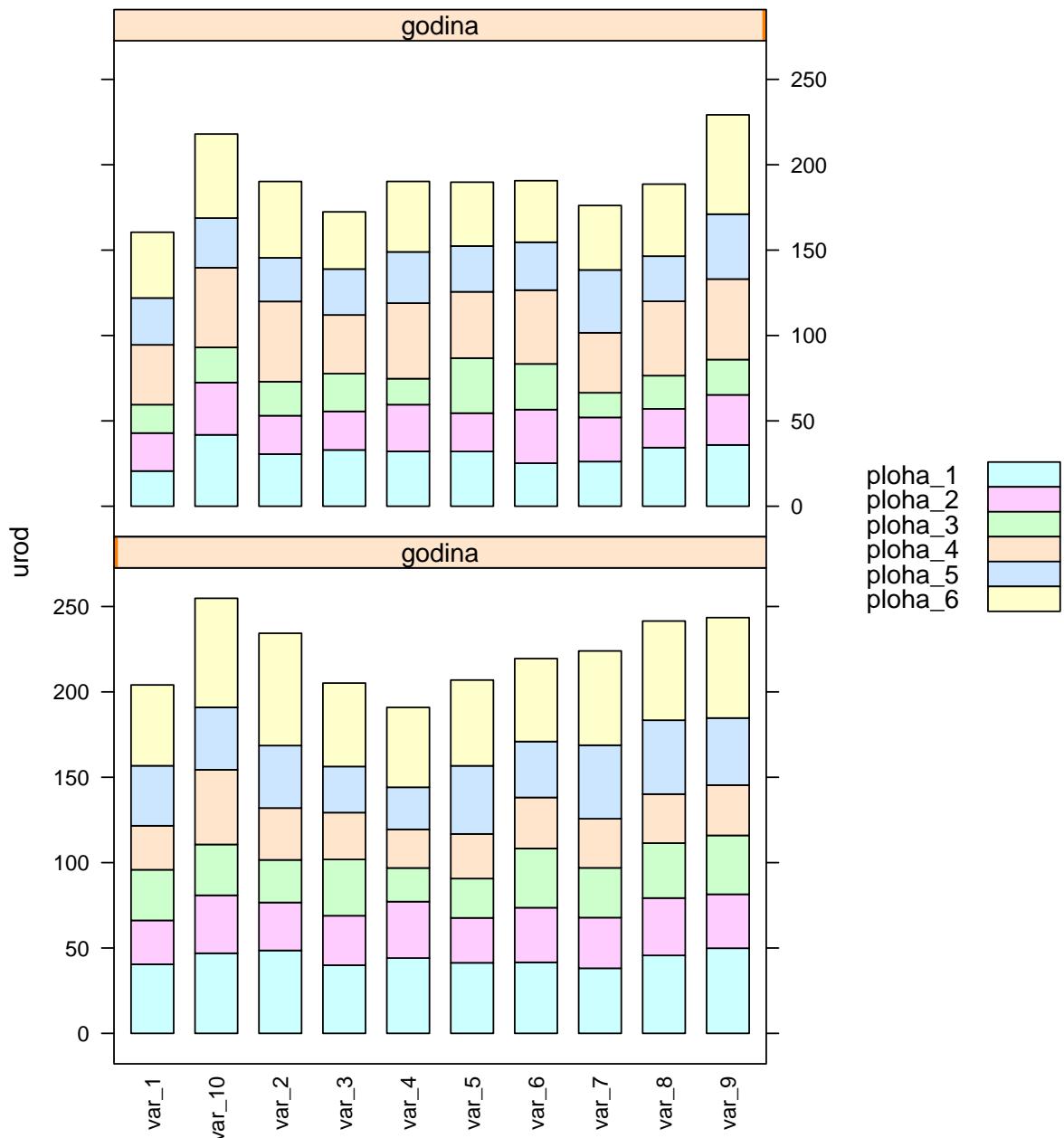
Slika 29: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable godina



Slika 30: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable



Slika 31: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable



Slika 32: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable

```
#ylab = "Urod u tonama",
scales = list(x = list(rot = 90)))
```

Za konačan izgled prikaza potrebno je podatke aggregirati prema godinama. Iako na ovome tečaju neće biti govora o različitim načinima manipulacije podacima (engl. *data wrangling*) samo demonstracija važnosti manipulacije podacima funkcijom *aggregate()*.

```
#nova funkcija aggregate
podaci.agg_godina<- aggregate(podaci_lattice_grafika["urod"], by = list(podaci_lattice_grafika$varijetet),
names(podaci.agg_godina) <- c("varijetet", "ploha", "urod"))
```

```
barchart(urod ~ varijetet, data = podaci.agg_godina,
groups = ploha, layout = c(1,1), stack = TRUE,
auto.key = list(space = "right"),
ylab = "Urod u tonama",
scales = list(x = list(rot = 90)))
```

Kako je ovo tečaj na kojem se pokušavamo upoznati s veoma velikim brojem pojmoveva kako iz samog sustava R tako i iz statistike u dalnjim primjerima tijekom tečaja grafičke prikaze ćemo davati samo u jednom grafičkom sustavu. Kako je za potrebe gradiva *base* sustav znatno fleksibilniji, primjeri će biti rađeni uz pomoć *base* grafike. Srce ima specijalizirane tečajeve o grafičkim sustavima u R-u.

Radi izuzetno velike popularnosti najnovijeg grafičkog sustava koji se temelji na gramatici grafike (engl. *Grammar of graphics*, Wilkinson 2005) spominjemo i treći sustav *ggplot* te izuzetno popularan paket *ggplot2* kojim ćemo se detaljno upoznati na nekom drugom tečaju.

**Napomena:** Ukoliko eksplisitno zovemo funkciju nekog paketa pišemo to sintaksom: *paket::funkcija*. Radi veoma velikog broja funkcija, moguće je ponavljanje neke funkcije u nekoliko kontribuiranih paketa. Na ovaj način kontrolirano zadamo funkciju željenog paketa.

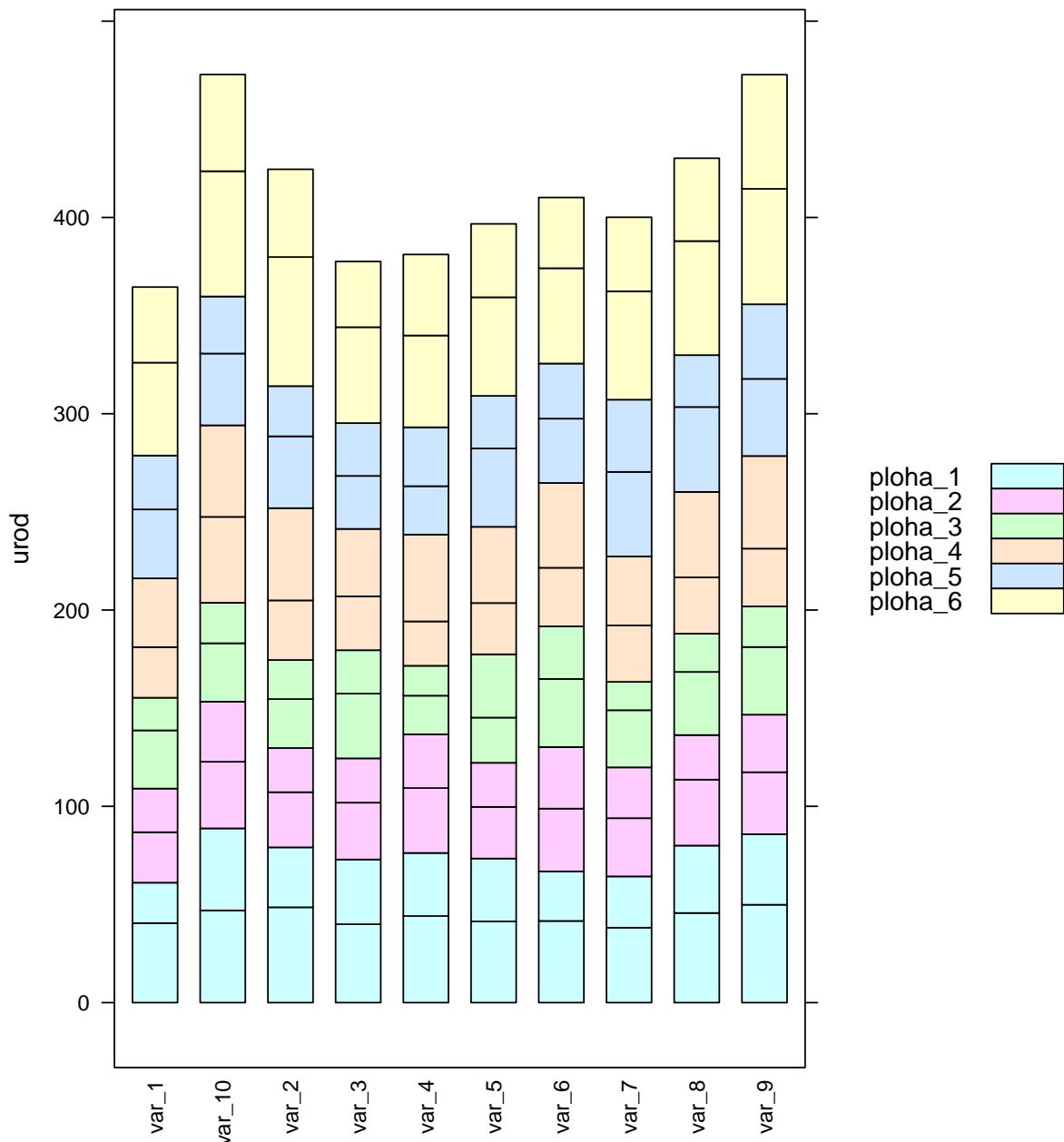
#### ZADACI ZA SAMOSTALNI RAD:

Smostalno provjerite naučeno radom na sljedećim zadacima:

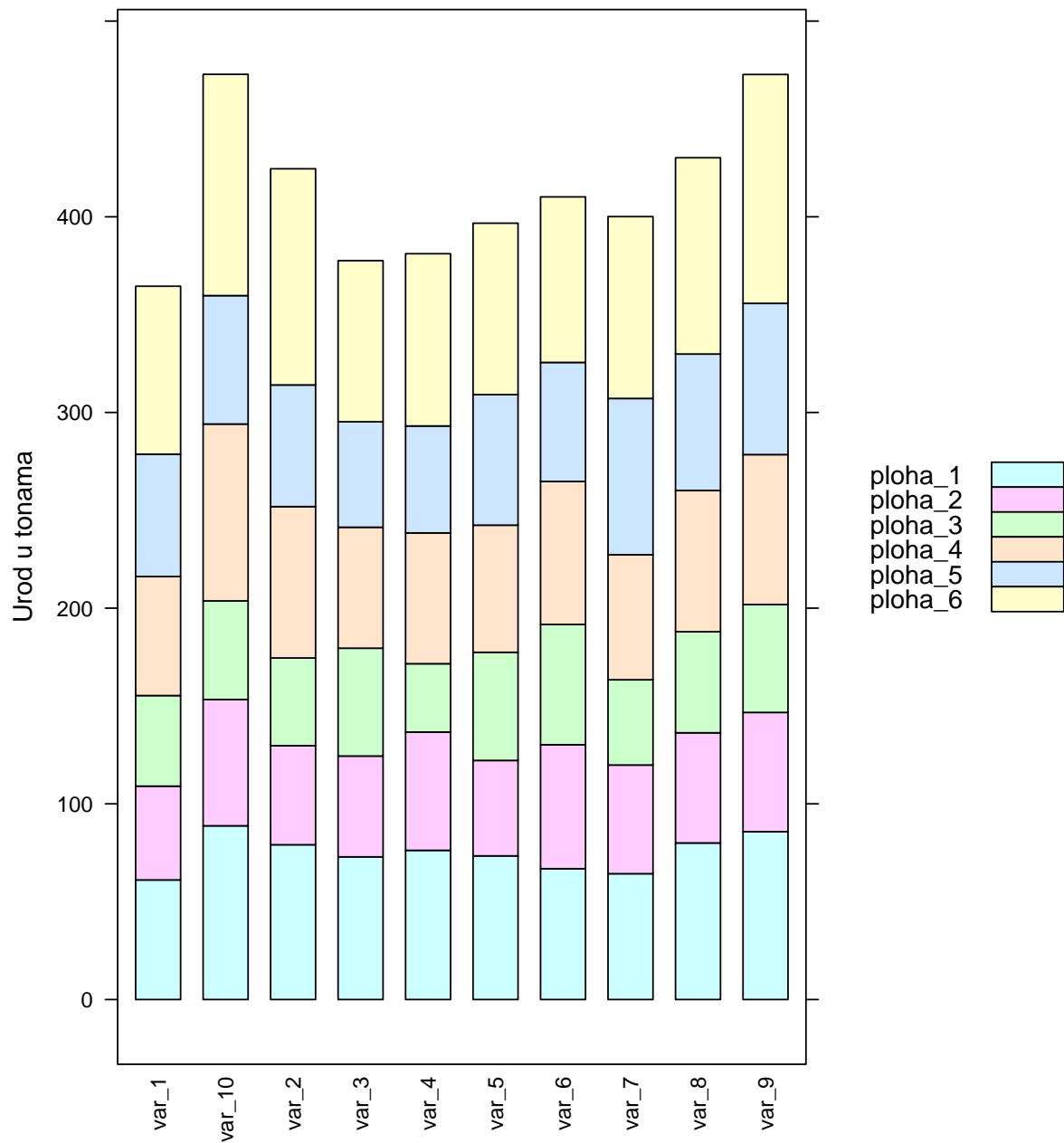
```
#ZADATAK 1: Konstruirajte točke na način da prate funkciju sin(x)/x,
#na području od -10 - 10. Nacrtajte rezultat. Stavite odgovarajući naslov.
#Neka vrijednosti na osi y budu orijentirane jednako kao na osi x. Liniju
#na grafu definirajte kao iscrtkanu, tamno sive boje, debljine 2. Snimite
#graf kao png datoteku 300 dpi na turdi disk.
```

```
#ZADATAK 2:Nacrtajte dijagram raspršenja (scatterplot) varijabli Length i
#Sepal.Width iz skupa podataka koji dolazi u sustavu iris na način da su točke
#obojane različito u ovisnosti o vrsti biljeke, Species.
```





Slika 33: Primjer podjele prikaza na onoliko panela koliko postoji nivoa faktorske varijable



Slika 34: Vizualizacija agregiranih podataka uroda po gpdinama

## 4 Uvod u statistiku uz sustav R

Pojam statistike uglavnom je povezan s analizom numeričkih podataka ali se pod pojmom statistike često misli i na jednostavno sumiranje skupova podataka. Statistika se kao znanstvena disciplina bavi razvojem metoda prikupljanja, opisivanja i analiziranja podataka te primjenom tih metoda u procesu donošenja zaključaka.

Primjenjena statistika je znanost učenja iz podataka na način da kontrolira i komunicira nesigurnost u zaključcima te na taj način pruža bitne smjernice za upravljanje znanstvenim i društvenim napretkom. U svakodnevnom životu riječ statistika koristimo kada brojčanim vrijednostima pokušavamo opisati bitne karakteristike nekog skupa podataka (populacija / uzorak).

Način na koji dolazimo do zaključka koristeći podatke mjerjenja ili opažanja zovemo statistička metoda. Statističke metode su znanstvena disciplina, dio primjenjene matematike, a uključuju načine prikupljana, organizacije, sistematizaciranja, analize i interpretacije podataka.

Primjena statističkih metoda veoma je široka. Nema znanstvene discipline i istraživanja koja ne koriste statistiku prilikom donošenja zaključaka.

Statističari primjenjuju statističke metode i način razmišljanja u raznim znanstvenim, društvenim i poslovnim područjima kao što su astronomija, biologija, obrazovanje, ekonomija, inženjerstvo, genetika, marketing, medicina, psihologija, javno zdravstvo, sport i drugo.

Svrha mjerjenja ili sakupljanja podataka, mogućnost donošenja zaključaka o mjerenoj pojavi. "Najbolja stvar u bavljenju statistikom je ta da statističar ima priliku igrati se u svačijem dvorištu." (John Tukey, Bell Labs, Sveučilište Princeton).

### 4.1 Osnovni pojmovi u statistici

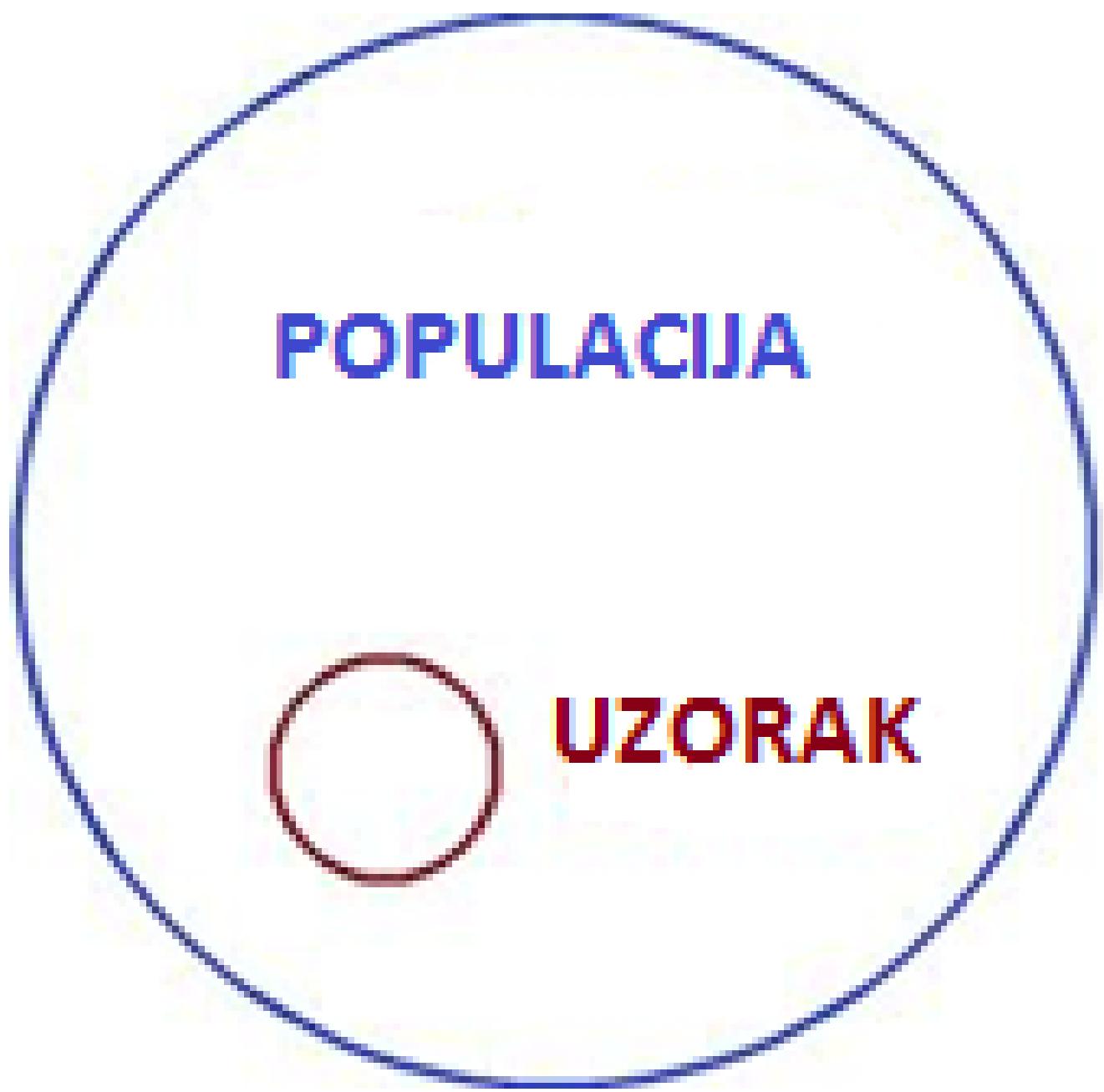
#### 4.1.1 Statistička populacija

U statistici, populacija je ukupan skup jedinki koje su predmet našeg interesa i o kojima želimo donositi zaključke. Populacija može biti konačna ili beskonačna.

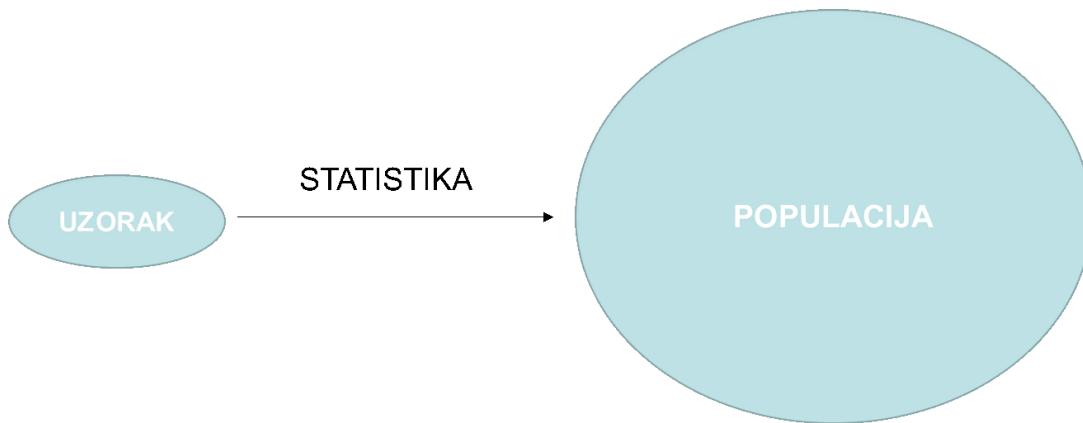
Primjeri statističkih populacija:

Ako želimo znati prosječnu masu svih 20-godišnjaka u Hrvatskoj, onda populaciju od interesa čine svi pojedinci stari 20 godina koji žive u Hrvatskoj. Ako želimo znati udio sredovječnih muškaraca koji nemaju srčani udar nakon što su konzumirali određeni lijek, onda populaciju čine svi sredovječni muškarci koji su primali ovu terapiju. Ako želimo znati postotak loše istokarenih dijelova nekog aparata u određenoj seriji proizvoda tada populaciju čine svi istokareni dijelovi iz određene serije. Ako želimo znati prosječan prihod pop pjevača na domaćoj estradnoj sceni tada je populacija skup svih pop pjevača, itd. Budući je mjerjenje svih jedinki u populaciji najčešće nemoguće, populacija se uzorkuje procesom selekcije podskupa. Uzorak se odabire na način da bude reprezentativan za populaciju koja se istražuje.





Slika 35: Populacija i uzorak



Slika 36: Statističko zaključivanje

#### 4.1.2 Statističko zaključivanje

Statističko zaključivanje je zaključivanje o populaciji od interesa, a osniva se na reprezentativnom uzorku iz populacije. Proces zaključivanja može se prikazati ovako:

#### 4.1.3 Uzorak

Uzorak je skup jedinki izvučen iz populacije prema jsano određenoj proceduri. Jedinice uzorka još zoveno opservacijama ili jedinicama uzorka. Uzorci se dijele na nevjerojatnosne i vjerojatnosne, koji se koriste pri statističkom zaključivanju.

**4.1.3.1 Uzorci bez definirane vjerojatnosti (ne-vjerojatnosni uzorci)** Ne-vjerojatnosni uzorci su uzorci dobiveni različitim načinima ali za koje vrijedi da postoje dijelovi populacije za koje ne postoji šansa ( $\text{šansa} = 0$ ) da budu izabrani u uzorak ili oni gdje se šansa izbora jedinke (elementa, opservacije) ne može precizno odrediti.

Ne-vjerojatnosni uzorci koriste se u pilot studijama, kvalitativnim istraživanjima, razvojima hipoteza i ne smiju se koristiti u statističkom zaključivanju.

Elementi se u uzorak odabiru temeljem pretpostavki o populaciji koja nas zanima kao osnovnom kriteriju za izbor. Iz razloga što elementi nisu izabirani na slučajan način, na ne-vjerojatnosni uzorcima ne možemo izračunati i odrediti pogrešku uzorkovanja (engl. *sampling error*) kao jednu od najvažnijih mjera u statistici. To je i razlog radi čega zaključke donesene na ovakvom tipu uzoraka ne bismo smjerili ekstrapolirati na populaciju tj. donositi zaključke o populaciji.

Metode za dobivanje ne-vjerojatnosnih uzoraka su prigodno uzorkovanje (engl. *convenience sampling*), *quota* uzorkovanje (kombinacija stratificiranog i prigodnog) te namjensko (engl *purposive*) uzorkovanje koje je također karakterizirano kao selektivno, subjektivno uzorkovanje. O ovim tipovima uzoraka neće biti više govora u ovome tečaju.

**4.1.3.2 Vjerojatnosni uzorci** Osnovna karakteristika vjerojatnosnih uzoraka je vjerojatnost veća od nule da budu izabrani u uzorak i ta vjerojatnost se može jasno odrediti. Ovo omogućuje nepristran (engl. *unbiased*) izračun populacijskih parametara ponderiranjem vrijednosti proporcionalno vjerojatnosti izbora jedinke u uzorak.

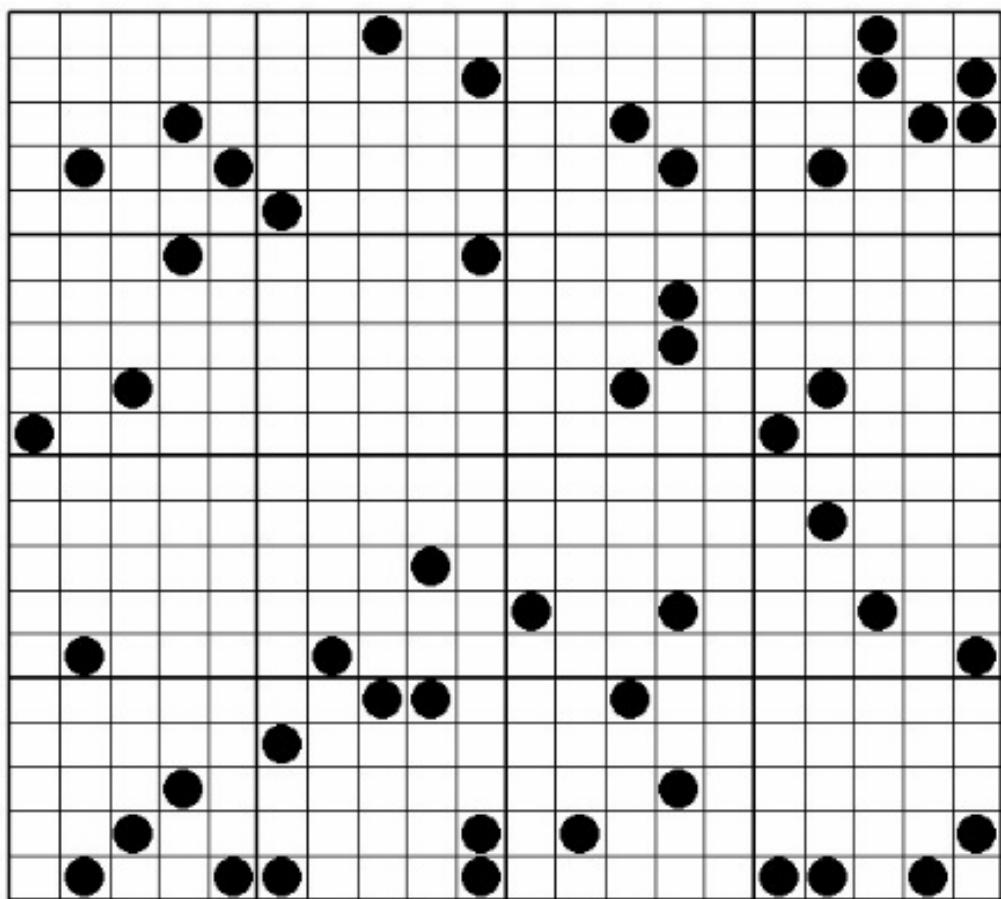
**4.1.3.2.1 Jednostavan slučajni uzorak** Slučajno izabiremo n jedinki iz populacije veličine N pri čemu svaka jedinka ima jednaku šansu (vjerojatnost) da bude izabrana u uzorak. Iz ovoga proizlazi da je vjerojatnost izbora svakog para jedinki, trojki itd. jednak. Ovo svojstvo minimizira pogrešku i variancu između jedinki uzorka veoma dobro procjenjuje cjelokupnu varijancu populacije, a time i preciznost rezultata.

PREDNOSTI: jednostavnost NEDOSTATCI: postoji šansa da uzorak nije dovoljno precizan; uzorak može biti preskup.

Može se dogoditi da potpuno slučajnim odabirom jedinki u uzorak on ne reflektira pravo stanje populacije. Također, veoma se lako može dogoditi da izbor jedinki prema nekom svojstvu bude prevelik / premalen te da izabrani uzorak ne reprezentira strukturu populacije. Ovo se rješava stratificiranim uzorkovanjem.

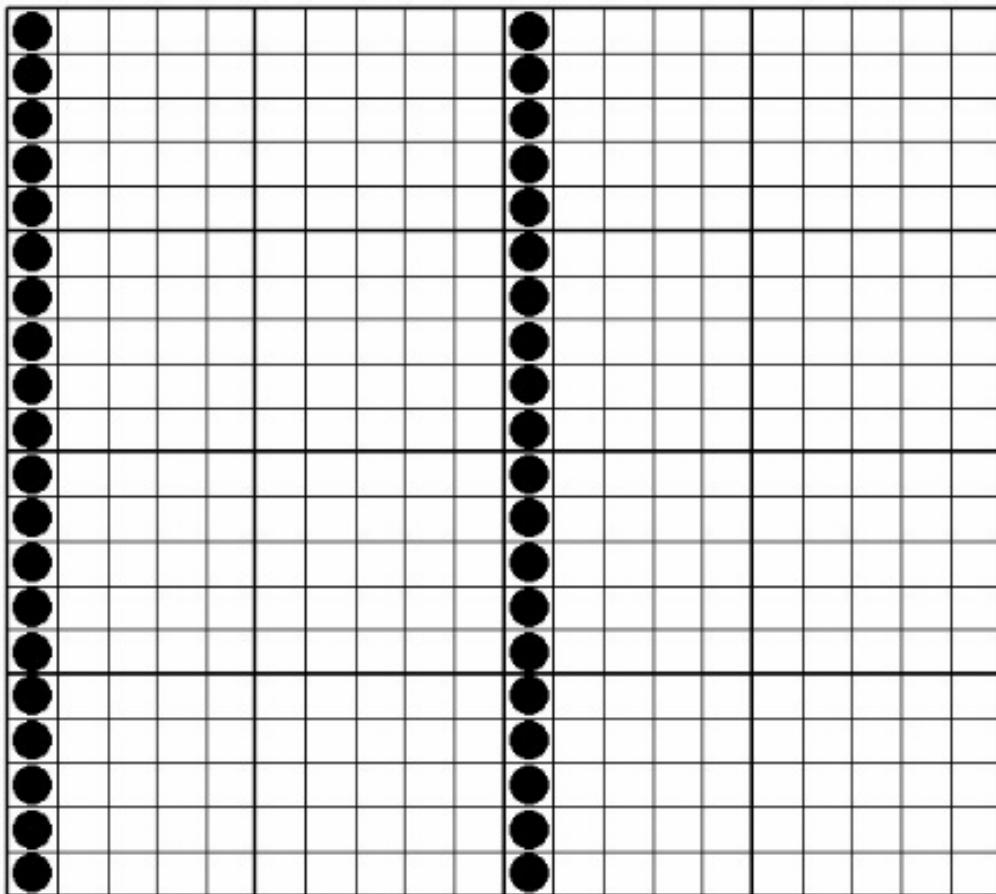


### SIMPLE RANDOM SAMPLE



Slika 37: Vjerojatnosni uzorci - jednostavan slučajni uzorak

## SYSTEMATIC SAMPLE



Slika 38: Vjerojatnosni uzorci - sistematski slučajni uzorak

Potpuno slučajni uzorak može biti veoma nezgodan za provedbu ukoliko ga želimo izvesti na izrazito velikim populacijama što ga čini i veoma skupim i teškim za provedbu. Budući su istraživačka pitanja, ipak, najčešće vezana za neke subpopulacije, to će i ovaj problem potpuno slučajnog jednostavnog uzorka biti riješen stratificiranim uzorkovanjem.

**4.1.3.2.2 Sistematski (sustavni) uzorak** Prva jedinka koja ulazi u uzorak izabire se slučajno; nakon toga izabiremo svaku n-tu jedinku iz populacije veličine N (npr. svaku desetu, svaku šestu). Broj n još se naziva i interval uzorka ili pomak (engl *skip*) Ovaj način izbora uzorka uvjetuje da svi elementi populacije imaju jednaku i poznatu vjerojatnost biti izabrani u uzorak.

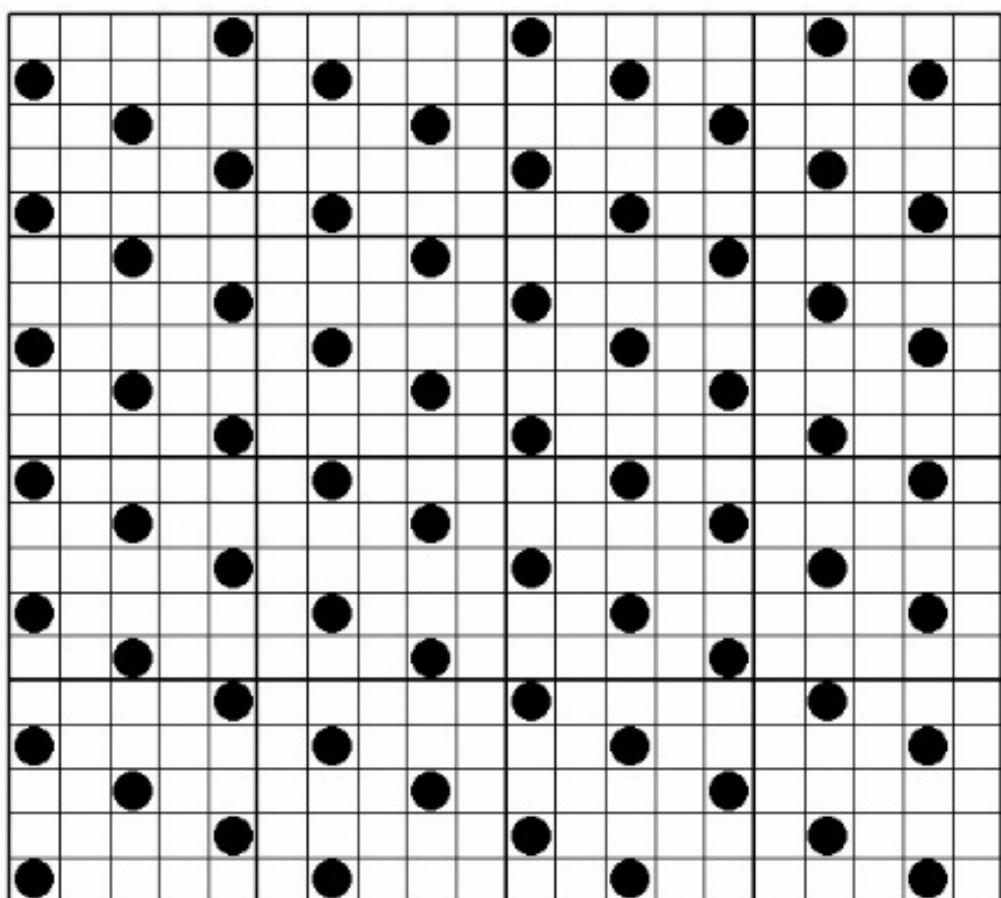
PREDNOSTI: jednostavno za objašnjenje i sprovođenje NEDOSTATCI: netočan ukoliko postoji obrazac u podacima koji se ponavlja ciklički.

Budući su jedinice uzorkovanja u sistematskom uzorku uniformno raspoređene ovakav način uzorkovanja nije dobar ukoliko unutar populacije postoji bilo kakav obrazac.

**4.1.3.2.3 Stratificirani uzorak** Zahtijeva podjelu populacije u nepreklapajuće podskupove (stratume). Nakon stratifikacije svaki se stratum može uzorkovati kao posebna subpopulacija. U ovisnosti o veličini podskupova, iz svakog se podskupa izabire slučajni uzorak odgovarajuće veličine. Način izbora uzorka unutar stratum može varirati (potpuno slučajni, sustavni)

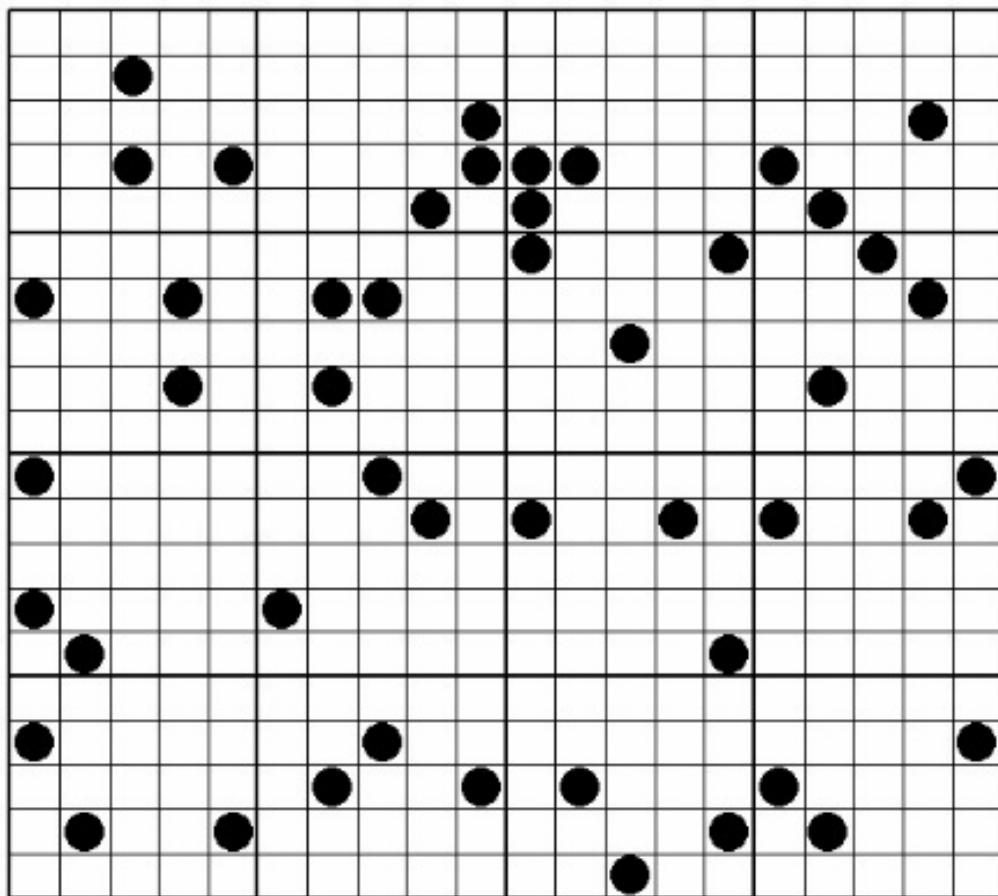


### SYSTEMATIC SAMPLE



Slika 39: Vjerojatnosni uzorci - sistematski slučajni uzorak 2

## STRATIFIED SAMPLE



Slika 40: Vjerojatnosni uzorci - stratificirani slučajni uzorak

PREDNOSTI: točnost

Izborom stratificiranog uzorka preciznost načih procjena o populaciji mogu biti bolje od zaključaka dobivenih temeljem jednostavnog slučajnog uzorka.

NEDOSTATCI: zahtijeva određeno predznanje o populaciji koja se uzorkuje (broj i veličina stratuma), odabir varijable za stratificiranje ponekad nije jednostavan, skup za provođenje, nije koristan ukoliko ne postoje homogene subpopulacije.

Postoje potencijalni nedostaci prilikom stratificiranog uzorkovanja. Uz to što povećava kompleksnost izbora uzorka povećava se i kompleksnost izračuna procjena populacijskih parametara. Dodatno, podjela populacija na strume prema jednom svojstvu ne mora nužno reflektirati stanje struma u nekom drugom.

Svakako se preporuča korištenje stratificiranog uzorka ukoliko vrijedi sljedeće:

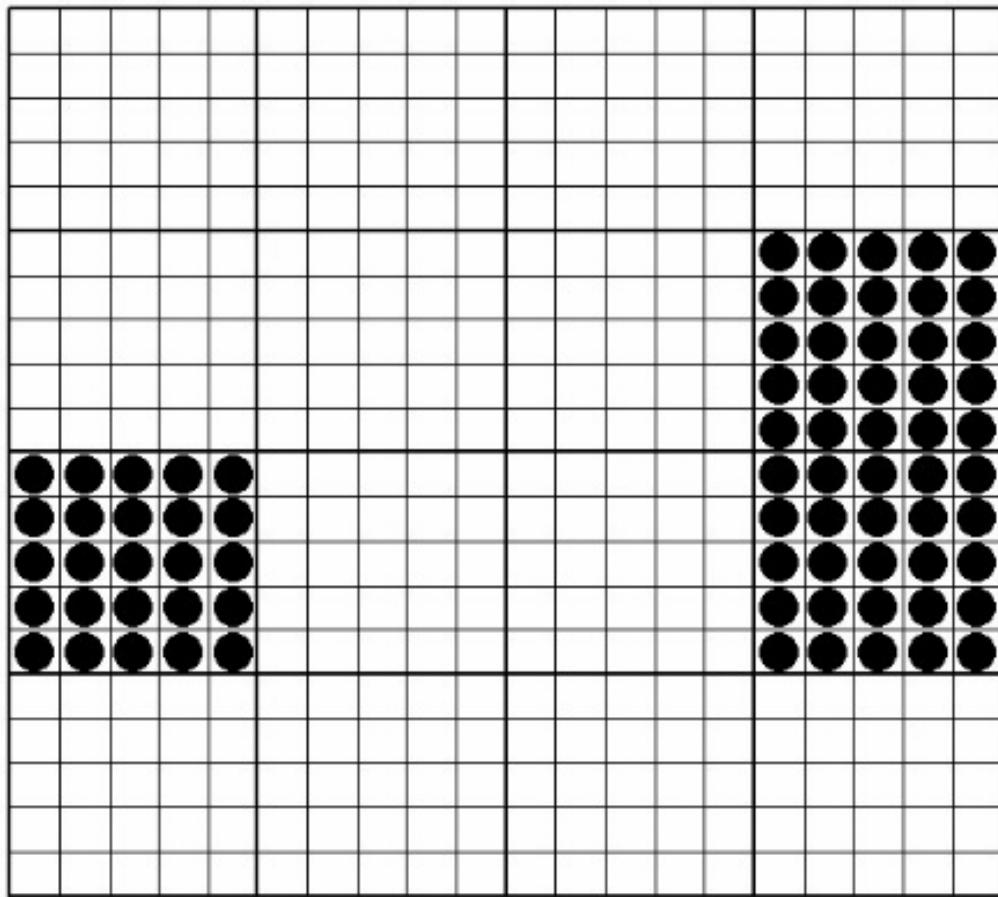
- Varijabilnost unutar struma je mala
- Varijabilnost između struma je velika
- Varijabla od interesa snažno je korelirana s varijablom prema kojoj smo stratificirali populaciju.

Poststratifikacija

Ukoliko se stratificiranje uzorka radi nakon njegova izbora, proces nazivamo poststratifikacijom. Najčešće se provodi radi



## CLUSTER SAMPLE



Slika 41: Vjerojatnosni uzorci - klaster slučajni uzorak

nedostatnog znanja o populacijskim stratumima prije izbora potpuno slučajnog uzorka. Uvođenje ponderiranja makar i uz sve nedostatke post-hoc procesa često daje bolje procjene populacijskih parametara od korištenja potpuno slučajnog uzorka.

**4.1.3.2.4 Klaster uzorak** Zahtijeva slučajan izbor klastera, nakon čega slijedi uzorkovanje svih jedinica unutar odabranih klastera. Slično stratificiranom uzorku, zahtijeva podjelu populacije na nepreklapajuće klastere.

PREDNOSTI: manji troškovi NEDOSTATCI: neprecizan ukoliko postoji velika varijabilnost među klasterima.

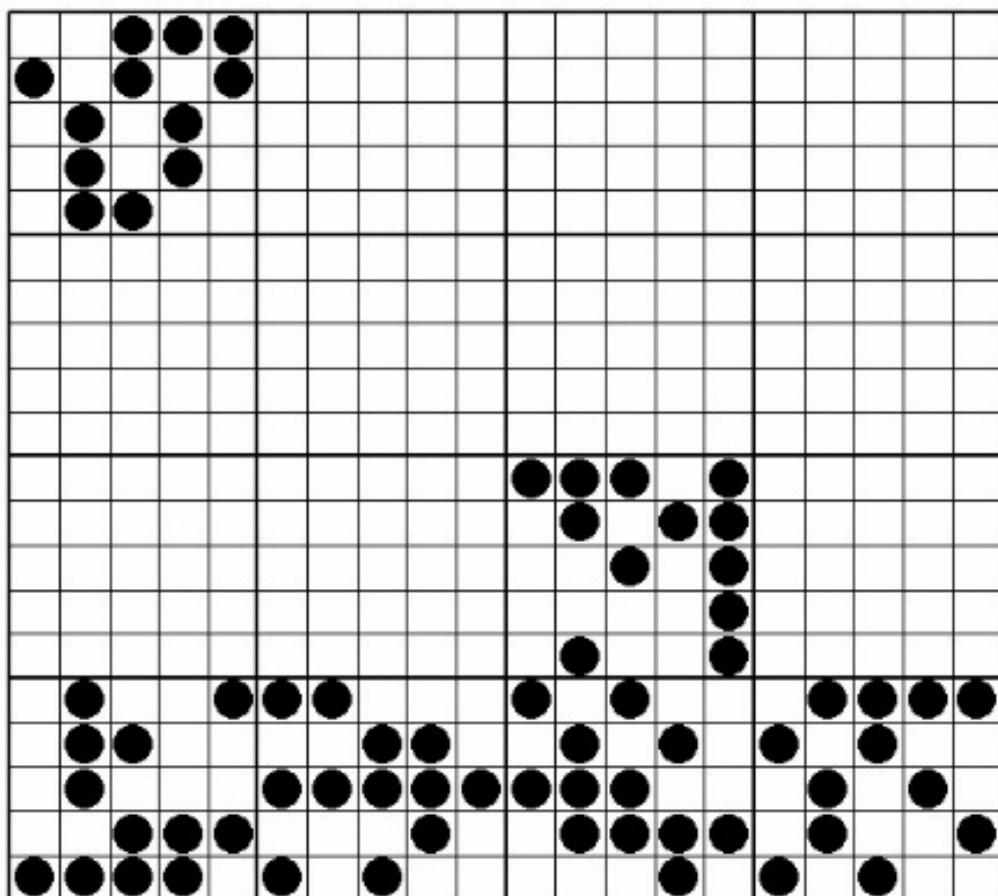
**4.1.3.2.5 Kombinirani uzorak** Klaster uzorak, pri čemu se iz svakog slučajno odabranog klastera izabire slučajni ili sistematski uzorak.

Ukoliko imamo neki od vjerojatnosnih uzorka tada proces statističkog zaključivanja možemo detaljnije opisati kao sljedeći proces:

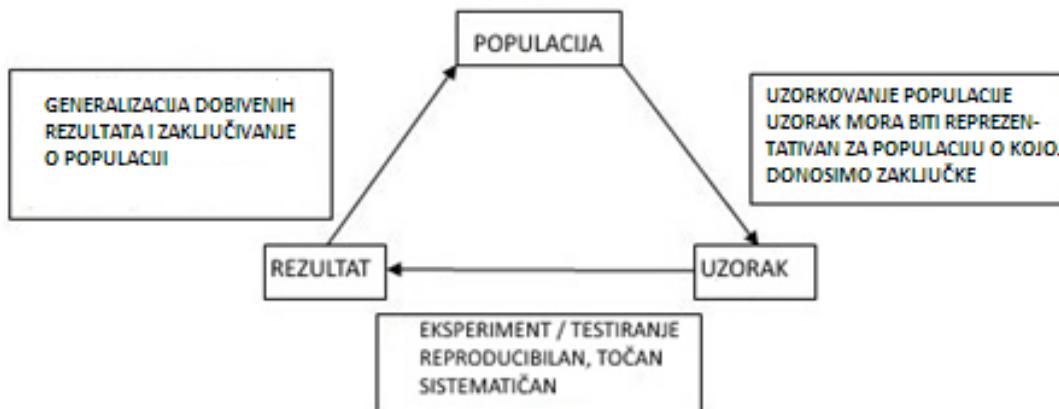
### 4.1.3.3 Zavisnost i nezavisnost dva uzorka



### TWO-STAGE SAMPLE (CLUSTER THEN SRS)



Slika 42: Vjerojatnosni uzorci - kombinirani slučajni uzorak



Slika 43: Proces statističkog zaključivanja

**4.1.3.3.1 Nezavisni uzorci** Ako mjerena svojstva od interesa na jednom uzorku ne daju nikakve informacije o mjerjenjima iste osobine na drugom uzorku (u nezavisnim uzorcima nalaze se različite jedinke). Nezavisni uzorci su uzorci prikupljeni iz dviju populacija gdje je proces selekcije jedinki u uzorak neovisan o procesu selekcije u drugoj populaciji.

**4.1.3.3.2 Zavisni uzorci** Zavisni uzorci su uzorci izabrani iz dvije populacije na način da ili 1) elementi uzorkovani su dijelovi obje populacije ili 2) izabrane jedinke (elementi) su slični u karakteristikama ili se mogu upariti s elementima izabranim iz prve populacije. Drugačije rečeno, zavisni uzorci su uparena mjerena osobina od interesa na istom skupu jedinki ili ukoliko su mjerena u jednom uzorku na neki način povezana s mjerjenjima u drugom uzorku.

Primjer nezavisnih uzoraka:

Slučajnim izborom, na početku semestra biramo 50 studenata sa druge godine, izabranog smjera i fakulteta Sveučilišta u Zagrebu i testiramo njihovo znanje iz matematike.

Slučajnim izborom, na kraju semestra, biramo 50 studenata sa druge godine izabranog smjera i fakulteta Sveučilišta u Zagrebu i testiramo njihovo znanje iz matematike.

Primjer zavisnih uzoraka:

Slučajnim izborom, na početku semestra, biramo 50 studenata sa druge godine izabranog smjera i fakulteta Sveučilišta u Zagrebu i testiramo njihovo znanje iz matematike.

Na kraju semestra, na istih 50 studenata sa druge godine izabranog smjera i fakulteta Sveučilišta u Zagrebu testiramo znanje iz matematike.

#### PITANJA ZA PONAVLJANJE:

Radi li se u sljedećim primjerima o zavisnim ili nezavisnim uzorcima?

- 1) Istraživač je proveo pokus kako bi utvrdio da li specifične vježbe oka mogu popraviti periferni vid kod djece. Slučajnom odabiru 10 osoba izmjerena je kvaliteta perifernog vida. Program vježbi je održan tijekom dva tjedna te je po završetku ponovno testirana kvaliteta perifernog vida na svim osobama.
- 2) Da li se u Dalmaciji riba lakše ulovi s broda ili obale? Jednak broj sati pokušaja ulova ribe s broda i obale napravilo je dvadeset dobrovoljaca jednom tijekom svakog mjeseca u godini.
- 3) Pretpostavimo da su izabrane dvije grupe ispitanika, studenata Sveučilišta u Zagrebu, slučajnim izborom 2010. i 2012. godine kako bi odgovorili na anketu o zadovoljstvu uslugom studentske prehrane. U oba uzorka ste izvučeni i Vi.



#### 4.1.4 Varijable

Varijabla je svojstvo ili stanje čije vrijednosti variraju. Vrijednosti varijable se mijenja u skladu s različitim čimbenicima. Vrijednosti nekih varijabli se mijenjaju brzo, poput vrijednosti dionica na burzi, dok se vrijednosti drugih varijable mijenjaju rijetko, primjerice promjena iste osobe tijekom života.

Konceptualno, varijable dijelimo na nezavisne i zavisne (ponekad zvane varijablama ishoda). Nezavisne varijable zovu se eksperimentalne kada se njima manipulira u istraživanju, ili prediktorske varijable kojima se ne manipulira u istraživanju. Cilj je ustanoviti utjecaj nezavisnih na zavisnu varijablu.

**PRIMJER:** Učitelja matematike zanimaju faktori (varijable) koji utječu na rezultate testova matematike njegovih učenika. Iz iskustva (poznavanje fenomena) pretpostavlja da bi na rezultate mogla utjecati količina vremena uložena u rješavanje zadataka te opća inteligencija učenika (IQ).

**Varijable:** Nezavisne (eksperimentalne/prediktorske): vrijeme učenja, IQ Zavisna (varijabla ishoda): ishod testa iz matematike.

Prema vrijednostima koje varijabla može poprimiti varijable mogu biti:

- kvalitativne (atributivne, kategoriskske): nominalne i ordinalne.

Kvalitativne varijable su one čiji se podaci ne mjere ili broje. Primjer kvalitativne varijable bio bi boja očiju, spol, mjesto stanovanja, pripadnost vjerskoj zajednici itd. Ponekad se za pripadnost nekoj kategoriji ili klasi koriste i brojevi ali treba biti svjestan da to oni u svojoj suštini nisu.

- kvantitativne (numeričke): diskrete i kontinuirane.

Kvantitativne varijable su one čije se vrijednosti mogu mjeriti ili brojati. Kvalitativne varijable su one mjerene na nominalnoj i ordinalnoj skali (s mjernim skalama upoznat ćemo se kasnije u tečaju).

**4.1.4.1 Mjerenje varijabli** Kako bismo mogli zaključivati o odnosima između varijabli, varijable se najprije moraju mjeriti. Mjerenje je konceptualizacija - pridruživanje brojeva vrijednostima varijable. Povezivanje koncepta (ideje) s mjeranjem (tehnika) čini vrijednosti varijable empirijski vidljivim. Proces mjerenja varijabli zahtijeva skale mjerenja koje možemo shvatiti kao shemu za numeričku reprezentaciju vrijednosti varijable.

Mjerenje može biti direktno za varijable kao što su visina, temperatura, težina ili pak indirektno kao što su primjerice varijable motivacija, inteligencija, pogodnost staništa za vrstu, znanje itd. U ovisnosti koji tip varijable mjerimo moramo izabrati adekvatno oruđe, mjerni instrument, kao što su primjerice vaga, mjerka, test, upitnik, termometar, upitnik osobnosti i drugi.

**4.1.4.1.1 Tipovi mjernih skala** Skale se razlikuju po tipu varijabli koje predstavljaju te svojstvima i interpretaciji brojeva sadržanih u skali. Vrsta skale kojom smo mjerili varijable određuje koje statističke postupke možemo koristiti prilikom njihove analize. Primjeri korištenja nominalne skale prikazani su na slici 44.

##### Nominalna skala

Vrijednosti dodijeljene varijabli predstavljaju opisne kategorije, ali nemaju uređene numeričke vrijednosti u odnosu na veličinu. Sluče samo za kategorizaciju jedinki u konačni broj grupa koje se međusobno ne preklapaju (disjunktnе grupe).

**PRIMJERI:** ime osobe, tip proizvoda, spol, religijsko opredjeljenje, nacionalnost. **RAČUNSKE OPERACIJE:** prebrajanje jedinki unutar grupe.

##### Ordinalna skala

Podatke je moguće poredati po veličini ili prema intenzitetu. Pokazuju relativan položaj elemenata u grupi. Primjeri korištenja nominalne skale prikazani su na slici 45.

**PRIMJER:** ocjena, stupanj obrazovanja, kvaliteta proizvoda.

**RAČUNSKE OPERACIJE:** >,<

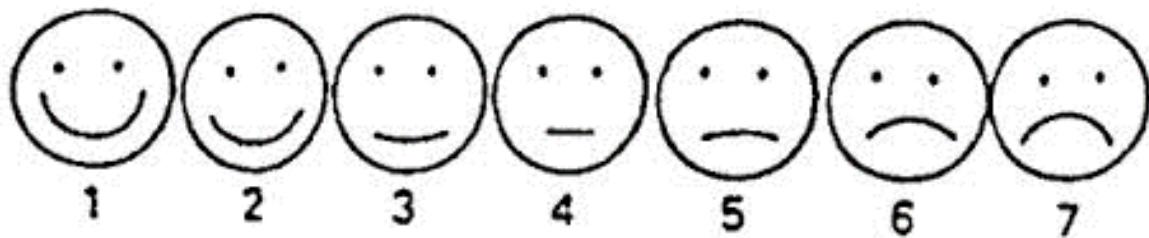
##### Intervalna skala

Vrijednosti varijable su numeričke, na kontinuiranoj skali (realni brojevi). Jednake razlike u izmjerenim vrijednostima varijable predstavljaju jednakе razlike mjerenoj svojstvu; ne postoji apsolutna nula. Primjeri korištenja nominalne skale prikazani su na slici 46.

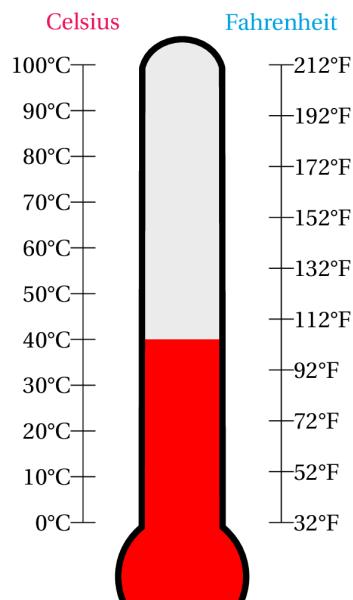




Slika 44: Primjer korištenja nominalne skale



Slika 45: Primjer korištenja ordinalne skale



Slika 46: Primjer korištenja intervalne skale

**PRIMJER:** vodostaj rijeke, temperatura u na Fehrenheit ili Celzijus skali, nadmorska visina.

**RAČUNSKE OPERACIJE:** oduzimanje, zbrajanje i ostalo, ali ne i omjeri vrijednosti

#### Omjerna skala

Skale posjeduju absolutnu nulu koja predstavlja odsutnost mjernog svojstva. Primjeri korištenja nominalne skale prikazani su na slikama 47 i 48.

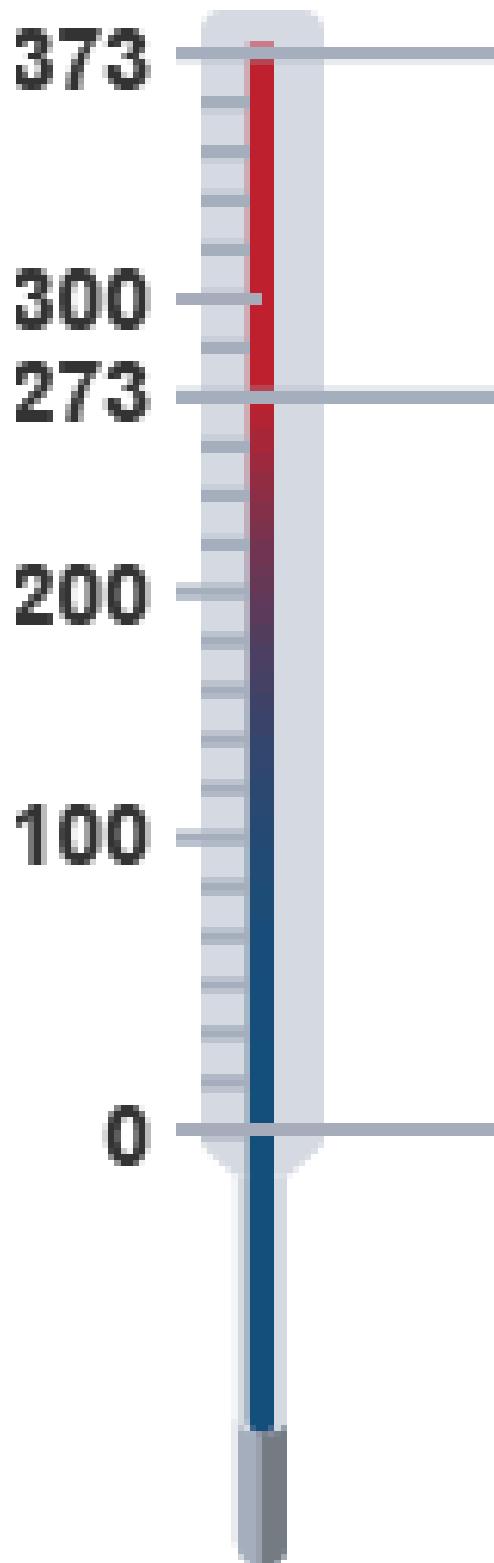
**PRIMJER:** masa, visina, temperatura u na Kelvin skali.

**RAČUNSKE OPERACIJE:** uz ranije navedeno omjeri vrijednosti variable imaju smisla.

Generalno, skale možemo poredati prema njihovoj kompleksnosti kako je prikazano na slici 49.



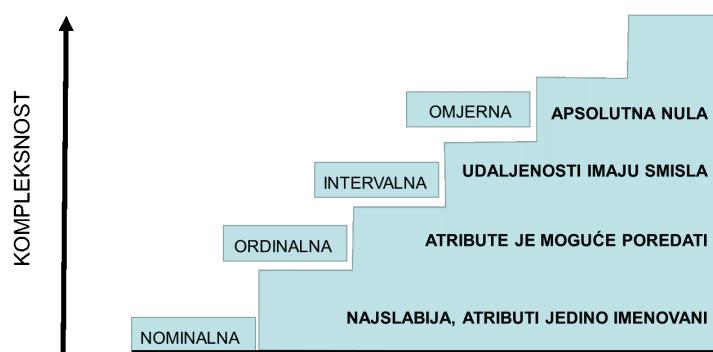
Slika 47: Primjer korištenja omjerne skale



kelvin

---

Slika 48: Primjer izljeđivanja omjerne skale



Slika 49: Odnos skala mjerjenja varijabli

#### 4.1.5 Podaci

Izmjerena svojstava od interesa dobivena na jedinkama tijekom istraživanja nazivamo podacima. Cilj je statistike pomoći istraživačima i praktičarima u organizaciji, analizi i interpretaciji rezultata analize podataka.

#### PITANJA ZA PONAVLJANJE:

- 1) Ukoliko vrijednosti mjerene na jednom uzorku ukazuju na neko svojstvo u drugom uzorku takav uzorak nazivamo:
- 2) Definirajte kojem tipu varijabli pripadaju slijedeća obilježja:
  - broj članova obitelji
  - gospodarska djelatnost
  - status studenta
  - temperatura u stupnjevima C
  - brojevi na dresovima nogometnika
  - starost
  - stil u plivanju
  - ocjena na testu
  - radni staž
  - bračni status
  - nacionalnost
  - školska spremna
  - duljina skoka
  - mjesto boravka
  - poređak u utrci
  - temperatura u stupnjevima K

## 4.2 Deskriptivna vs. inferencijalna statistika

Statističke metode povezane su s dva tipa istraživanja:

- Opis skupa podataka - opisna ili deskriptivna statistika
- Izbor uzorka iz većeg skupa i njihovo korištenje za zaključivanje o čitavom skupu. Takvi zaključci nazivaju se statistički zaključci.
- Deskriptivna statistika

Deskriptivna statistika buhvata postupke uređivanja, tabličnog i grafičkog prikazivanja podataka, te izračunavanje opisnih statističkih pokazatelja.

- Inferencijalna statistika

Inferencijalna - temelji se na teoriji vjerojatnosti i proučava metode kojima se pomoću dijela informacija (reprezentativni uzorak) donose zaključci o cijelini (populacija).

Veoma često želimo donijeti zaključke o veoma velikim skupovima, populacijama, koji su veoma često i neprebrojni. Jedan od načina je uzimanje adekvatnog slučajnog uzorka te donošenje adekvatnih statističkih zaključaka o populaciji temeljem izabranog uzorka.



#### 4.2.1 Deskriptivna statistika

Metodama deskriptivne statistike opisujemo odabrani uzorak / populaciju što uključuje računanje statističkih pokazatelja (statistika / parametara) poput aritmetičke sredine (prosjek), maksimuma, minimuma i dr. Generalizacija zaključaka nije moguća.

Metode deskriptivne statistike služe za organizaciju, sumiranje i vizualizaciju podataka. Podatke organiziramo u obliku tablica i / ili grafova dok deskriptivni pokazatelji (npr. prosjek), služe za sumiranje podataka. Sumarni pokazatelj za cijelu populaciju naziva se parametar dok se sumarni pokazatelj uzorka naziva statistika, slika 50.

Analiza podataka veoma često završava samo deskriptivnom statistikom, bilo zbog nereprezentativnog uzorka ili nedovoljnog znanja praktičara.

Istraživači koriste deskriptivnu statistiku kako bi izvještavali o svom uzorku / populaciji. Sumirajući informacije, deskriptivna statistika ubrzava i pojednostavljuje proces razumijevanja karakteristika bilo populacije ili dijela populacije.

Tablice podataka (tabele) mogu biti jednostavne i prikazivati jedan statistički niz (jedno obilježje) i skupne tj. prikazivati dva ili više nizova (dva ili više obilježja). Kombinirane tablice ili tablice kontingencije - prikazuju jedan niz promatran prema dva ili više obilježja.

**4.2.1.1 Distribucija frekvencija jedne varijable** Distribucija frekvencija varijable daje uređeni prikaz varijable (njezinih uzlazno sortiranih vrijednosti); ovisi o tipu varijable.

##### Kvalitativna / diskretna kvantitativna varijabla:

Distribucija frekvencija kvalitativne (kategoriskske) ili diskrete kvantitativne varijable definirana je brojem (frekvencijom  $f$ ) jedinki po svakoj kategoriji / diskretnoj vrijednosti varijable.

- Suma frekvencija po svim pojedinim vrijednostima varijable jednaka je veličini uzorka  $N$ .
- Relativna frekvencija (proporcija) -  $p=f/N$
- Postotak ( $\% = p * 100$ )
- Suma svih proporcija iznosi 1
- Suma svih postotaka iznosi 100.

##### Kvantitativna / kontinuirana varijabla

Raspon vrijednosti kontinuirane varijable jednak je razlici između najveće i najmanje vrijednosti.

Raspon se dijeli na konačan broj intervala koji se dodiruju (ali ne preklapaju).

- Frekvencija ( $f$ ) je broj pojavljivanja jedinki u pojedinom intervalu
- Relativna frekvencija (proporcija) -  $p=f/N$
- Postotak ( $\% = p * 100$ )
- Suma svih proporcija iznosi 1
- Suma svih postotaka iznosi 100.

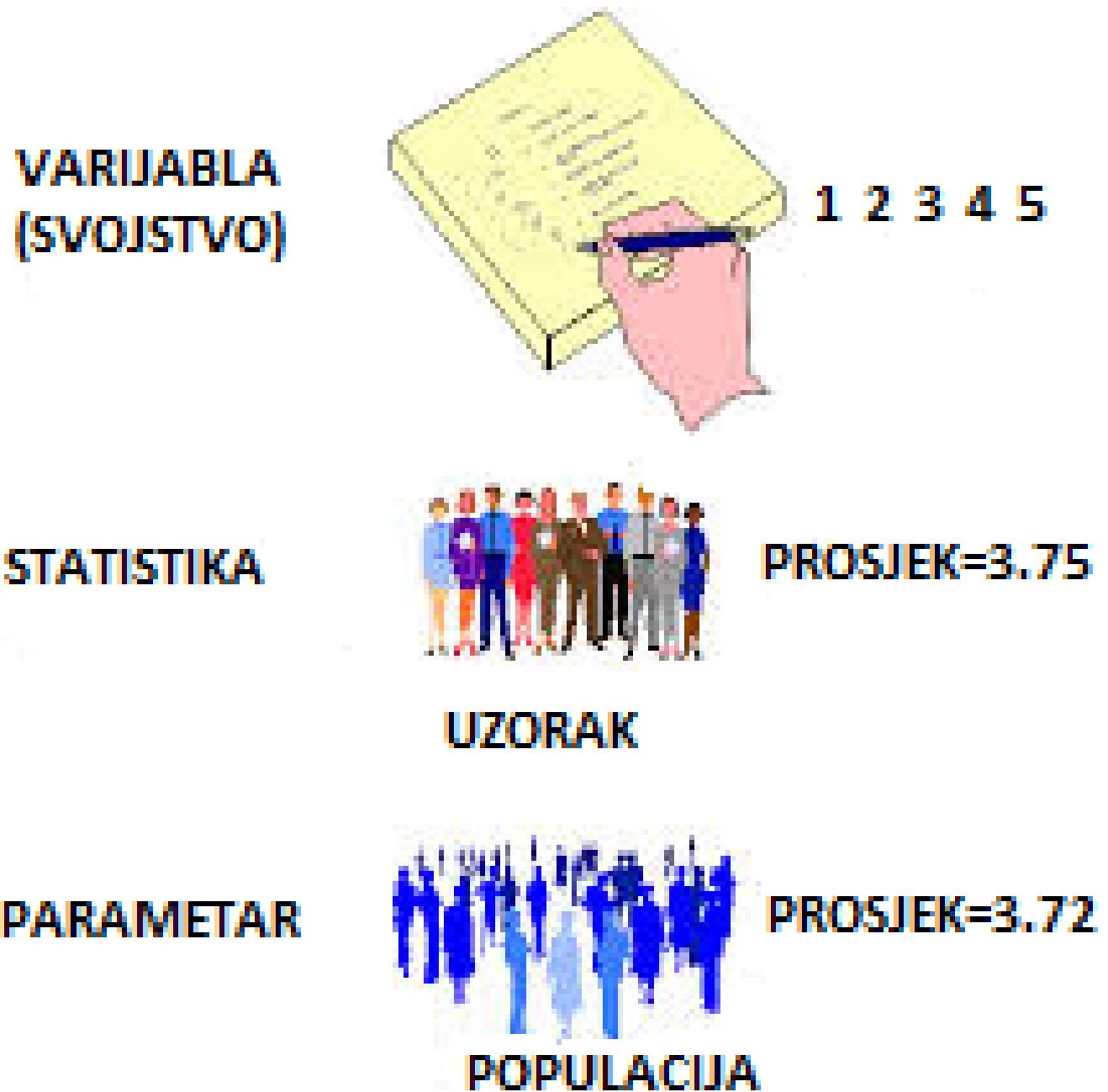
Distribucija frekvencija može se prikazati numerički i grafički.

Numerički prikaz distribucije frekvencija daje tablica frekvencija koja sadrži: frekvencije, proporcije, postotke i kumulativne postotke po pojedinačnim vrijednostima ili intervalima vrijednosti varijable (ovisno o tipu).

Primjer tablice frekvencije (pojedinačne vrijednosti) dan je u tablici 51.

Primjer tablice frekvencije (intervali vrijednosti) dan je u tablici 52.





Slika 50: Parametar populacije nasuprot statistike uzorka

REZULTAT	FREKVENCIJA	PROPORCIJA	POSTOTAK	KUMULATIVNI POSTOTAK
0	2	0,02	1,90	1,90
2	1	0,01	0,95	2,86
3	6	0,06	5,71	8,57
4	6	0,06	5,71	14,29
5	6	0,06	5,71	20,00
6	12	0,11	11,43	31,43
7	14	0,13	13,33	44,76
8	16	0,15	15,24	60,00
9	8	0,08	7,62	67,62
10	34	0,32	32,38	100,00
TOTAL	105	1,00	100,00	

Slika 51: Primjer tablice frekvencije

FREQ	IQ_klasa	POSTOTAK	KUM_POSTOTAK
8	[80,100)	30,77	30,77
11	[100,120)	42,31	73,08
5	[120,140)	19,23	92,31
1	[140,160)	3,85	96,15
1	[160,180)	3,85	100,00

Slika 52: Primjer tablice frekvencije, intervali vrijednosti



## 4.2.2 Grafičke metode za prikaz kvantitativnih varijabli

U nastavku su dani najčešći grafički prikazi gustoće/frekvencije jedne slučajne varijable.

**4.2.2.1 Grafički prikazi distribucije frekvencije jedne varijable** Kako bismo se upoznali s najčešćim grafičkim prikazima učitat koristit ćemo skup podataka o izmišljenoj djeci iz dva razreda 2 koji nosi informacije o pripadnosti razredu, visini i spolu.

Tablica 2: Podaci na kojima ćemo se upoznati s grafičkim prikazima frekvencije jedne varijable

IQ	RAZRED	VISINA	SPOL
107	A	170	Ž
100	A	168	M
101	A	149	Ž
110	A	156	M
99	A	176	M
111	A	158	M
110	A	168	Ž
108	A	146	Ž
104	A	151	Ž
102	A	154	Ž

**4.2.2.1.1 Histogram** Histogram je najrašireniji oblik prikaza kontinuiranih kvantitativnih varijabli. Histogram predstavlja raspodjelu frekvencija nekog skupa podataka nakon što se podaci svrstaju u intervalne klase ili razrede (engl. *bin*). Histogramskim prikazom možemo prikazati frekvencije i relativne frekvencije.

Histogramski prikaz radimo na način da na os x nanosimo intervale vrijednosti, a na os y frekvencije (ili relativne frekvencije). Visina stupca je broj jedinki u pojedinom intervalu. Obratite pažnju na sliku 53 kako izgled histograma varira ovisno o širini intervala (izabranoj podjeli).

Za izradu Unutar sustava R funkcija *hist()*, funkcija *histogram()* paketa *lattice*.

Prije primjera crtanja histogramskog prikaza jedne kontinuirane varijable upoznajmo se s funkcijom *hist()* te strukturu nastalog objekta koji crtamo. Proučite razlike između prikaza na slici 53.

Prema želji/potrebi, prilikom izrade histograma moguće je prilagoditi broj klasa ili napraviti specifične intervale korištenjem argumenta *breaks*. Bilo koji dio objekta nastalog provođenjem funkcije *hist()* (rezultat objekt klase *list*) moguće je izlučiti kao zaseban objekat.

Histogramski prikazi su neizbjegni u prikazima usporedbi uzorka s teoretskim distribucijama, primjera na slici 54 prikazani su podaci o IQ učenika dva razreda kao relativne frekvencije standardiziranih vrijednosti sa superponiranim standardnom normalnom distribucijom.

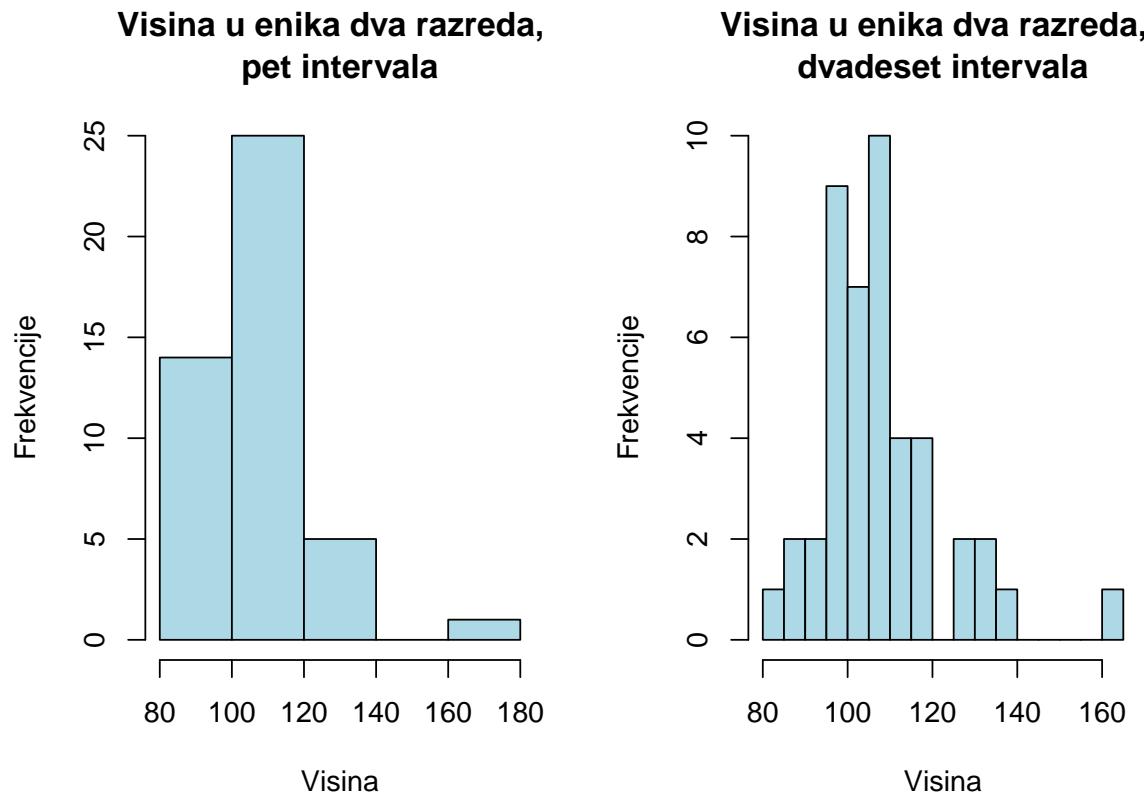
**4.2.2.1.2 Stupčasti dijagram** Adekvatan grafički prikaz za skup mjerena koji se sastoji od diskretnih vrijednosti merenja ili kategorijalnih podataka je stupčasti dijagram (engl. *barplot*). Crtamo objekt koji dobijemo funkcijom *table()* kojom prebrojimo pojavljivanje u kategorijama. Primijetite da su stupci u prikazu međusobno odvojeni, za razliku od histogramskog prikaza, iako se i na histogramskom prikazu intervali, klase, mogu dodatno razmaknuti prema želji. Tada histogramski prikaz veoma sliči stupčastom dijagramu.

U sljedećem primjeru korištenjem funkcije *table()* možemo izračunati frekvenciju svake pojedine zabilježene vrijednosti varijable te potom dobivene frekvencije nacrtati.

```
#frekvencije pojavljivanja pojedine vrijednosti u skupu podataka
kategorije <- table(razredi$IQ)
kategorije
```

80	87	89	93	96	97	98	99	100	101	102	103	104	105	106	107	108	109
1	1	1	2	1	1	3	2	2	1	1	2	2	1	1	1	1	3
110	111	112	115	116	118	119	120	127	128	131	140	162					





Slika 53: Dva histogramska prikaza jednakog skupa podataka, različit broj intervalnih klasa

```
4 2 1 1 1 1 1 1 1 2 1 1
```

Ponekad nije moguće napraviti, ili ne želimo napraviti, frekvencije svake pojedine izmjerene vrijednosti kategoriske varijable već i u ovome slučaju raspon izmjerenih vrijednosti želimo klasificirati na manji broj nepreklapajućih klasa. To je moguće napraviti funkcijom `cut()` za što sljedi primjer. Jedan od primjera za ovakvu klasifikaciju zamislite da ste studentima dali ispit na kojem su mogli prikupiti neki broj bodova u rasponu od 0-100. Osim što možete vidjeti frekvencije svakog od mogućih 101 varijanti, korisno nam je dobivene bodove podijeliti u intervale za koje ćemo pridružiti ocjene koje su u rasponu od 1-5.

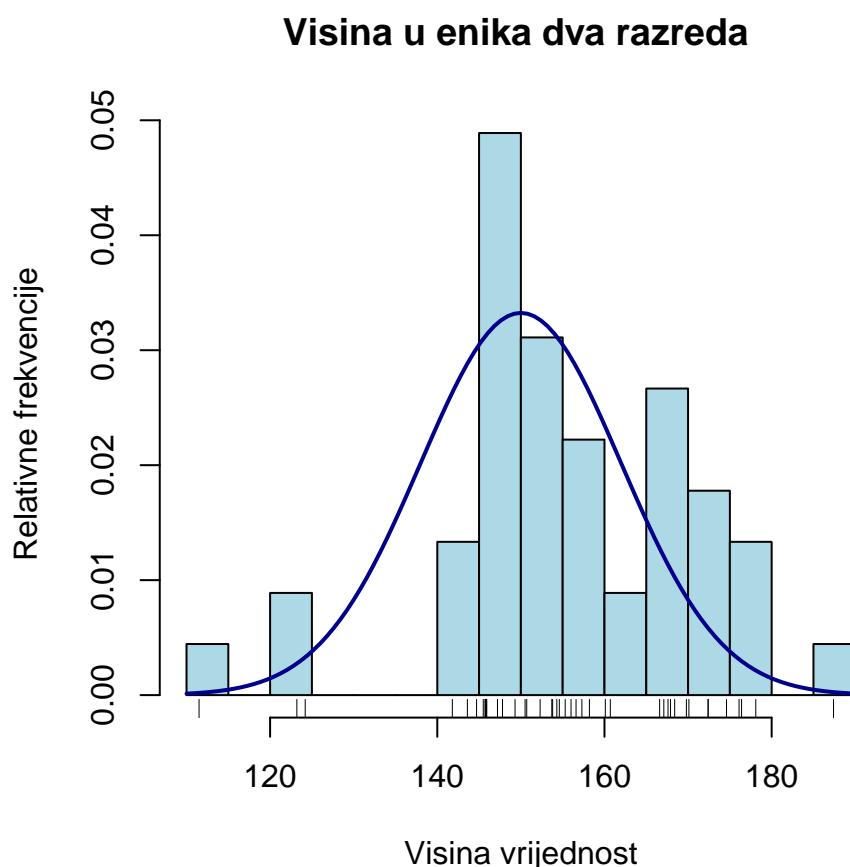
```
kategorije2 <- cut(razredi$IQ,
                     breaks=c(min(razredi$IQ), median(razredi$IQ), max(razredi$IQ)),
                     include.lowest=T)
kategorije2
```

```
[1] [80,107] [80,107] [80,107] (107,162] [80,107] (107,162] (107,162]
[8] (107,162] [80,107] [80,107] (107,162] (107,162] [80,107] (107,162]
[15] [80,107] (107,162] (107,162] (107,162] [80,107] [80,107] (107,162]
[22] [80,107] (107,162] (107,162] [80,107] [80,107] [80,107] (107,162]
[29] (107,162] (107,162] [80,107] (107,162] (107,162] [80,107] [80,107]
[36] [80,107] [80,107] (107,162] (107,162] (107,162] [80,107] [80,107]
[43] [80,107] (107,162] [80,107]
Levels: [80,107] (107,162]
```

Dodatno, veoma često želimo neki skup vrijednosti klasificirati relativno u odnosu na cijelokupan skup. U primjeru koji slijedi raspon varijable koja nosi informaciju o visini učenika razdijelit ćemo u kategorije u ovisnosti o mjerama deskriptivne statistike cijelokupnog skupa podataka, rezultata funkcije `summary()`.

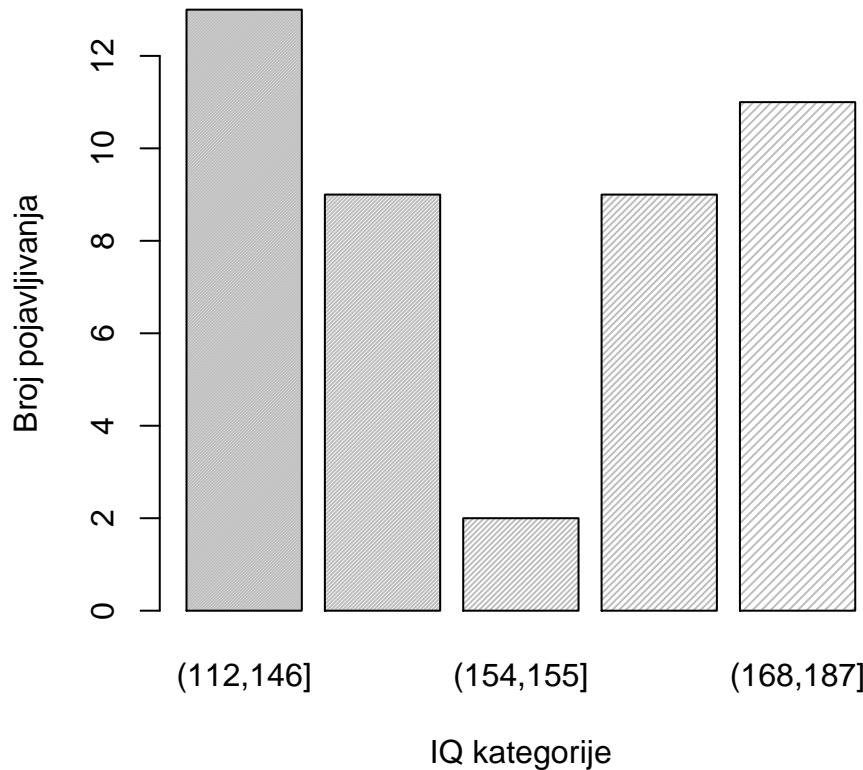
```
#funkcija cut za priprejanje pojedinoj kategoriji, interval
kategorije3 <- cut(razredi$VISINA, summary(razredi$VISINA))
#ispis
kategorije3
```





Slika 54: Histogramskih prikaz, usporedba podataka sa standardnom normalnom

## Stupasti dijagram, table



Slika 55: Barplot, kategorijski podaci

```
[1] (168,187] (168,187] (146,154] (155,168] (168,187] (155,168] (155,168]
[8] (112,146] (146,154] (146,154] (155,168] (146,154] (112,146] <NA>
[15] (168,187] (112,146] (168,187] (112,146] (154,155] (168,187] (112,146]
[22] (112,146] (146,154] (168,187] (146,154] (112,146] (146,154] (168,187]
[29] (112,146] (155,168] (155,168] (168,187] (146,154] (154,155] (155,168]
[36] (112,146] (112,146] (155,168] (168,187] (112,146] (112,146] (146,154]
[43] (112,146] (155,168] (168,187]
Levels: (112,146] (146,154] (154,155] (155,168] (168,187]
```

Primjer stupčastog dijagrama, slika ??.

Unutar sustava R funkcija `barplot()` iz paketa `base` ili funkcija `bwplot()` paketa `lattice`.

**4.2.2.1.3 Strukturni dijagram (engl. pie)** Strukturni dijagrami, tzv. pite, veoma su su popularni, ali ih mnogi analitičari/statističari ne smatraju pogodnim za prezentaciju podataka. Jedan od razloga je mogućnost manipuliranja učincima korištenjem boja, početnim kutom, sjenčanjem ili 3D vizualiziranjem strukturnog dijagrama.

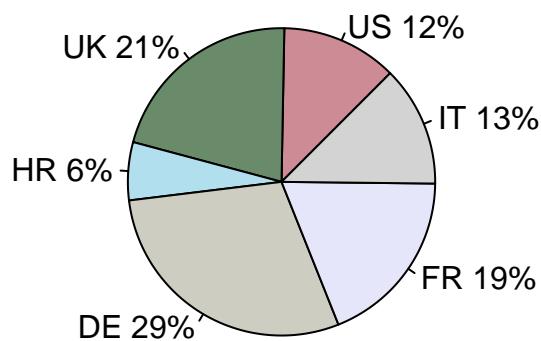
Strukturni dijagram unutar sustava R možemo izraditi funkcijom `pie()`.

**4.2.2.1.4 Graf stabljika - list (engl. stem and leaf)** Grafički prikaz *stabljika - list* unutar sustava R možemo napraviti funkcijom `stem()`.

Svako opažanje podijeli se na dva dijela (stabljiku i list). Stabljika su višedecimalne znamenke, a list niše decimalne znamenke. Stabljike se ispišu u stupac i poredaju po veličini od najmanje do najveće. Odgovarajući list svakog opažanja se napiše u redak kod odgovarajuće stabljike, slika ??.



## Strukturni dijagram podataka



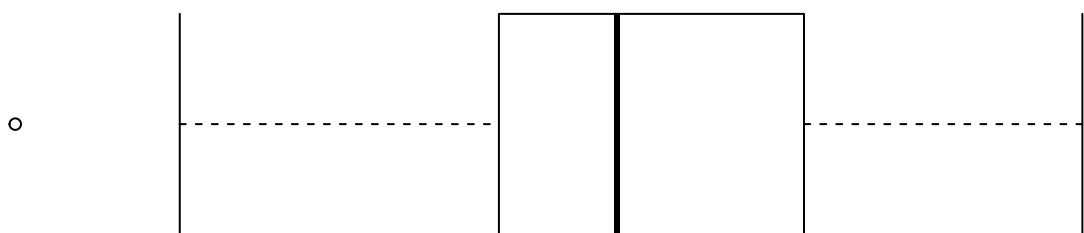
Slika 56: Strukturni dijagram

The decimal point is 1 digit(s) to the right of the |

```
8 | 079
9 | 336788899
10 | 001233445678999
11 | 00001125689
12 | 078
13 | 11
14 | 0
15 |
16 | 2
```

## Boxplot podataka o visinama u enika u dva razreda

min	25 per	median	75 per	max
111.5	145.9	154.3	167.6	187.4



Slika 57: Box-Whisker graf s označenim važnim vrijednostima

**4.2.2.1.5 Grafički prikaz kutija i brk (engl. *Box-Whisker*)** Na ovom izuzetno popularnom i čestom grafičkom prikazu imamo gotove sve mjere deskriptivne statistike na jednom grafu i možemo reći da prikazuje distribuciju kvantitativne varijable, slika 58. Ovo je, vjerojatno, najčešći način prikaza sumarnih mjera distribucije jedne varijable koji ćete pronaći u knjigama i/ili znanstvenim radovima. Funkcija unutar sustava R kojom radimo ovu vrstu grafičkog prikaza je *boxplot()* iz paketa *base* ili funkcija *bwplot()* iz paketa *lattice*.

Prisjetimo se podataka o IQ učenika dva razreda na kojima ćemo, takošer, izraditi, Box\_Whisker prikaze:

```
#sumarna statistika varijable
summary(razredi$IQ)

Min. 1st Qu. Median Mean 3rd Qu. Max.
80      99     107    108    112    162

#sumarna statistika varijableIQ za svaki razred
tapply(razredi$IQ, razredi$RAZRED, summary)

$A
Min. 1st Qu. Median Mean 3rd Qu. Max.
89      100    108    108    111    140

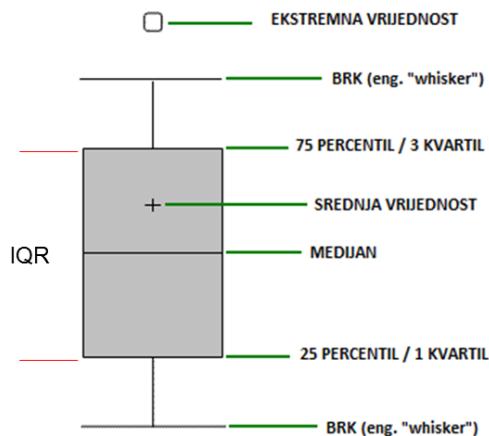
$B
Min. 1st Qu. Median Mean 3rd Qu. Max.
80.0    98.5   105.0 108.3 114.0 162.0

par(mfrow=c(3,1), mar=c(0.7,0.7,0.7,0.7))
#nacrtamo boxlot svih podataka
boxplot(razredi$IQ, horizontal=F)

#priprema labela
sumarna_statistika <- summary(razredi$IQ)

#stavljamo željeni tekst na zadane koordinate
text(rep(1.35,5), sumarna_statistika,
  labels=c('minimum', '1st Qu.', 'median', 'mean', '3rd Qu.', 'maximum'),
  cex=.9)
```

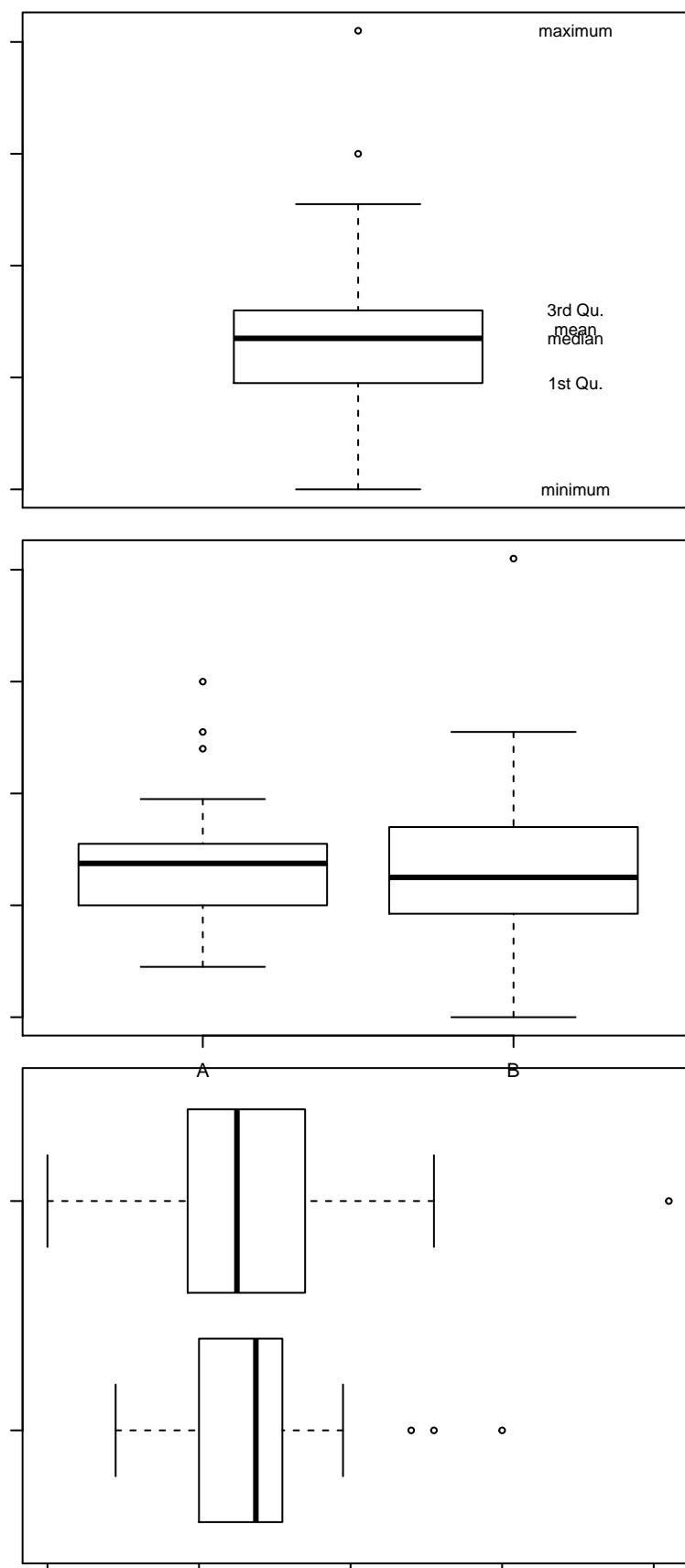




Slika 58: Shema Box-Whisker grafičkog prikaza

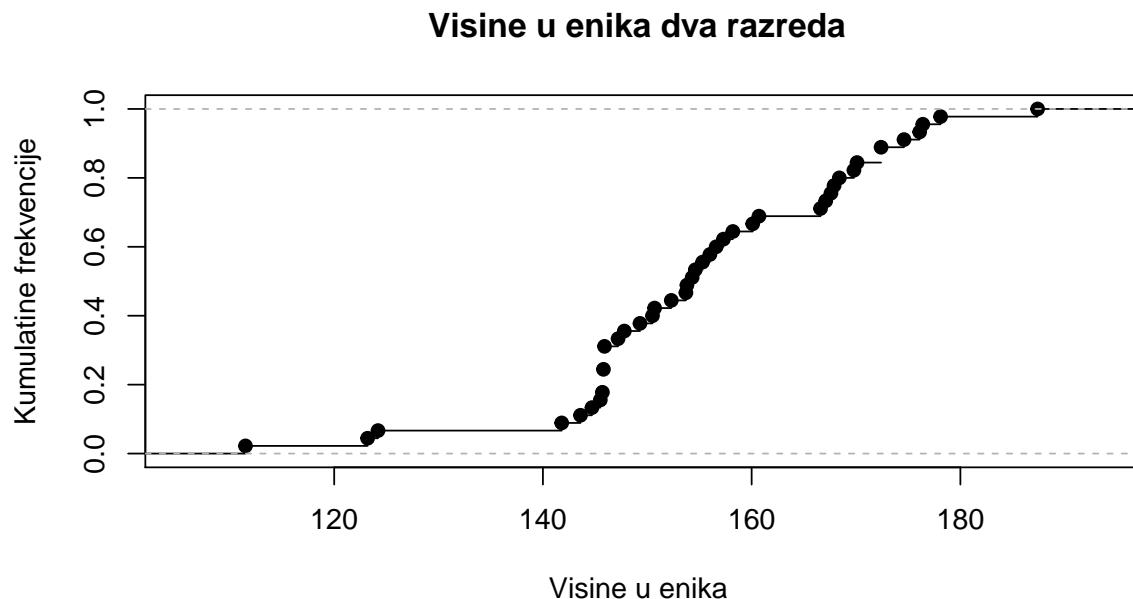
```
#boxplot prikaz za svaki razred posebno  
boxplot(razredi$IQ ~ razredi$RAZRED, horizontal=F)  
  
#horizontalni boxplot prikaz za svaki razred posebno  
boxplot(razredi$IQ ~ razredi$RAZRED, horizontal=T)
```





Slika 59: Box-plot prikaz a) cijelokupne varijable, b) boxplot prikaz za svaki razred posebno , c) horizontalni boxplot prikaz za svaki razred posebno





Slika 60: Kumulativna funkcija distribucije

**4.2.2.1.6 Kumulativne relativne frekvencije** Za razliku od teoretskih cumulativnih funkcija distribucije (**p** funkcija), ponekad želimo nacrtati empirijske kumulativne frekvencije

#### 4.2.3 Deskriptivna statistika - numeričko sumiranje podataka

Sumiranje podataka o nekom skupu (populaciji / uzorku) možemo podijeliti na mjere:

**centralne tendencije podataka** kao što su

- srednja vrijednost - sredina - prosjek
- medijan
- mod i

**varijabilnosti podataka** kao što su

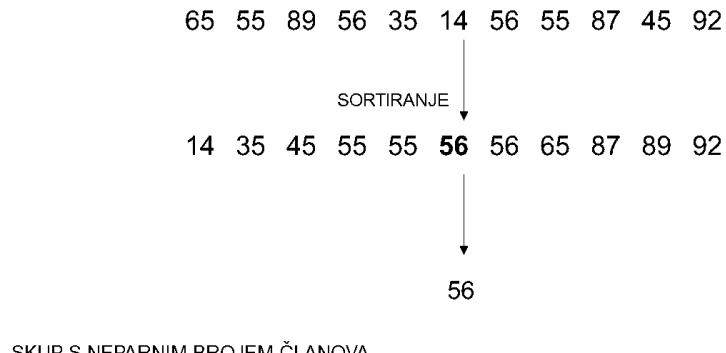
- apsolutna varijabilnost
- standardna devijacija
- varijanca
- raspon
- interkvartilni raspon
- relativna varijabilnost
- koeficijent varijacije i

**mjere relativnog položaja** kao što su

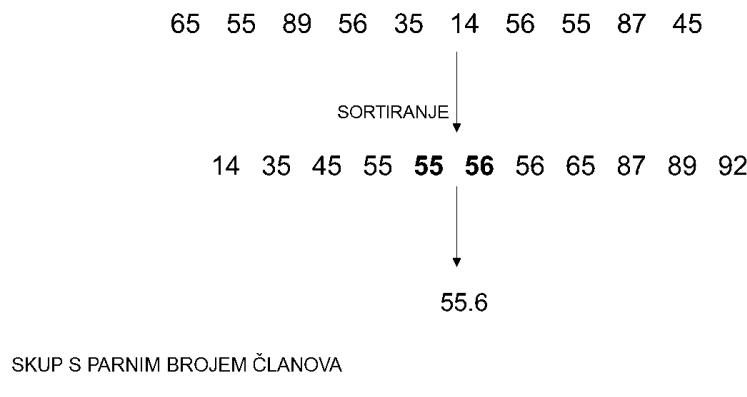
- percentili
- z-vrijednost.

Unutar sustava R postoji više funkcija kojima možemo dobiti sumarne statistike jedne varijable; unutar osnovnog paketa **base** to je funkcija **summary()** ili funkcija **by()** kada želimo sumarnu statistiku za jednu varijablu po kategorijama neke druge varijable.





Slika 61: Određivanje medijana na skupu s neparnim brojem članova



Slika 62: Određivanje medijana na skupu s parnim brojem članova

**4.2.3.1 Mjere centralne tendencije** Prosjek ili aritmetička sredina je srednja vrijednost varijable. Funkcija kojom je računamo u R-u je `mean()`.

Prosjek slučajne varijable još se naziva i njezinim očekivanjem koje se označava

$$\mathbb{E}X$$

$$\bar{x} = \frac{\sum x_i}{n}$$

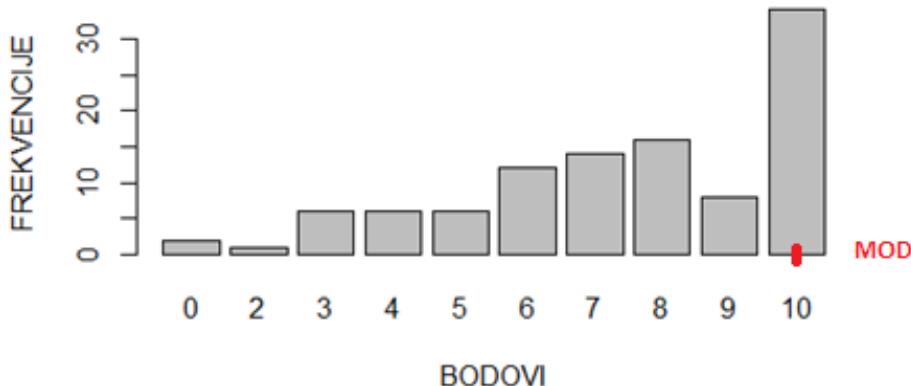
**4.2.3.2 Medijan** Medijan je ona vrijednost varijable koja ukupni broj vrijednosti varijable dijeli na dva jednako velika dijela nakon što smo vrijednosti varijable sortirali (rangirali). Određuje se u ovisnosti o broju vrijednosti varijable koji može biti paran ili neparan.

Ukoliko računamo medijan na skupu s neparnim brojem članova:

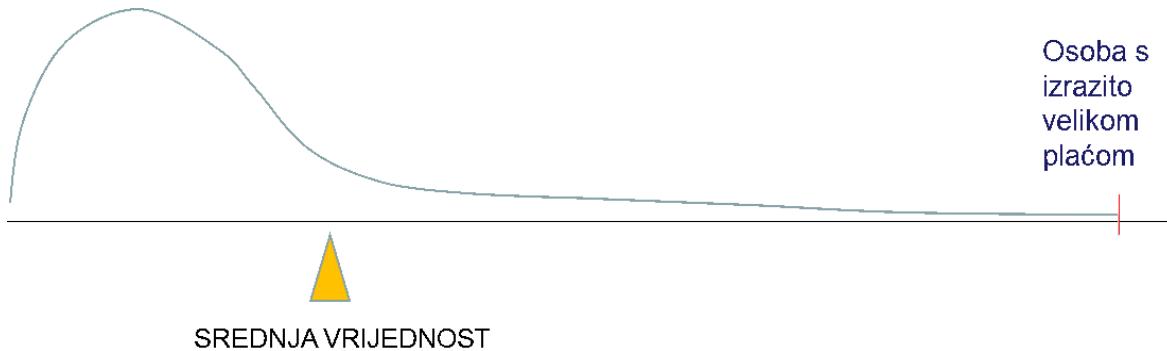
Primjer računanja medijana na skupu s parnim brojem članova:



### Barplot - diskretna varijabla



Slika 63: Mod - približno određivanje grafičkim prikazom



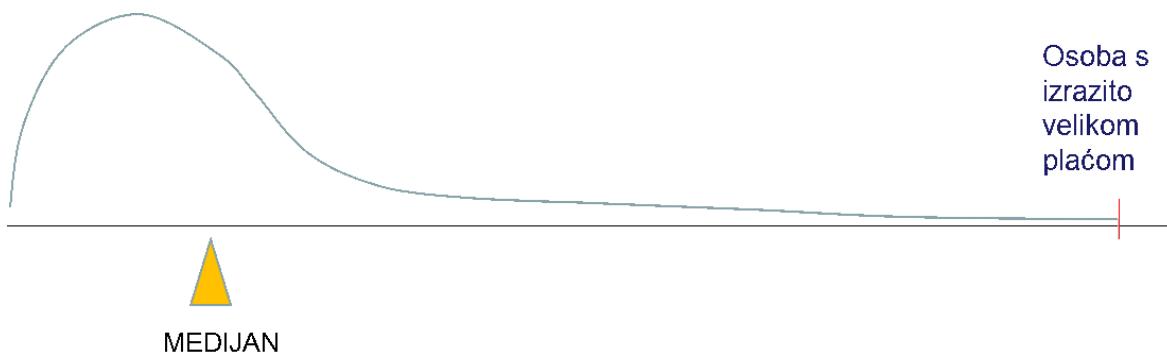
Slika 64: Neke karakteristike mjera centralne tendencije, prvi dio

**4.2.3.3 Mod** Mod je najčešća vrijednost u skupu podataka ili drugačije rečeno ona vrijednost u skupu koja ima najveću frekvenciju, slika 63.

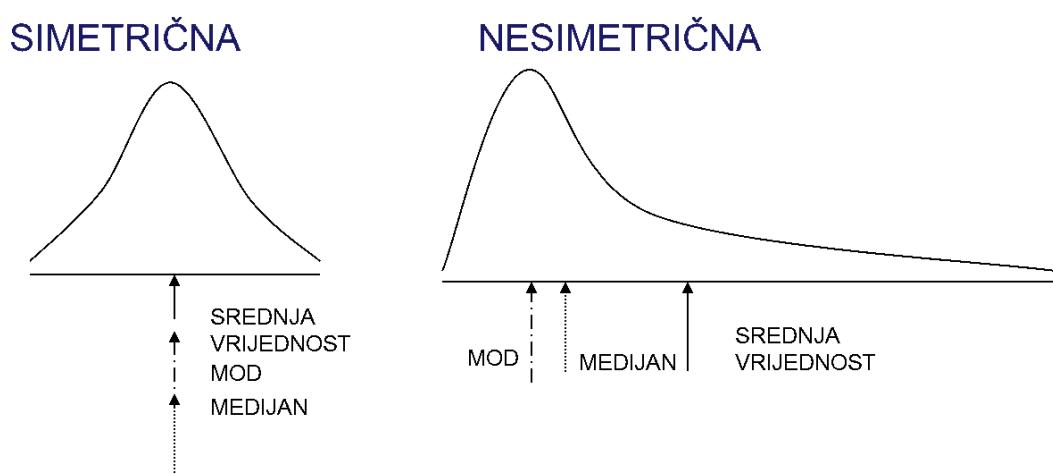
Neke karakteristike mjera centralne tendencije

Srednja vrijednost, aritmetička sredina veoma je osjetljiva na ekstremne (netipične) vrijednosti varijable (engl. *outlier*), proučite slike 64 i 65 PRIMJER: Distribucija plaća u državi:

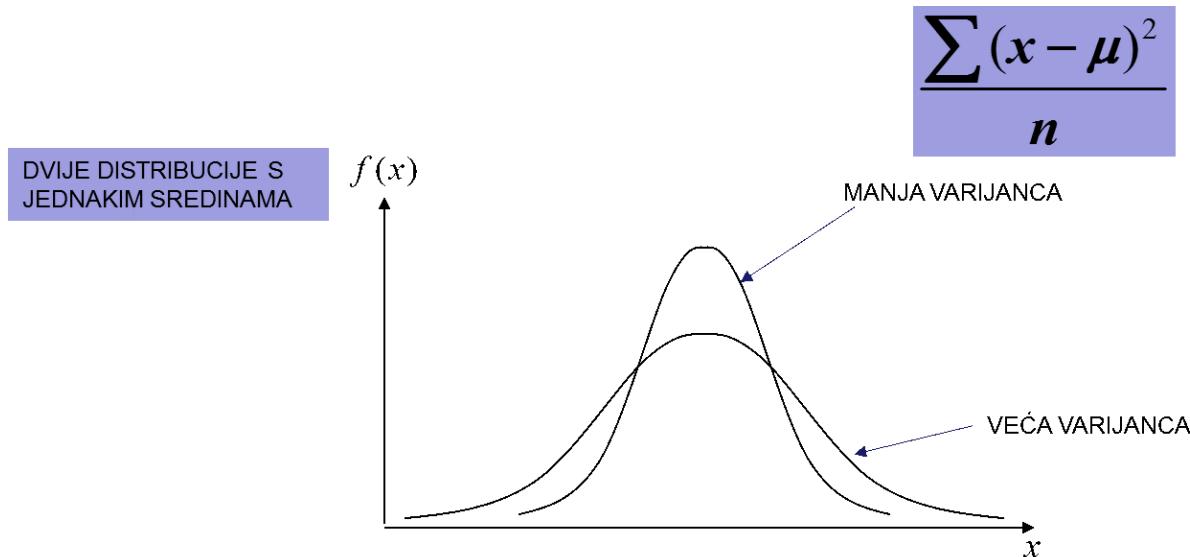
Medijan nije osjetljiv na ekstremne vrijednosti (engl. *outlier*) kao što je to bila srednja vrijednost. Proučite sliku 66 i prokomentirajte.



Slika 65: Neke karakteristike mjera centralne tendencije



Slika 66: Utjecaj simetričnosti distribucije na mjere centralne tendencije podataka



Slika 67: Primjer dvije normalne raspodjele jednake srednje vrijednosti ali različitih varijanci

#### 4.2.4 Mjere varijabilnosti podataka

**4.2.4.1 Devijacija** Devijacije se definira kao odstupanje pojedinačne vrijednosti varijable od prosjeka varijable.

$$d = \sum_{i=1}^n (x_i - \bar{x})$$

Svojstvo absolutne devijacije je to da je suma devijacija varijable uvijek 0.

**4.2.4.2 Varijanca** Varijanca je mjera raspršenja (disperzije) u skupu podataka i jedna je od najvažnijih mjera u statistici. Matematički je definirana kao prosječna kvadrirana udaljenost između svakog pojedinog mjerenja od prosjeka skupa ali drugačije rečeno prosječna kvadrirana devijacija od sredine.

Varijanca je prosječna suma kvadriranih devijacija, suma kvadriranih odstupanja od prosjeka varijable.

**4.2.4.2.1 Varijanca populacije nasuprot varijanci uzorka** Varijanca populacije odnosi se na vrijednost varijance koja se izračunava iz podataka o populaciji, a varijanca uzorka je varijanca izračunata iz uzorka. Zbog ove vrijednosti nazivnika u formuli za varijancu u slučaju podataka o uzorku je "n-1", a to je "N" za podatke o populaciji. Kao rezultat toga, i varijance i standardna devijacija izvedena iz podataka o uzorku su veće nego onih dobivenih na populaciji.

Kako je način izračuna standardne devijacije i varijance za populaciju (populacijski parametar) i uzorak (statistika na uzorku) je različit te se različito i označava.

VARIJANCA POPULACIJE:

$$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \mu)^2}{n}$$

VARIJANCA UZORAKA:

$$s^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}$$

Funkcija unutar sustava R kojom računamo varijancu uzorka je `var()`. Napominjemo da većina statističkih programa pretpostavlja izračune na uzorku te sumu dijeli s  $n - 1$ .

Izgled normalne distribucije (razdiobe) jednake sredine i različite varijance podataka.



RAZRED_1	RAZRED_2	MEAN_1	MEAN_2	DEV_1	DEV_2	SS_1	SS_2
102	127	110,54	110,23	-8,54	16,77	72,91	281,21
128	131	110,54	110,23	17,46	20,77	304,91	431,36
131	96	110,54	110,23	20,46	-14,23	418,67	202,51
98	80	110,54	110,23	-12,54	-30,23	157,21	913,90
140	93	110,54	110,23	29,46	-17,23	867,98	296,90
93	120	110,54	110,23	-17,54	9,77	307,60	95,44
110	109	110,54	110,23	-0,54	-1,23	0,29	1,51
115	162	110,54	110,23	4,46	51,77	19,91	2680,05
109	103	110,54	110,23	-1,54	-7,23	2,37	52,28
89	111	110,54	110,23	-21,54	0,77	463,91	0,59
106	109	110,54	110,23	-4,54	-1,23	20,60	1,51
119	87	110,54	110,23	8,46	-23,23	71,60	539,67
97	105	110,54	110,23	-13,54	-5,23	183,29	27,36
				0,00	0,00	2891,23	5524,31

Slika 68: Svojstvo devijacije sumiranjem daje nulu

Pogledajte tablicu \ref{tab:devijacija problem} i prokomentirajte radi čega nam mjera devijacije nije dobra mjera raspršenja te moramo kvadrirati udaljenosti od srednje vrijednosti?

**4.2.4.3 Standardna devijacija** Standardna devijacija računa se kao drugi korijen iz variance varijable. Ova mjera raspršenja vrijednosti varijable od prosjeka izražena je u istim jedinicama mjerena kao i varijabla pa ju je lakše interpretirati nego varijantu koja je izražena u kvadriranim jedinicama mjerena. Funkcija unutar sustava R kojom računamo standardnu devijaciju je `sd()` ili korištenjem dviju funkcija `sqrt(var())`.

Jednako kao i prilikom definiranja varianca, matematički se standardna devijacija populacije  $\sigma$  računa drugačije za populaciju i zorak. Za populaciju vrijedi:

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu)^2}{N}}$$

gdje je  $N$  veličina populacije, dok se na uzorku vrijedi:

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}}$$

gdje je  $n$  veličina uzorka.

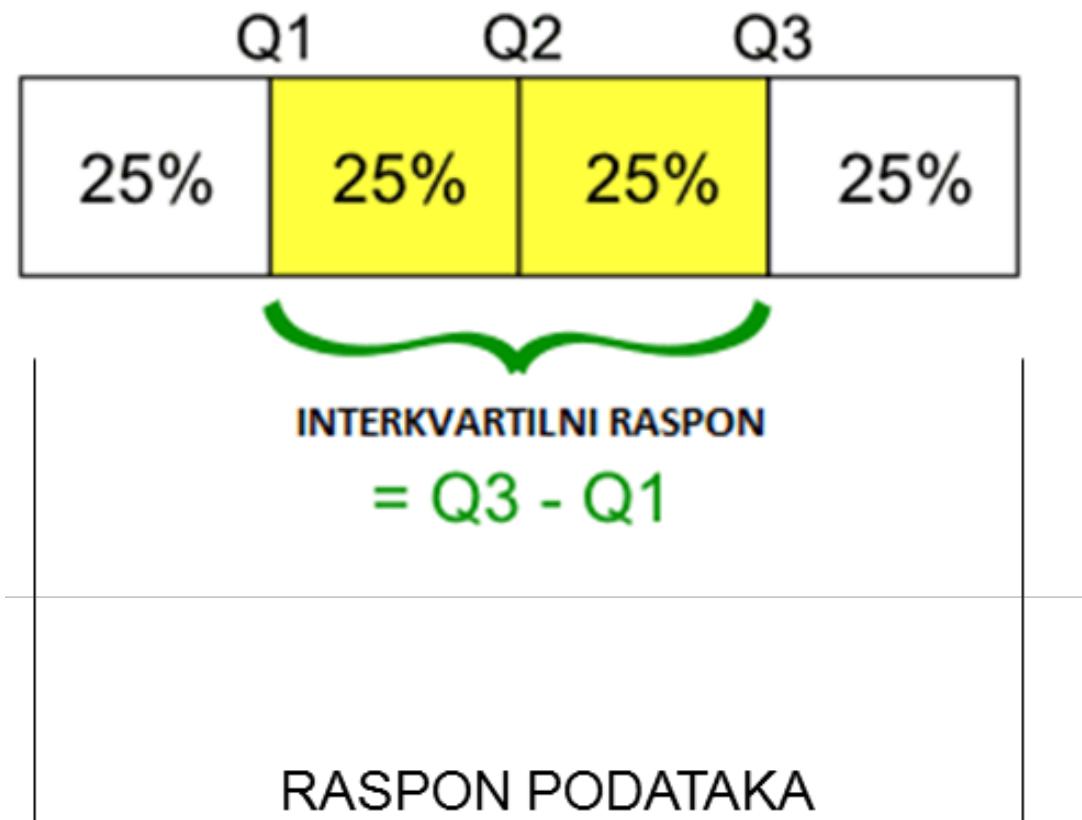
**4.2.4.4 Raspon podataka** Generalno raspon podataka je razlika između najveće i najmanje vrijednosti u skupu.

$$raspon = \text{minimum} - \text{maksimum}$$

**4.2.4.5 Interkvartilni raspon (IQR)** Kvartili dijele podatke u četiri jednakobrojna skupa.

Interkvartilni raspon (IQR)= Q3 - Q1 je razlika 3. kvartila (ispod kojeg leži 75% vrijednosti varijable) i 1. kvartila (ispod kojeg leži 25% vrijednosti varijable). Unutar IQR-a nalazi se 50% vrijednosti varijable. Unutar sustava R interkvartilni raspon podataka možemo dobiti korištenjem funkcije `IQR()`.





Slika 69: Shematski prikaz podataka prema kvartilima

$$IQR = \text{quantile}(x, 3/4) - \text{quantile}(x, 1/4)$$

**4.2.4.6 Koeficijent varijacije** Koeficijent varijacije je relativna mjera varijabilnosti izražena u postotcima. Ovakav način prikaza varijabilnosti često je lakša za poimanje od apsolutnih vrijednosti.

$$\text{Koeficijent varijacije (CV)} = \frac{\sigma}{\mu} * 100$$

**4.2.4.7 Mjere relativnog položaja** U mjere relativnog položaja spadaju percentili i z-vrijednosti.

Percentil,  $p$ , podatka je vrijednost opažanja  $p_i$ , takvog da je  $100p\%$  opažanja manje od  $y_i$  i  $100(1-p)\%$  opažanja veće od  $y_i$ .

#### PITANJA ZA PONAVLJANJE:

- 1) Dodajte naziv mjere deskriptivne statistike uz sljedeće definicije: srednja vrijednost opervacija koja rangirane vrijednosti od najmanje do najveće dijeli na dva jednako brojna skupa podataka; najčešće zabilježena vrijednost u skupu podataka; suma vrijednosti podijeljena s brojem opervacija.
- 2) Prilikom organizacije izmjerjenih podataka (dolje) korištena je \_\_\_\_\_ tablica. Nadopunite imena stupaca u tablici odgovarajućim riječima:



FREQ	IQ_klasa		
8	[80,100)	30,77	30,77
11	[100,120)	42,31	73,08
5	[120,140)	19,23	92,31
1	[140,160)	3,85	96,15
1	[160,180)	3,85	100,00

- 3) Izračunajte devijaciju te standardnu devijaciju za sljedeći skup podataka: 1: 3, 5, 7, 10, 10, 6.
- 4) Kreirajte rang varijablu sljedeće distribucije? (25, 17, 23, 23, 24, 25, 23) \_\_\_\_\_ Koja je tvrdnja o varijanci točna? (izaberite jednu tvrdnju): računa se kao suma kvadriranih devijacija svake opservacije od prosjeka varijable. računa se kao suma devijacija svake opservacije od prosjeka varijable. je drugi naziv za standardnu devijaciju. to je prosječna vrijednost zabilježene varijable.
- 5) Koji je mod sljedeće distribucije? (25, 17, 23, 23, 24, 25, 23) \_\_\_\_\_
- 6) Medijan je bolji pokazatelj centralne tendencije podataka od prosjeka kada je distribucija: simetrična asimetrična normalna bimodalna

U dijelu koji slijedi upoznat ćemo se kako dobiti/izračunati željene mjere deskriptivne statistike unutar sustava R na probnom skupu podataka.

```
#učitamo korisničke pakete s repozitorija CRAN koji su nam potrebni
library("xlsx")
#funkcija getwd nam daje informacije o trenutnom radnom direktoriju
getwd()
```

```
[1] "C:/Users/hd/PODACI/SRCE_razno/Moje_tecajevi/S720_upgrade"
```

Funkcija setwd postavlja radni direktorij prema našim specifikacijama.

```
setwd("~/S720")
```

Veoma korisna funkcija za ispis datoteka željenog direktorija i formata:

```
list.files(getwd(), pattern="xlsx")
```

```
[1] "dijametar.xlsx"           "dva_razreda_iq.xlsx"
[3] "dva_razreda_iq_long.xlsx" "p.xlsx"
[5] "par_vs_nepar.xlsx"       "strukture_podataka.xlsx"
```

Učitat ćemo podatke s kojima ćemo se često susretati tijekom tečaja. Riječ je o podacima o učenicima dva izmišljena razreda s informacijama o njihovoj propadnosti razredu, inteligenciji, visini i spolu.

```
#?read.xlsx
razredi<- read.xlsx("dva_razreda_iq_long.xlsx", as.data.frame=T, stringsAsFactors = F, sheetIndex=1)
```

Uvijek, prije početka bilo kakve analize, pogledajmo strukturu podataka.

```
str(razredi)
```

```
'data.frame': 45 obs. of 4 variables:
 $ IQ      : num 107 100 101 110 99 111 110 108 104 102 ...
 $ RAZRED: chr  "A" "A" "A" "A" ...
 $ VISINA: num  170 168 149 156 176 ...
 $ SPOL   : chr  "L" "M" "L" "M" ...
```



```
#pogledajmo imena varijabli
names(razredi)
```

```
[1] "IQ"      "RAZRED" "VISINA" "SPOL"
```

Kategoriske varijable u R-u nazivaju se faktori (eng. "factor"):

```
#definiramo varijablu RAZRED kao faktorsku varijablu
#primijetite da "recikliramo" staru varijablu koja je bila tipa "character"
razredi$RAZRED <- factor(razredi$RAZRED, levels=c("A", "B"))

#frekvencija pojavljivanja za svaku pojedinu vrijednost IQ
table (razredi$IQ)
```

80	87	89	93	96	97	98	99	100	101	102	103	104	105	106	107	108	109
1	1	1	2	1	1	3	2	2	1	1	2	2	1	1	1	1	3
110	111	112	115	116	118	119	120	127	128	131	140	162					
4	2	1	1	1	1	1	1	1	1	1	2	1	1				

Opis, deskripcija varijable:

```
#minimalna vrijednost
min(razredi$IQ)
```

```
[1] 80
```

```
#maksimalna vrijednost
max(razredi$IQ)
```

```
[1] 162
```

```
#sumarna statistika
summary((razredi$IQ))
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
80	99	107	108	112	162

Upoznat ćemo se kako doći do mjera centralne tendencije u sustavu R.

Medijan podatka:

```
median(razredi$IQ)
```

```
[1] 107
```

Srednja vrijednost

```
mean(razredi$IQ)
```

```
[1] 108
```

Varijanca:

```
#računa varijancu uzorka
var(razredi$IQ)
```

```
[1] 213
```

Standardna devijacija:

```
sd(razredi$IQ)
```

```
[1] 14.6
```

Raspon:

```
range(razredi$IQ)
```



```
[1] 80 162
```

Interkvartilni raspon:

```
IQR(razredi$IQ)
```

```
[1] 13
```

Željeni kvantili:

```
#predodređeni kvantili  
quantile(razredi$IQ)
```

```
0% 25% 50% 75% 100%  
80 99 107 112 162
```

```
#tražimo zadani percentil, 70percentil  
quantile(razredi$IQ, 0.73)
```

73%

111

Za statističku obradu podataka neobično je važno znati na pravilan način koristiti specifične R petlje. Jedna od najbitnijih `tapply()` kako smo već rekli, računa proizvoljnu zadanu funkciju na skupu podataka za svaki nivo neke druge faktorske varijable. Na primjeru koji slijedi na podacima o dva zamišljena razreda, dobit ćemo izračune za svaki razred. Zamislite da ste dobili tablicu s više desetaka tisuća redaka, rezultati ispita iz matematike na državnoj maturi te da želite izračunati sumarnu statistiku za svaki razred ili školi posebno.

```
#raspon podataka za svaki razred posebno  
tapply(razredi$IQ, razredi$RAZRED, range)
```

\$A

```
[1] 89 140
```

\$B

```
[1] 80 162
```

Funkcija koju primjenjujemo je proizvoljna funkcija iz sustava R ali i bilo koja funkcija koju je pripremio korisnika a odgovara podacima na kojih ih želimo primijeniti.

```
#interkvartilni raspon, svaki razred posebno  
tapply(razredi$IQ, razredi$RAZRED, IQR)
```

```
A      B  
10.5 15.5
```

Koeficijent varijacije:

Koeficijent varijacije je relativna mjera varijabilnosti jedne varijable izražena u postotkom. To je korisna vrijednost jer nam dopušta izravnu usporedbu varijabilnosti u uzorcima izmjer enim različitim jedinicama ili s vrlo različitim sredstvima. Na primjer, s koeficijentima varijacije možemo usporediti relativnu varijabilnost duljine i visine. Ponekad se još naziva i relativna standardna devijacija (RSD), a definirana je kao standardna devijacija podijeljena s prosjekom:

$$c_v = \frac{\sigma}{\mu}$$

Unutar sustava R koeficinet varijacije možemo izračunati funkcijom `cv()` paketa `raster` ili napraviti korisničku funkciju kako je prikazano u primjeru:

```
library(raster)
```

```
Warning: package 'raster' was built under R version 3.4.3
```

```
Loading required package: sp
```

```
Warning: package 'sp' was built under R version 3.4.3
```



```
x<- c(27.75, 24.5, 25.5, 26, 25, 27.75, 26.5, 27, 26.75, 26.75, 27.5)

#zamislite samo promjenu mjernih jedinica
x2 <- x*10

#promjer glave djeteta
y <- c(17.5, 17.1, 17.1, 17.3, 16.9, 17.6, 17.3, 17.5, 17.3, 17.5, 17.5)
```

Prokomentirajte dobivene koeficijente varijacije.

```
cv(x)
```

```
[1] 4.14
```

```
cv(x2)
```

```
[1] 4.14
```

```
cv(y)
```

```
[1] 1.27
```

Koja varijabla jače varira?

Koeficijent varijacije možemo izračunati i jednostavnom korisničkom funkcijom:

```
#korisnička funkcija
CV <- function(mean, sd){
  (sd/mean)*100
}
CV(mean(x), sd(y))
```

```
[1] 0.83
```

```
CV(mean(y), sd(x))
```

```
[1] 6.32
```

#### ZADACI ZA SAMOSTALAN RAD:

```
# ZADATAK 1): Pripremite podatke i izradite struktirni dijagram,  
#popularnu putu za varijablu IQ.
```

```
# ZADATAK 2): Izradi ne istom skupu podataka graf stabljkike i lista.
```

```
#ZADATAK 3): Napravite objekt imena naša_raspodjela i u njega unesite relativne  
#frekvencije svake vrijednosti varijable IQ koristeći funkciju hist().
```

```
#ZADATAK 4): Izradite stupčasti dijagram varijable IQ na način da vrijednosti  
#varijable kategorizirate prema mjerama deskriptivne statistike koristeći funkciju  
#cut() na način da prvi interval bude od minimuma do 25 percentila, drugi od 26-75  
#percentila te treći od 76 - maksimuma.
```

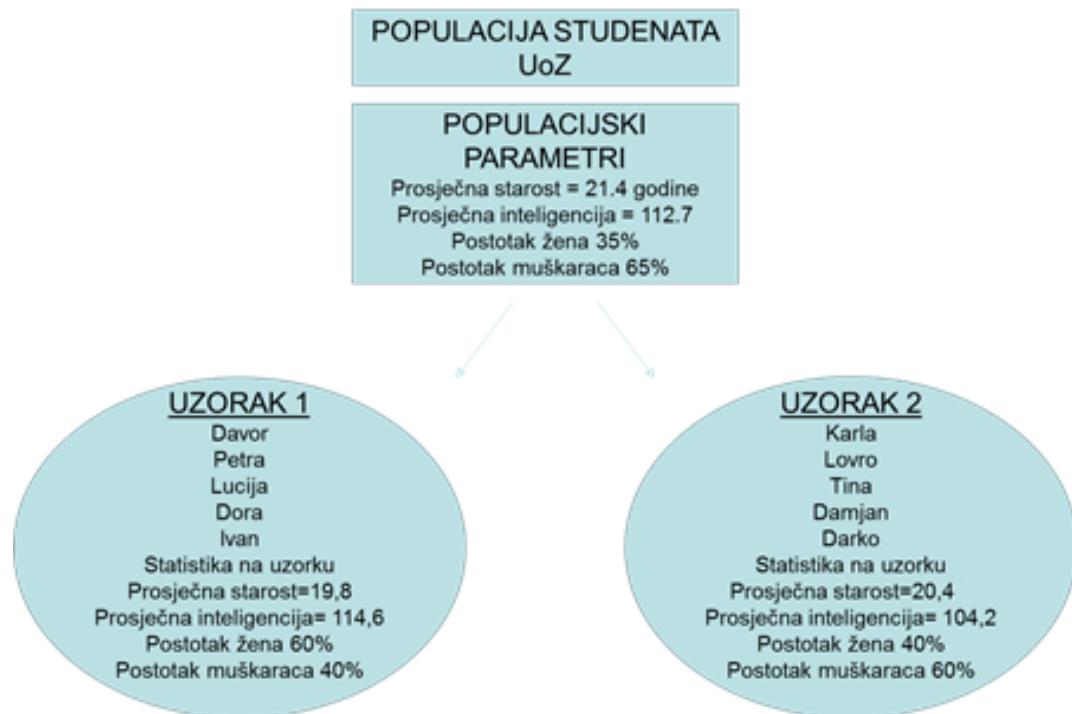
#Inferencijalna statistika

Inferencijalna statistika uključuje metode kojima na osnovu sumarnih mjera dobivenih na reprezentativnom uzorku donosimo zaključke o populaciji. Budući je uzorak kojim raspolažemo samo dio populacije, on nam daje ograničenu informaciju o populaciji. Statistika izračunata na uzorku radi toga nesavršeno procjenjuje populacijski parametar. Proučite sliku 70 i prokomentirajte.

### 4.3 Slučajna varijabla

Slučajna varijabla je varijabla čije vrijednosti variraju na slučajan način. Slučajna varijabla može poprimiti skup mogućih različitih vrijednosti, svaku s pridruženom vjerojatnošću.





Slika 70: Uzorkovanje iz populacije, primjer dva slučajna uzorka i statistika izračunatih iz njih

Pojam vjerojatnosti vezan je uz pojam slučajnog eksperimenta. Eksperiment je proces opažanja ili mjerena, čiji ishod tek treba odrediti; slučajan znači da je ishod eksperimenta neizvjestan.

Ishodi slučajnog eksperimenta mogu biti jednostavni i / ili složeni. Jednostavnii ihodi se ne mogu dekomponirati, a kombinacijom jednostavnih ishoda dobivaju se svi mogući ishodi slučajnog eksperimenta.

Uz svaki ishod slučajnog eksperimenta vezana je šansa (vjerojatnost) pojavljivanja tog ishoda.

Vjerojatnost mora zadovoljavati sljedeća svojstva:

- Vjerojatnost ( $P$ ) određenog jednostavnog ishoda je broj u intervalu  $[0, 1]$  koji mjeri šansu da ishod slučajnog eksperimenta bude upravo taj određeni ishod
- Suma vjerojatnosti svih jednostavnih ishoda slučajnog eksperimenta je 1
- Vjerojatnost nemogućeg ishoda je 0
- Vjerojatnost sigurnog ishoda je 1.

U analizi podataka, obično, moramo se ograničiti na ispitivanje (reprezentativnog) uzorka ove skupine i procjenu svojstava populacije iz ovog uzorka.

Slučajna varijabla može biti:

**Diskretna** - varijabla koja poprima konačan broj vrijednosti ili prebrojivo mnogo njih.

Primjer diskretne slučajne varijable: živ/mrtav, bacanje kockice, ocjena u školi.

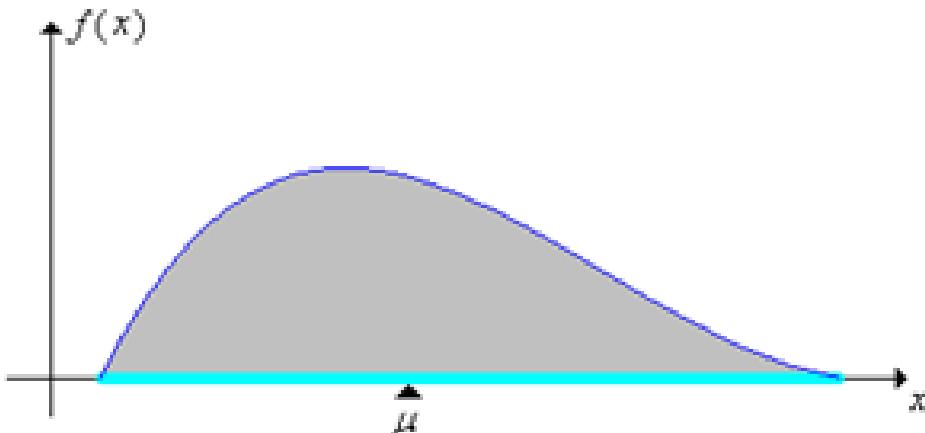
**Kontinuirana** - varijabla koja poprima bilo koju vrijednost iz nekog intervala realnih vrijednosti.

Primjer kontinuirane slučajne varijable: Temperatura, krvni tlak, masa, brzina, realni brojevi od 1-25.

Uz svaku slučajnu varijablu vezene su dvije funkcije: **funkcija vjerojatnosti** i **kumulativna funkcija distribucije** ako se radi o **diskretnoj varijabli**, odnosno **funkcija gustoće vjerojatnosti** i **kumulativna funkcija distribucije** ako se radi o kontinuiranoj varijabli.

**Kumulativna funkcija distribucije (CDF)** slučajne varijable  $X$  označava se s  $F(x)$ , pri čemu je  $F(x) = P(x \leq X)$ , za  $-\infty < x < \infty$  ili  $a \leq x \leq b$ . Funkcija nam daje vjerojatnost da varijabla  $X$  poprimi bilo koju vrijednost manju ili jednaku od  $x$ . Funkcija unutar sustava R koja daje vjerojatnost  $p$  za dani  $x$  je  $pnorm(x)$  za standardnu normalnu distribuciju i njoj inverzna  $qnorm(p)$  koja za dani  $p$  vraća kvantil  $x$ .





**MOD – MAKSIMUM FUNKCIJE.**

**MEDIJAN –  $P(X < x) \leq 1/2$**

**$P(X > x) \geq 1/2$**

Slika 71: Očekivanje kontinuirane slučajne varijable

Funkcija unutar sustava R koja crta empirijsku (iz uzorka) kumulativnu distribuciju frekvencija je funkcija `ecdf()`.

**Funkcija gustoće vjerojatnosti (PDF)** slučajne varijable X označava se s  $f(x)$  pri čemu se  $f(x)$  dobiva iz pripadne CDF.

Svojstva  $f(x)$ : -  $f(x) > 0$  za svaki  $x$ ,  $a \leq x \leq b$ , - ukupna površina (vjerojatnost) ispod funkcije  $f(x)$  između  $a$  i  $b$  iznosi 1.

#### 4.3.1 Kontinuirana slučajna varijabla

Kontinuirana slučajna varijabla može poprimiti neizmjerno beskonačan broj mogućih ishoda. Generalno, očekivanje (srednja vrijednost) slučajne varijable iznosi:

$$\mathbb{E}X = \int x_i p(x_i) dx$$

Proučite sliku 71 s označenim očekivanjem kontinuirane funkcije  $f(x)$ .

##### 4.3.1.1 Neke važne kontinuirane distribucije

**4.3.1.1.1 Normalna (Gaussova) distribucija** Najvažnija od svih teorijskih distribucija (u klasičnoj statistici). Funkcija je definirana na intervalu  $[-\infty, \infty]$ , a funkcija gustoće distribucije dana je formulom:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$

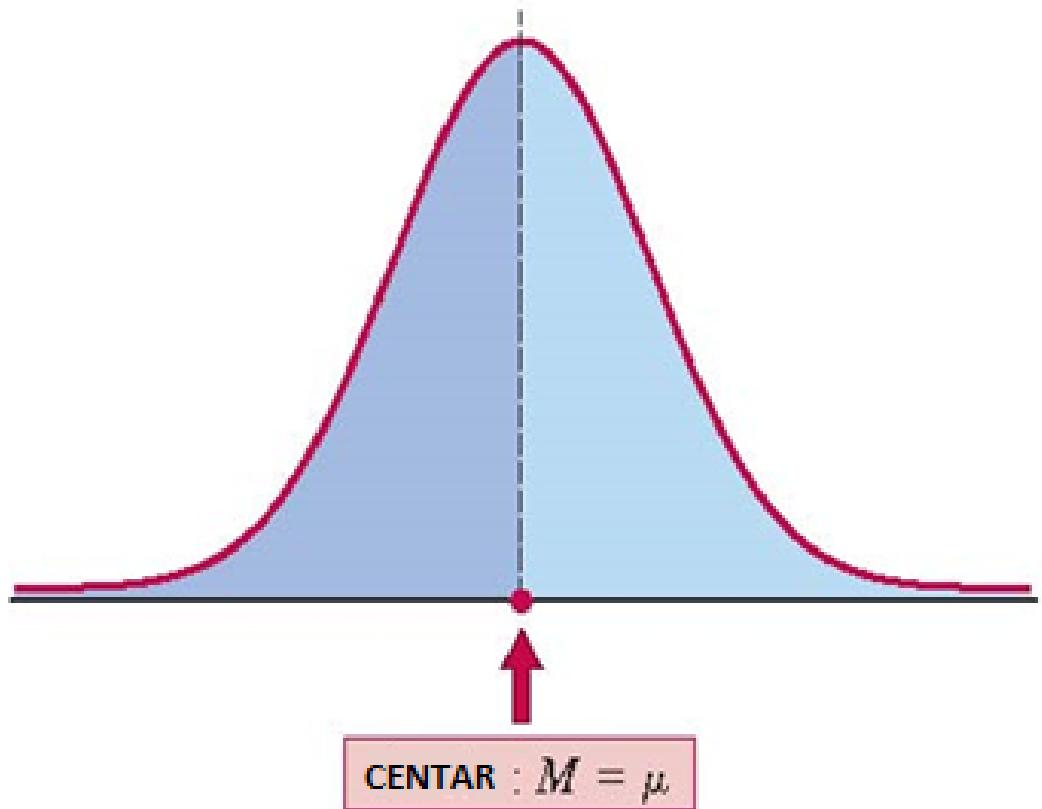
Distribucija ima karakterističan zvonolik oblik oko sredine:

Svaka teorijska normalna krivulja ima vertikalnu os simetrije koja dijeli zvonoliki oblik na dvije identične polovine.

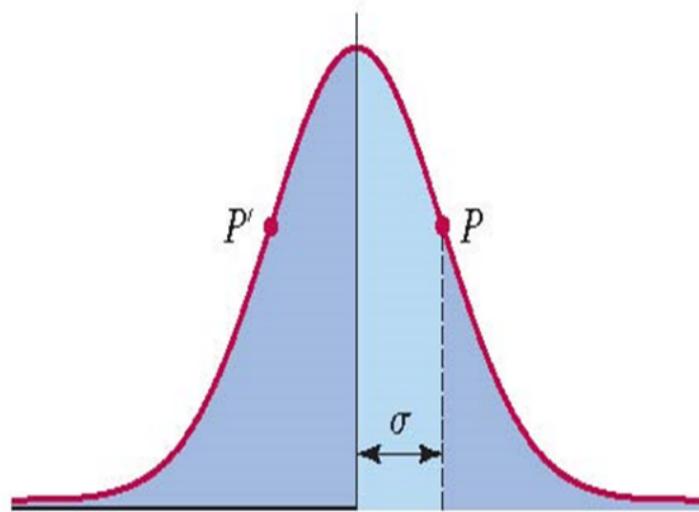
Svojstva normalne distribucije:

- Medijan, mod i sredina su jednaki
- Standardna devijacija normalne distribucije je horizontalna udaljenost od sredine do jedne od točaka pregiba kako je prikazano na slici 73.



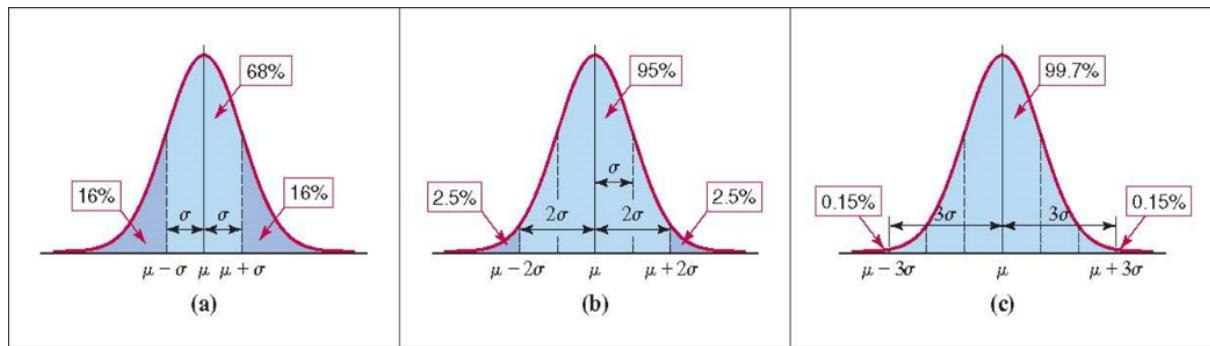


Slika 72: Normalna distribucija, srednja vrijednost (srednjak)



Slika 73: Grafičko određivanje standardne devijacije





Slika 74: Postotak vrijednosti koji se nalazi unutar jedne, dvije i tri standardne devijacije u normalnoj distribuciji

- Pravilo 68-95-99.7 - U svakoj normalnoj distribuciji približno 68% vrijednosti dolazi između 1 *sd* od sredine distribucije  $-1 < z < 1$ .

#### Standardiziranje normalne distribucije i Z-vrijednost

Standardiziranje normalne varijable  $X$  je proces kojim izražavamo koliko je svaka vrijednost varijable udaljena od sredine  $\mu$  koristeći standardnu devijaciju  $\sigma$  kao jedinicu mjerena.

##### #### Standardna normalna distribucija

Standardna normalna distribucija je normana distribucija sa sredinom 0 i standardnom devijacijom  $sd = 1$ . Svaku normalnu distribuciju moguće je transformirati u standardnu normalnu  $Z$  transformacijom.

Kontinuirana slučajna varijabla određena je funkcijom gustoće

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$

Funkcija je definirana na intervalu  $[-\infty, \infty]$ , a funkcija gustoće distribucije dana je formulom:

$$f(x) = \frac{1}{\sqrt{2\pi} \times 1} e^{-(x-0)^2/2 \times 1}$$

**4.3.1.1.2 Z vrijednost (Z skor)**  $Z$  skor je mjera udaljenosti pojedinog mjerjenja izražena u jedinicama standardne devijacije ( $\sigma$ ). Pri tome vrijedi sljedeće:

- $Z$  skor = 0 označava sredinu
- Pozitivan  $Z$  skor označava vrijednost desno od sredine
- Negativan  $Z$  skor označava vrijednost lijevo od sredine
- $Z$  skor = 1: jedna SD desno od sredine
- $Z$  skor -2: dvije SD lijevo od sredine.

Kumulativna funkcija distribucije slučajne varijable  $Z$  zadana je kao:

$$\Phi(z) = \int_{-\infty}^z \frac{1}{\sqrt{2}} e^{-x^2/2} dx$$

Na slici 75 shematski je prikazana transformacija normalne distribucije srednje vrijednosti 100 i standardne devijacije 10  $N(100, 10)$  u standardnu normalnu distribuciju  $N(0, 1)$ . Na slici 76 prokomentirajte značenje vrijednosti  $Z$ .

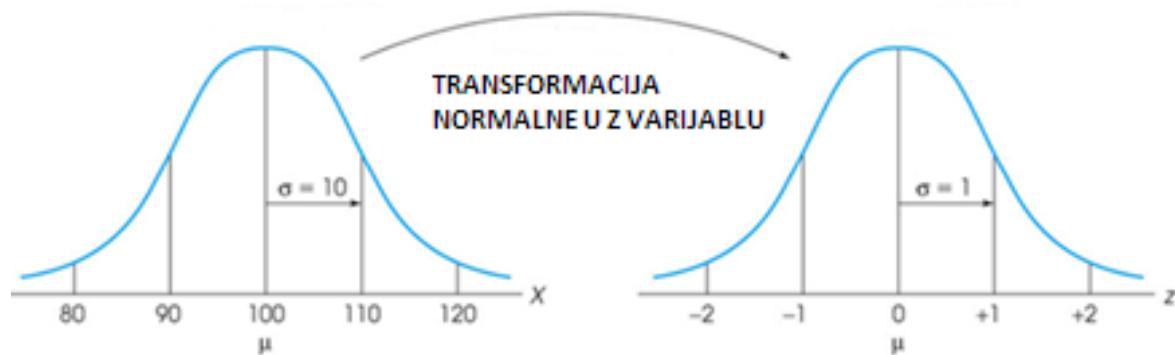
#### Korištenje standardnih normalnih tablica

Na slici 77 dan je primjer traženja kumulativne vjerojatnosti za zadani  $Z$ .  $Z$  tablice možemo koristiti i inverzno na način da za zadalu površinu, kumulativnu vjerojatnost tražimo kritičnu vrijednost  $Z$ .

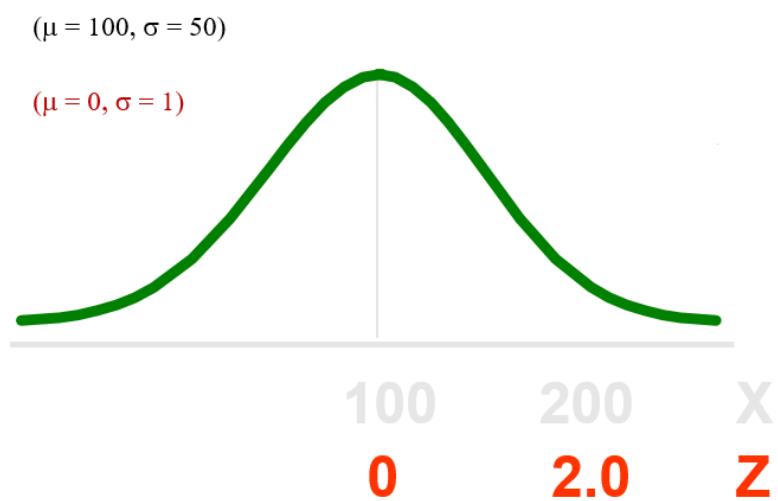
Određivanje vrijednosti  $Z_A$

Često nas zanima odrediti vrijednost  $Z$  za zadalu vjerojatnost. Na primjer, ukoliko imamo zadalu površinu ispod krivulje (vjerojatnost), koja joj  $Z$  vrijednost pripada (na x osi)?

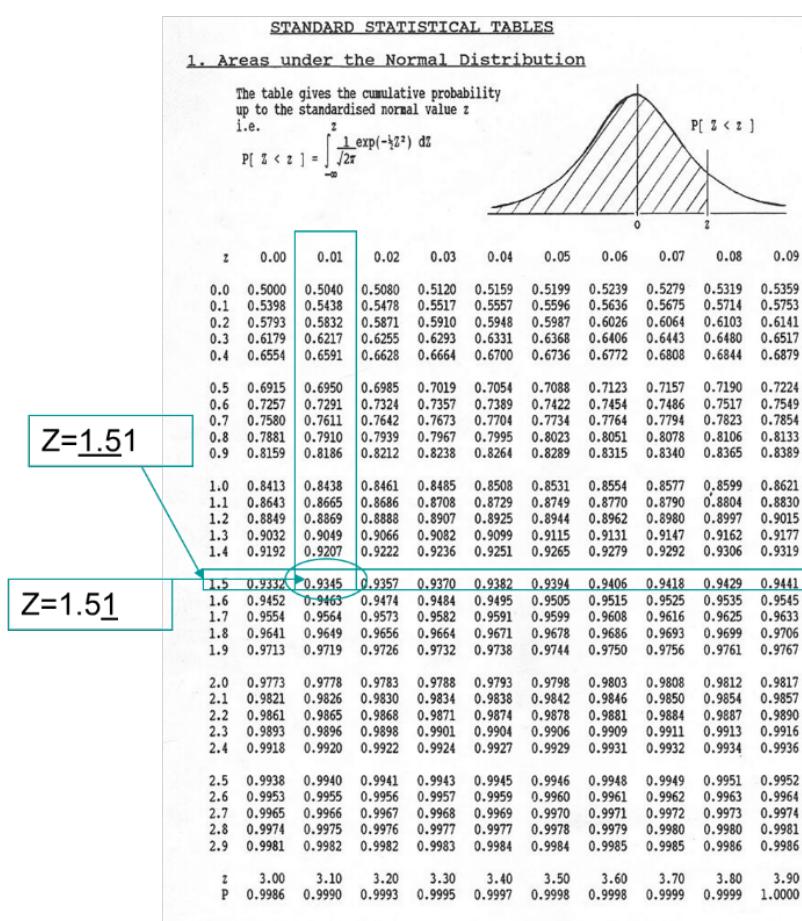




Slika 75: Transformacija normalne distribucije u standardnu normalnu



Slika 76: Rezultat transformacije normalne distribucije u standardnu normalnu



Kolika je površina ispod krivulje lijevo od  $Z=1.51$  na standardnoj normalnoj krivulji?

Površina je 93.45%

$Z=1.51$

$Z=1.51$

Slika 77: Primjer korištenja klasičnih Z tablica za traženje kumulativne vjerojatnosti dosezanja Z vrijednosti



$$P(Z > Z_A) = A$$

Primjer izračuna Z-skora na podacima o visinama osoba, uzetih sa stranice [http://socr.ucla.edu/docs/resources/SOCR\\_Data/SOCR\\_Data\\_Dinov\\_020108\\_HeightsWeights.html](http://socr.ucla.edu/docs/resources/SOCR_Data/SOCR_Data_Dinov_020108_HeightsWeights.html).

```
#učitavamo podatke
x <- read.csv("weights.csv", header = F, sep = ";", dec = ",")

#imenujmo kolone
names(x) <- c("Visina", "Masa")

#prebacimo u SI sustav mjernih jedinica
x$Visina <- x$Visina*2.5
x$Masa <- x$Masa*0.453592

#pogledajmo strukturu podataka
#skup podataka o visinama i masi osoba
str(x)
```

```
'data.frame': 25000 obs. of 2 variables:
 $ Visina: num 164 179 174 171 169 ...
 $ Masa : num 51.3 61.9 69.4 64.6 65.5 ...
```

Najprije računamo parametre populacije, imamo izmjerenu cjelokupnu populaciju:

```
#standardna devijacija populacije, sd računa za uzorak
populacija_sd <- sd(x$Visina)*sqrt((length(x$Visina)-1)/(length(x$Visina)))
populacija_mean <- mean(x$Visina)

#izračun z-skora za svaku opservaciju
z <- (x$Visina - populacija_mean)/populacija_sd

#ispis prvih 10 vrijednosti z- skora
z[1:10]
```

```
[1] -1.164 1.855 0.740 0.119 -0.107 0.372 0.950 1.061 -0.049 -0.638
```

Izaberimo jednu opservaciju iz skupa podataka, primjerice onu na 1000 redku i pogledajmo njezinu Z vrijednost:

```
z[1000]
```

```
[1] -0.149
```

Vizualizirajmo raspodjelu varijable od interesa, *visina* i pogledajmo gdje se unutar naše distribucije nalazi opservacija rednog broja 1000:

```
#izračunamo kumulativnu vjerojatnost do zadanog x-y (1000-ta opservacija)
p_donja1 <- pnorm(x$Visina[1000], populacija_mean, populacija_sd)
#korištenjem z-skora opservacije
p_donja2 <- pnorm(z[1000])

p_gornja1 <- 1 - p_donja1
p_gornja2 <- 1 - p_donja2
```

```
p_gornja1
```

```
[1] 0.559
```

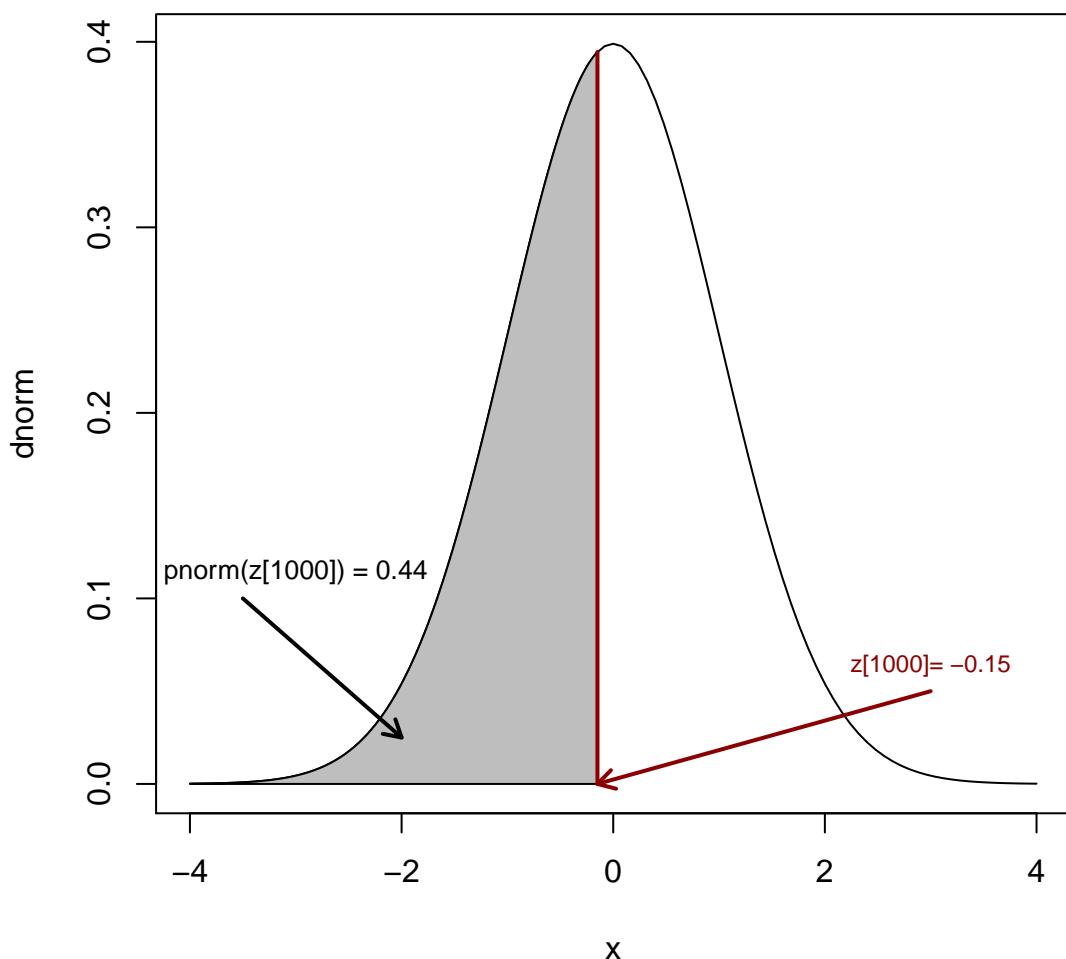
```
p_gornja2
```

```
[1] 0.559
```

Pogledajmo izračunato:



$$X \sim \text{Normal}(0,1)$$



Slika 78: Primjer izračuna Z-skora, 1

Upoznat ćemo se s načinom korištenja vjerojatnosnih distribucija u statističkom programu R. Posebnu pažnju ćemo posvetiti normalnoj razdiobi i standardnoj normalnoj razdiobi te t-distribuciji iako sve navedeno vrijedi i za sve ostale definirane i ugrađene distribucije vjerojatnosti slučajne varijable u sustavu.

Kroz nekoliko primjera ćemo se upoznati kako riješiti zadatke koji se temelje na normalnoj raspodjeli traženjem normalnih vjerojatnosti.

Primjer: Vrijeme potrebno za sastavljanje jednog računala je normalno distribuirano sa srednjom vrijednosti 50 minuta i standardnom devijacijom od 10 minuta: Kolika je vjerojatnost da računalo bude sastavljeno između 45 i 60 minuta? Ovaj zadatak može se riješiti samo jednom linijom kôda ali na ovom primjeru ćemo i sve nacrtati.

```
#računamo potrebnu vjerojatnost
p <- pnorm(60, mean=50, sd=10) - pnorm(45, mean=50, sd=10)
p
```

[1] 0.533

Isti zadatak ali i grafički prikaz radi boljeg razumijevanja, slika 79:

```
#parametri zadatka
mean<-50; sd<-10
donja<-45; gornja<-60

#područje na kojem je funkcija definirana
x <- seq(-4,4,length=100)*sd + mean

#gustoća funkcija po svim mogućim vrijednostima x-a
hx <- dnorm(x,mean,sd)
```

Crtamo base grafikom, funkcija `plot()`, argument `type="n"` samo definira dimezije grafa bez crtanja:

```
#iniciranje grafa
plot(x, hx,
      type="n",
      xlab="Vrijeme sastavljanja računala",
      ylab="",
      main="Normalna distribucija",
      axes=FALSE)

#proizvoljna os
axis(1, at=seq(0, 160, 20), pos=0)

#defineiramo područje koje želimo
i <- x >= donja & x <= gornja

#dodajemo liniju na graf, funkcija lines()
lines(x, hx)

#dodajemo poligon na graf, funkcija polygon()
polygon(c(donja,x[i],gornja), c(0,hx[i],0), col="gray")

#računamo vjerojatnost (identično kao prva linija kôda)
površina <- pnorm(gornja, mean, sd) - pnorm(donja, mean, sd) #area=površina=vjerojatnost

rezultat <- paste("P(",donja,"< IQ <",gornja,") = ", signif(površina, digits=3))

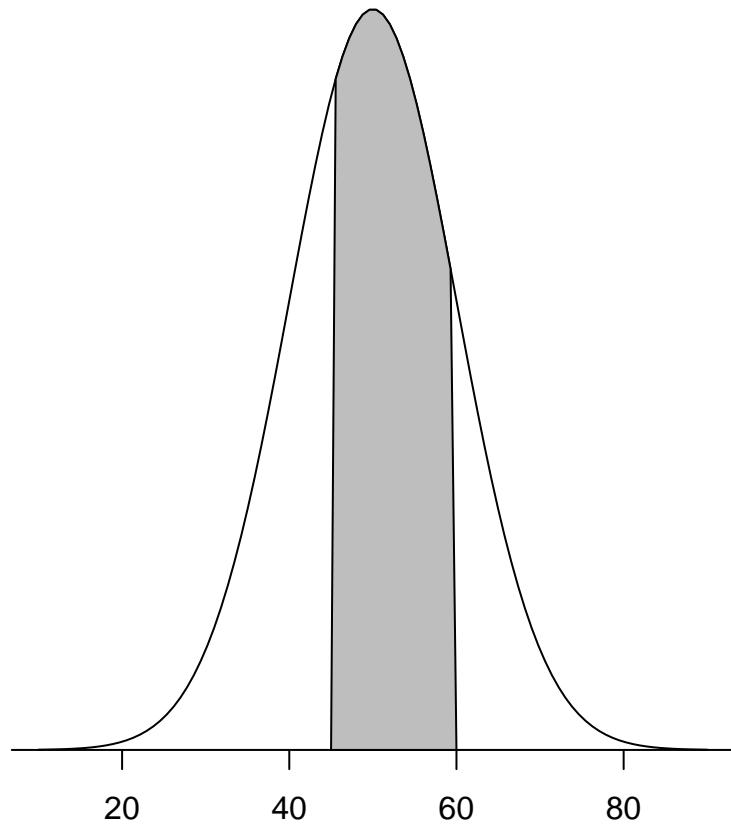
#dodajemo tekst na graf
mtext(rezultat,3)
```

Cijeli zadatak smo mogli riješiti i na način da poznatu normalnu distribuciju, u našem slučaju  $N(50, 10)$  prebacimo u standardnu normalnu Z-transformacijom i nastavimo izračun vjerojatnosti na standardnoj normalnoj  $N(0, 1)$ . Napravite zadatak samostalno na ovaj način. Način prebacivanja normalne distribucije u Z-skor dan je u primjeru koji slijed:



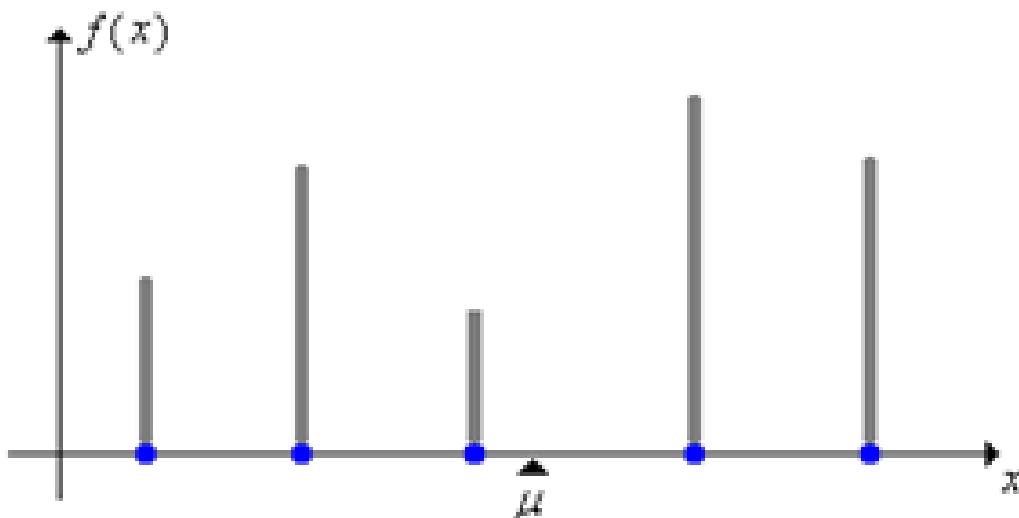
## Normalna distribucija

$$P( 45 < IQ < 60 ) = 0.533$$



Vrijeme sastavljanja ra unala

Slika 79: Grafički prikaz zadatka; izračun normalnih vjerovatnosti



## CENTAR MASE DISKRETNE SLUČAJNE VARIJABLE

Slika 80: Očekivanje diskretne slučajne varijable

Na ovome tečaju nemamo vremena upoznati se sa svim važnim distribucijama/razdiobama jedne kontinuirane slučajne varijable. Upoznali smo se s normalnom i t distribucijom jer se na ovaj način ponašaju sredine uzoraka te ove matematičke funkcije koristimo u procesu testiranja hipoteza o srednjoj vrijednosti. Na nekom drugom tečaju ćemo se upoznati s još nekim veoma važnim distribucijama kontinuirane slučajne varijable kao što su primjerice  $\chi^2$  distribucija prema kojoj se ponosa distribucija varijanci uzoraka ali i F distribuciju prema kojoj se ponaša omjer dviju varijanci. Te se distribucije radi ovoga koriste za testiranja hipoteza o varijanci ili pak omjeru varijanci čime čine osnovu ispisu gotovo svih statističkih metoda, počevši od analize varijance (ANOVA), linearog modeliranja (lm, glm, mix-model i slično).

### 4.3.2 Diskretna slučajna varijabla

Slučajnu varijablu  $X$  je diskretna slučajna varijabla ukoliko vrijedi:

- postoji konačan broj mogućih ishoda  $X$ , ili
- postoji precizno beskonačan broj mogućih ishoda  $X$ .

Brojčano beskonačan broj mogućih ishoda znači da postoji odnos odnos između ishoda i skupa cijelih brojeva. Nijedna takva jednoznačna korespondencija ne postoji za nebrojeno beskonačan broj mogućih ishoda.

Očekivanje diskretne slučajne varijable definiramo kao:

$$\mathbb{E}X = \sum x_i p(x_i)$$

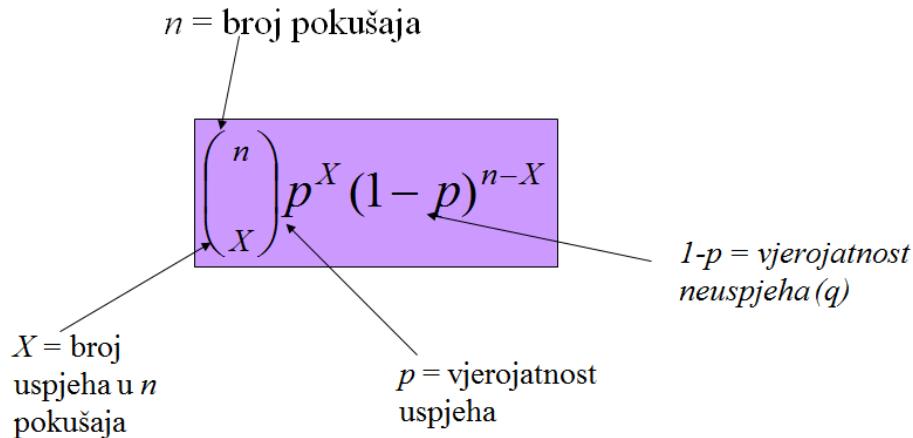
Proučite sliku 80 s označenim očekivanjem kontinuirane funkcije  $f(x)$ .

#### 4.3.2.1 Neke važne diskrete distribucije

**4.3.2.1.1 Binomna distribucija** Zamislimo eksperiment s dva moguća ishoda  $H$  ili  $T$ . Neka je

$$P(H) = p$$





Slika 81: Binomna distribucija, shematski

i

$$P(T) = 1 - p = q$$

Ovo je tzv. Bernoullijev eksperiment. Sada zamislimo  $n$  nezavisnih eksperimenata u kojem svaki eksperiment ima dva moguća ishoda. To je tzv. Binomni eksperiment. Neka je  $X$  broj ishoda  $H$ . To zovemo Binomna slučajna varijabla s parametrom  $n$  i  $p$ .

Drugim riječima, Bernoullijeva slučajna varijabla ima dva moguća ishoda, primjerice 0 ili 1. Binomna razdioba je zbroj neovisnih i identično distribuiranih Bernoulliovih slučajnih varijabli.

Binomna distribucija je dana izrazom

$$P(X=x) = \binom{n}{x} p^x (1-p)^{n-x}.$$

$$\mathbb{E}X = np, \mathbb{V}X = np(1-p)$$

Srednja vrijednost (očekivanje):  $\mu = np$

Varijanca:  $\sigma^2 = npq$

Standardna devijacija:  $\sigma = \sqrt{npq}$

Slično kako smo se upoznali s koritenjem klasičnih z i t tablica, tako ćemo kroz sljedeći primjer pokazati korištenja binomnih tablica.

Primjer: Pedeset posto odraslih djelatnika provodi manje od 20 minuta putujući na posao. Ukoliko slučajnim izborom 6 djelatnika, kolika je vjerojatnost da njih točno troje trebaju manje od 20 minuta do posla?

Pogledamo distribuciju za  $n = 6$  i vjerojatnost  $p = 0.5$ , pronaći ćete da vjerojatnost pojavljivanja točno 3 puta iznosi 31.2%, slika 82. Naravno, ovakve primjere ćemo rješavati uz pomoć sustava R.

##### Proporcije Binomna distribucija čini osnovu za statistiku proporcija. Proporcije su upravo binomna distribucija podijeljena s veličinom uzorka  $n$ . Primjerice, ukoliko imamo uzorak od 200 ljudi među kojima su 60 pušači,  $x = 60$ , a  $\hat{p} = 0.30$ . Statistika na proporcijama je slična binomnim vjerojatnostima, razlikuje se za faktor  $n$ .

Često nam je važnija proporcije neke pojave u populaciji od broja. U statističkom uzorkovanju, proporcija događaja se koristi kako bi se odredila prava proporcija pojave u populaciji. Za svaki jednostavni slučajni uzorak (SRS) veličine  $n$ , proporcija uzorka se računa na način:

$$\hat{p} = \frac{\text{frekvencija pojave u uzorku}}{n} = \frac{x}{n}$$

Primjer: U jednostavnom slučajnom uzorku od 50 studenata jednog fakulteta, 10 ih je iz Zagreba:  $\hat{p} = 10/50 = 0.2$ , tj 0.2 je proporcija Zagrepčana u uzorku.



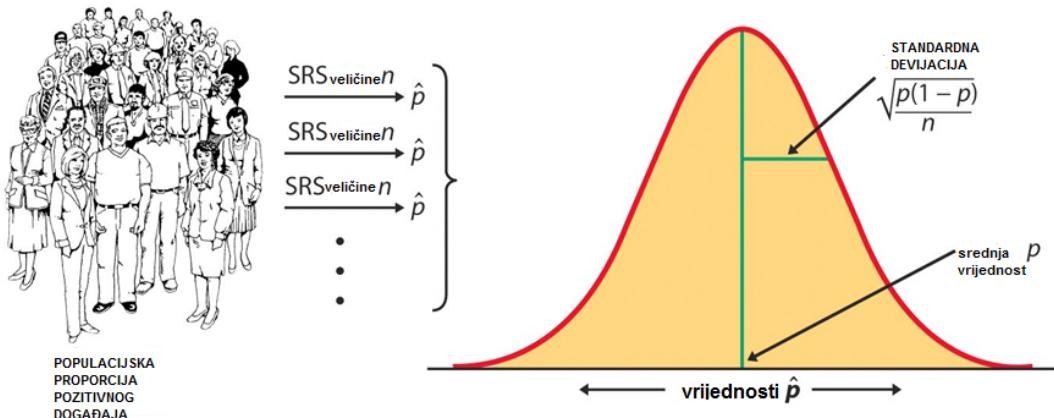
		<i>p</i>												
<i>n</i>	<i>x</i>	.01	.05	.10	.15	.20	.25	.30	.35	.40	.45	.50	.55	.60
2	0	.980	.902	.810	.723	.640	.563	.490	.423	.360	.303	.250	.203	.160
	1	.020	.095	.180	.255	.320	.375	.420	.455	.480	.495	.500	.495	.480
	2	.000	.002	.010	.023	.040	.063	.090	.123	.160	.203	.250	.303	.360
3	0	.970	.857	.729	.614	.512	.422	.343	.275	.216	.166	.125	.091	.064
	1	.029	.135	.243	.325	.384	.422	.441	.444	.432	.408	.375	.334	.288
	2	.000	.007	.027	.057	.096	.141	.189	.239	.288	.334	.375	.408	.432
	3	.000	.000	.001	.003	.008	.016	.027	.043	.064	.091	.125	.166	.216
6	0	.941	.735	.531	.377	.262	.178	.118	.075	.047	.028	.016	.008	.004
	1	.057	.232	.354	.399	.393	.356	.303	.244	.187	.136	.094	.061	.037
	2	.001	.031	.098	.176	.246	.297	.324	.328	.311	.278	.234	.186	.138
	3	.000	.002	.015	.042	.082	.132	.185	.236	.276	.303	.312	.303	.276
	4	.000	.000	.001	.006	.015	.033	.060	.095	.138	.186	.234	.278	.311
	5	.000	.000	.000	.000	.002	.004	.010	.020	.037	.061	.094	.136	.187
	6	.000	.000	.000	.000	.000	.000	.001	.002	.004	.008	.016	.028	.047

Slika 82: Primjer korištenja binomne tablica

Statistika za proporcije temelji se na normalnoj distribuciji jer se binomna distribucija može aproksimirati normalnom za  $np > 5$ .

Distribucija uzorka – proporcije Distribucija uzoraka za proporcije  $\hat{p}$  gotovo nikada nije normalna ali kako se povećava veličina uzorka postaje približno normalna. Pogledajte i prokomentirajte slike 83 i 84.

### Statistika za proporcije



Slika 83: Statistika za proporcije temelji se na binomnoj distribuciji koja se pak zamjenjuje normalnom za uzorce veće od 5



The diagram illustrates the connection between binomial distribution properties and normal distribution properties for proportions.

**BINOMNA**

$$\mu_x = np$$

$$\sigma_x^2 = np(1-p)$$

$$\sigma_x = \sqrt{np(1-p)}$$

**PROPORCIJE**

$$\mu_{\hat{p}} = p$$

$$\sigma_{\hat{p}}^2 = \frac{np(1-p)}{n^2} = \frac{p(1-p)}{n}$$

$$\sigma_{\hat{p}} = \sqrt{\frac{p(1-p)}{n}}$$

Annotations:

- A callout box labeled "Razlika faktor n" points to the term  $np$  in the first equation.
- A callout box labeled "Razlika faktor n" points to the term  $np(1-p)$  in the second equation.
- A callout box labeled "P-hat "proporcije uzorka"" points to the term  $p$  in the third equation.

Slika 84: Statistika za proporcije temelji se na binomnoj distribuciji koja se pak zamjenjuje normalnom za uzorce veće od 5

X	P(X)
0	=.135
1	=.27
2	=.27
3	=.18
4	
5	
...	...

$p(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$

Slika 85: Primjer izračuna Poisson vjerojatnosti ??

**4.3.2.1.2 Poisson distribucija** Poissonova distribucija se koristi za prebrojavanja – događaje koji se događaju pri konstantnoj stopi tijekom vremena. Poisson distribucija daje vjerojatnosti da zabilježimo x događaja u određenom vremenu (ili određenom prostoru).

$$P(x) = \frac{e^{-\lambda} \lambda^x}{x!}$$

gdje je  $\lambda$  = očekivani broj u zadanim okvirima (vrijeme).

Srednja vrijednost (očekivanje):  $\mu = \lambda$

Varijanca:  $\sigma^2 = \lambda$

Standardna devijacija:  $\sigma = \sqrt{\lambda}$

Primijetite da su za varijablu distribuiranu po Poisson distribuciji varijanca i sredina jednake!

Primjerice, ukoliko znamo da se prosječno zabilježe 2 nova slučaja mjesечно zarazom Virusom zapadnog Nila na području Danske tada možemo izračunati vjerojatnosti pojave 0, 1, 2, 3, 4, 5, 6... slučajeva Danskoj sljedećeg mjeseca izračunom Poisson vjerojatnosti, slika ??.

Na ovome tečaju nemamo dalje vremena upoznavati se sa svim značajnim razdiobama/distribucijama diskretne slučajne varijable.

#### Upoznavanje ugrađenim funkcijama distribucija u sustavu R

Kao što smo u mnogo navrata spomenuli, R je statistički program i okruženje. Tako su i raspodjele (distribucije) vjerojatnosti slučajnih varijabli za ogroman broj parametarski određenih razdioba već ugrađene u sustav i dolaze instalacijom R-a unutar paketa `stats`. Osnovna funkcija za pristup mnogobrojnim funkcijama za rad s distribucijama slučajne varijable je funkcija `distributions()`. Upoznajmo se s funkcijom te popisom mnogobrojnih distribucija kontinuiranih i diskretnih slučajnih varijabli.

```
#informacije o ugrađenim vjerojatnosnim distribucijama u sustavu R
?distributions
```

Primjer traženja pomoći za željenu distribuciju - binomna distribucija:

```
?dbinom
```

Upoznavanje s d,p,q,r na normalnoj i t razdiobi

Normalna distribucija

```
help(Normal)
?Normal
```

**d** - `dnorm()`



Ukoliko imamo skup vrijednosti, funkcija vraća visinu funkcije vjerojatnosti za zadatu točku. Ukoliko zadamo samo točku, funkcija prepostavlja da želite vrijednost za standardnu normalnu distribuciju, srednja vrijednost=0 i standardna devijacija=1. Funkcija putem svojih opcija omogućava određivanje željenu sredine i sd normalne distribucije.

Nacrtajmo standardnu normalnu distribuciju:

```
x <- seq(-3,3,0.1)
plot(x=x, y=dnorm(x, mean=0, sd=1), type='l')
```

Gustoća (visina) na vrijednosti x=0:

```
dnorm(0)
```

Dodat ćemo ovu vrijednost na otvoreni graf.

Gustoća (visina) na vrijednosti x=0, za normalnu distribuciju sa srednjim vrijednosti=4, sd=1 (defoult)

Neka druga normalna distribucija, srednja vrijednost 10, sd=3:

```
x    <- seq(-2,23,length=1000)
y    <- dnorm(x,mean=10, sd=3)

#crtamo
plot(x,y, type="l", lwd=1,main='N(10,3)')
#dodajemo ravnu liniju
abline(v=10, lty=2, col=rgb(0.9,0.5,0.7), lwd=2)
```

Možemo izračunati gustoću funkcije (visinu) za bilo koju točku x, bilo koje normalne distribucije.

```
#prouzvoljna normalna distribucija
#srednja vrijednost 4; standna devijacija 1
#tražimo vrijednost na 0
dnorm(0,mean=4)
```

[1] 0.000134

```
#srednja vrijednost 4; standna devijacija 10
#tražimo vrijednost na 0
dnorm(0,mean=4, sd=10)
```

[1] 0.0368

Traženje gustoće odjednom na većem broju točaka, stavljenih u vektor v:

```
#tražimo vrijednost za cikup vrijednosti x
v <- c(0,1,2)
dnorm(v)
```

[1] 0.399 0.242 0.054

Nacrtat ćemo standardnu normalnu distribuciju na rasponu od -10 do 10:

```
#sekvenca za koju ćemo tražiti izračun gustoće (visinu funkcije)
x <- seq(-10,10,by=0.1)
```

```
#pogledajmo prvih nekoliko vrijednosti
head(x)
```

[1] -10.0 -9.9 -9.8 -9.7 -9.6 -9.5

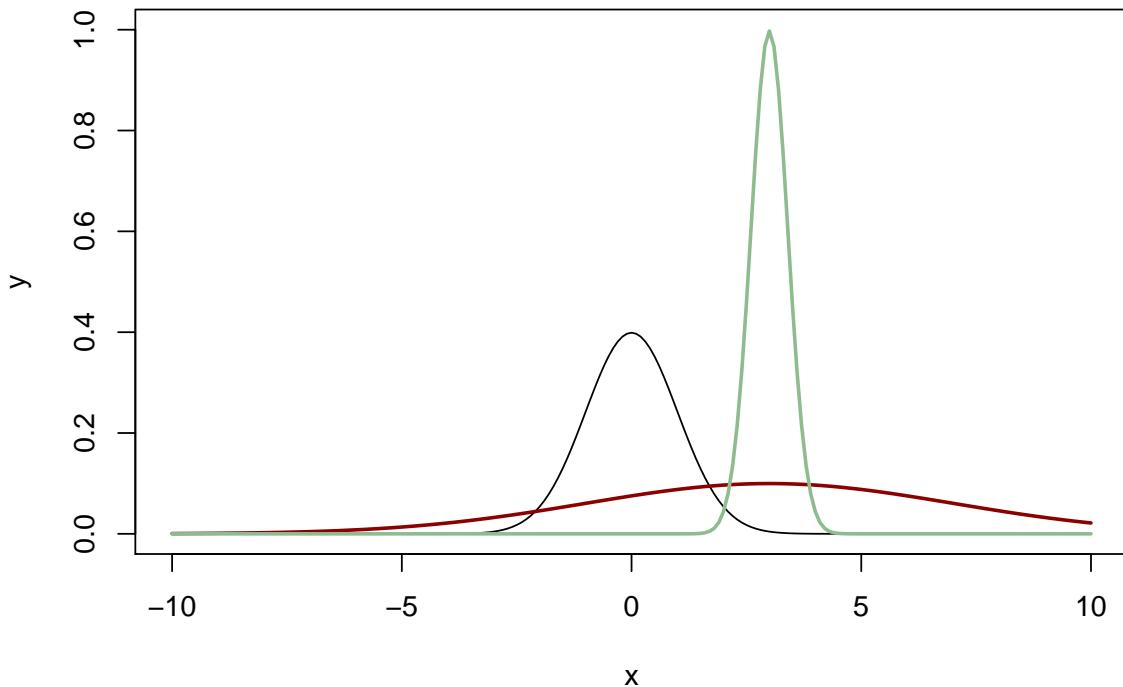
```
#visina standardne normalne na sekvenci
y <- dnorm(x)
```

```
#pogledajmo prvih nekoliko vrijednosti
head(y)
```

[1] 7.69e-23 2.08e-22 5.57e-22 1.48e-21 3.88e-21 1.01e-20



### Funkcije gusto a, visine za nekoliko normalnih distribucija



Slika 86: Gustoće funkcije nekoliko normalnih distribucija

```
#crtamo izračunate vrijednosti
plot(x,y,
      type="l",
      main="Funkcije gustoća, visine \n za nekoliko normalnih distribucija",
      cex=.8,
      ylim=c(0, 1))

#računamo gustoće, visine funkcije za neku drugu
#normalnu distribuciju i nadodajemo grafu
y <- dnorm(x,mean=3,sd=4)
lines(x, y, lwd=2, col="red4")

#nova normalna distribucija, manja standardna devijacija
y <- dnorm(x,mean=3,sd=.4)
lines(x, y, lwd=2, col="darkseagreen")
```



**p - pnorm()**

Za zadani broj ili vektor vrijednosti računa vjerojatnost da normalno distribuirana varijabla poprimi vrijednost manju od zadane. Kumulativna funkcija distribucije. Prihvata jednake opcije kao i funkcija *dnorm()*:

Možemo dobiti kumulativne vjerojatnosti za bilo koju normalnu distribuciju:

```
#sekvenca od -4 do 4 x (z, ako radimo sa standarnom normalnom)
x <- seq(-10,10,by=0.1)
head(x)

[1] -10.0 -9.9 -9.8 -9.7 -9.6 -9.5

#kumulativna vjerojatnost dosezanja x, z jer radimo na standardnoj normalnoj distribuciji
y <- pnorm(x)
head(y)

[1] 7.62e-24 2.08e-23 5.63e-23 1.51e-22 4.00e-22 1.05e-21
```

Izračunato možemo, prema želji i vizualizirati. Nacrtat ćemo parove vrijednosti, x i visinu funkcije na  $f(x)$  tj. y. Prokomenti-rajte sliku 87

```
#tip prkaza kao linija
plot(x,y, type='l',
      main="Kumulativna vjerojatnosti \nza nekoliko normalnih distribucija",
      cex=.8)

#računamo kumulativne vjerojatnosti za neku drugu
#normalnu distribuciju i nadodajemo grafu
y <- pnorm(x,mean=3,sd=4)
lines(x, y, lwd=2, col="red4")

#nova normalna distribucija, manja standardna devijacija
y <- pnorm(x,mean=3,sd=.4)
lines(x, y, lwd=2, col="darkseagreen")
```

Tražimo kumulativnu vjerojatnost za  $x=0$  ( $z=1$ ), za standarnu normalnu distribuciju:

```
pnorm(0)
```

```
[1] 0.5
```

Tražimo kumulativnu vjerojatnost za  $x=1$  ( $z=1$ ), za standarnu normalnu distribuciju:

```
pnorm(1)
```

```
[1] 0.841
```

Ako želimo kumulativnu vjerojatnost iznad zadane vrijednosti, argument *lower.tail*. U primjeru koji slijedi tražimo vjerojatnost premašivanja zadanog kvantila ili vrijednosti (na vrijednosti 0) za normalnu distribuciju standardne devijacije 1 (predodređeno, nije napisano) i srednje vrijednosti 2.

```
pnorm(0,mean=2, lower.tail=F)
```

```
[1] 0.977
```

Vjerojatnost dosezanja  $z=1$  na standardnoj normalnoj distribuciji:

```
pnorm(1)
```

```
[1] 0.841
```

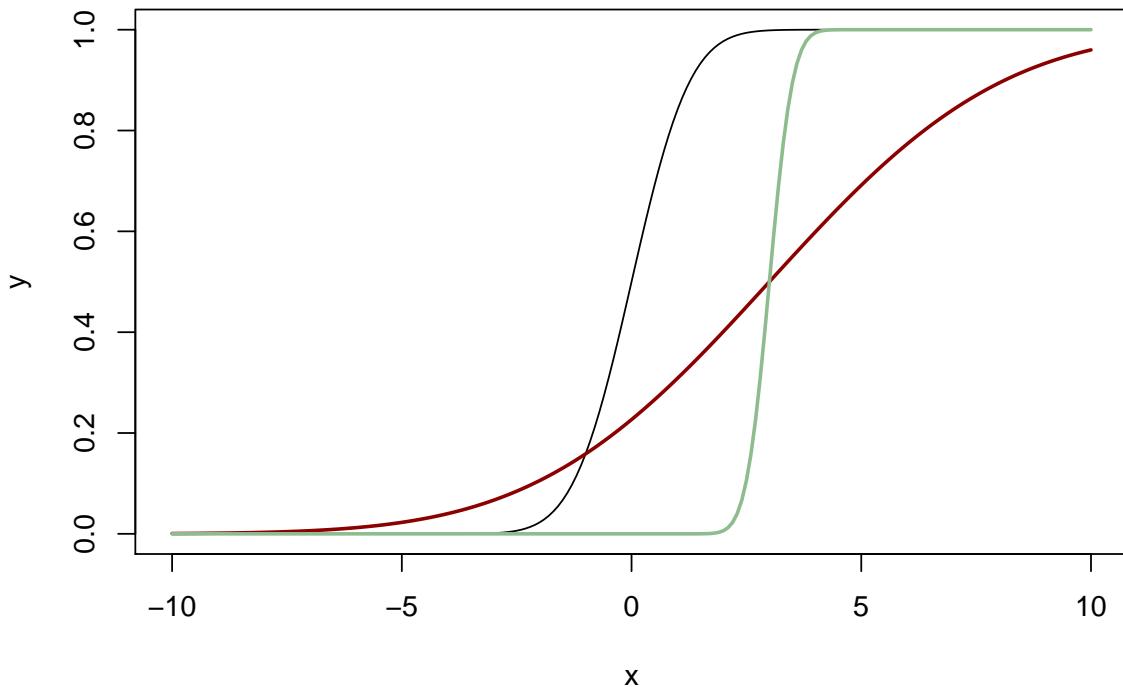
Vjerojatnost prekoračenja  $z=1$  na standardnoj normalnoj distribuciji:

```
pnorm(1,lower.tail=FALSE)
```

```
[1] 0.159
```



## Kumulativna vjerojatnost za nekoliko normalnih distribucija



Slika 87: Kumulativne vjerojatnosti nekoliko normalnih distribucija

Vjerojatnost prekoračenja  $x=0$ , za normalnu distribuciju sredine 2 i standardne devijacije 1:

```
pnorm(0,mean=2,lower.tail=FALSE)
```

```
[1] 0.977
```

```
q - qnorm()
```

Funkcija `qnorm()` inverzna je funkciji `pnorm()`. `qnorm()` vraća informaciju na kojoj vrijednosti Kumulativna Funkcija Distribucije doseže zadani vrijednosti. Sukladno tome, kada imamo standardnu normalno distribuiranu slučajnu varijablu funkcija vraća pridruženu Z vrijednost.

U primjeru koji slijedi tražimo kvantil na kojem kumulativna funkcija vjerojatnosti poprimi vrijednost 0.5 za standardnu normalnu distribuciju:

```
qnorm(0.5)
```

```
[1] 0
```

Primijetite da u sljedećem primjeru imamo isti upit:

```
qnorm(0.5,mean=1)
```

```
[1] 1
```

Tražimo kvantil na kojem kumulativna funkcija vjerojatnosti poprimi vrijednost 0.5 za normalnu distribuciju srednje vrijednosti 1 i standardne devijacije 2:

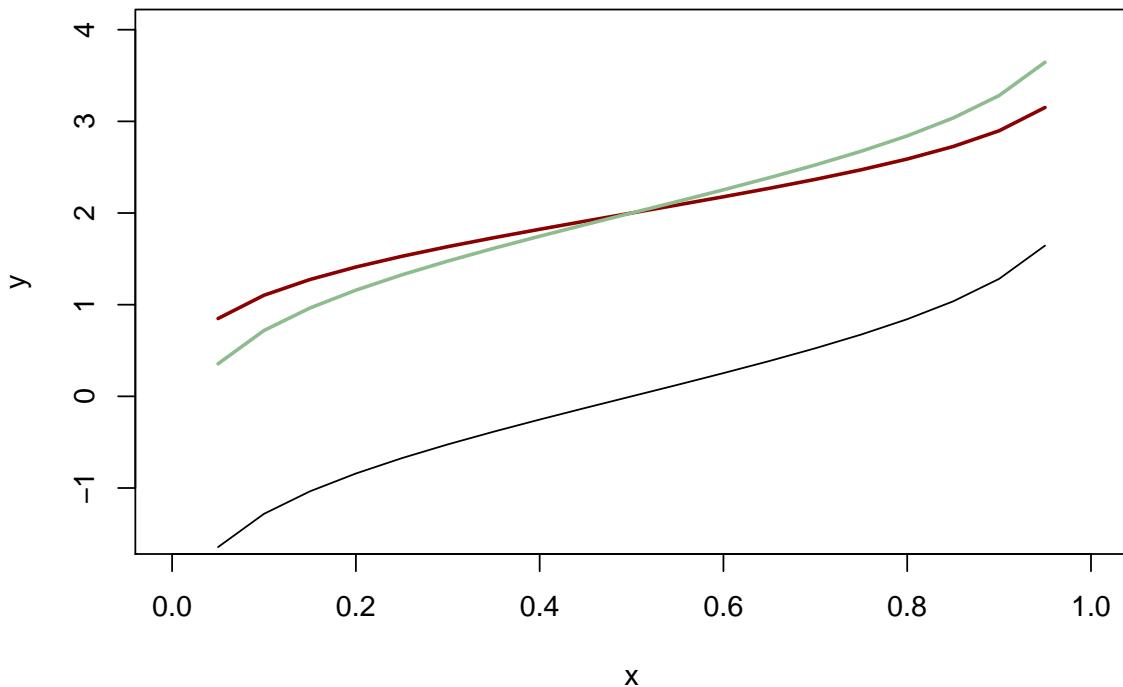
```
qnorm(0.5,mean=1, sd=2)
```

```
[1] 1
```

Tražimo kvantil na kojem kumulativna funkcija vjerojatnosti premaši vrijednost 0.3 za normalnu distribuciju srednje vrijednosti 1 i standardne devijacije 2:



## Funkcije kvantila za standardnu normalnu distribuciju



Slika 88: Funkcija kvantila za standardnu normalnu distribuciju

```

qnorm(0.3,mean=2,sd=2, lower.tail=F)

[1] 3.05

x <- seq(0,1,by=.05)
y <- qnorm(x)
plot(x,y,
      type="l",
      main="Funkcije kvantila \n za standardnu normalnu distribuciju",
      cex=.8,
      ylim=c(-1.5, 4))

#računamo kvantile za drugu distribuciju
y2 <- qnorm(x,mean=2,sd=.7)

#nadodajemo funkcije kvantila
lines(x, y2, lwd=2, col="darkred")

#računamo kvantile za treću distribuciju
y3 <- qnorm(x,mean=2,sd=1)

#nadodajemo funkcije kvantila
lines(x, y3, lwd=2, col="darkseagreen")

r - rnorm()

```

Funkcija `rnorm()` generira slučajnu varijablu, zadane veličine iz normalne distribucije. Obvezan argument je veličina varijable, opcionalno možemo odrediti sredinu i standardnu devijaciju.



Generiramo slučajnu normalnu varijablu duljine 100, standardna normalna distribucija. Funkcija koristi generator slučajnih brojeva kao početnu vrijednost te ukoliko želimo imati jednake slučajne vrijednosti (pseudoslučajne) potrebno je prije provođena funkcije namjestiti generator na jednaku vrijednost. Ukoliko to ne napravimo generator će kao početnu vrijednost koristiti vrijeme sustava u trenutku pozivanja naredbe, funkcija `Sys.time()`.

```
#postavljanje generatora slučajnih brojeva
set.seed(1)

#slučajna 4 broja iz standardne normalne distribucije
x <- rnorm(200)

#više histograma na jednom prikazu
hist(rnorm(200,mean=-2), col=rgb(0,0,0,0.2),
      main="Četiri generirane slučajne varijable \nfunkcijom rnorm()", 
      xlim=c(-6, 20),
      ylim=c(0, 60))

hist(rnorm(200,mean=5, sd=.8), col=rgb(0,0,.2,0.2), add=T)

hist(rnorm(200,mean=3, sd=1.5), col=rgb(0.2,0,0,0.2), add=T)

hist(rnorm(200,mean=7, sd=3), col=rgb(0,0,0,0.2,0.2), add=T)
```

#### **ZA SAMOSTALAN RAD:**

Na sličan način rješite sljedeće zadatke. Barem jedan od zadataka adekvatno i vizualizirajte.

```
#ZADATAK 1: Kolika je vjerojatnost  $P(Z < 1.6)$ ?

#ZADATAK 2: Kolika je vjerojatnost  $P(Z > 1.6)$ ?

#ZADATAK 3: Kolika je vjerojatnost  $P(Z < -2.23)$ ?

#ZADATAK 4: Kolika je vjerojatnost  $P(Z < 1.52)$ ?

#ZADATAK 5: Kolika je vjerojatnost  $P(0.9 < Z < 1.9)$ ?

#ZADATAK 6: Povrat investicije je normalno distribuiran sa sredinom 10% i
#standardnom devijacijom od 5%. Kolika je vjerojatnost gubitka novca? (Matematički
#napisano: Kolika je vjerojatnost  $P(X < 0)$ )?

#ZADATAK 7: Poznato je da studenti Svaučilišta u Zagrebu rješavaju ispit iz
#matematike (broj bodova) 78 sa standardnom devijacijom 10.4. Pronađite postotak
#studenata koji su riješili test s više od 89 bodova.

#ZADATAK 8: Kolika je vjerojatnost da ćemo dobiti 10 glava ukoliko bacimo novčić
#20 puta? (binomna)

#ZADATAK 9: Provodite studiju o vjerojatnosti nastanka bolesti. Poznato je da je
#vjerojatnost razvoja bolesti ozračenih osoba iznosi 0.05. Slučajno izaberete
#500 ozračenih osoba. a) Za koliko pretpostavljate da će razviti bolest?
#Izračunajte +- standardna devijacija procjene.

#ZADATAK 10: Kolika je očekivana vrijednost i varijanca za jedno bacanje novčića?

#ZADATAK 11: Bacamo novčić 10 puta. Koja je očekivana vrijednost
#i varijanca za ishod glava?

#ZADATAK 12: Poznato je za London da je 57% dana u godini oblačno,
#jednako raspoređeno tijekom svih mjeseci. Pronađite srednju vrijednost,
```



```
#varijancu i standardnu devijaciju broja oblačnih dana tijekom lipnja.
```

```
#ZADATAK 13: Ukoliko je broj poziva koji primite na svoj mobilni telefon Poisson process  
#s konstantnom stopom Lambda=2 poziva na sat, kolika je vjerojatnost da će vam telefon zazvoniti  
#ukoliko ste ga zaboravili ugasiti u kino dvorani gledajući film koji traje 1.5 sat?  $X \sim \text{Poisson}(\Lambda = 2)$ 
```

```
#ZADATAK 14: Objasnite što računamo u narednim linijama:
```

```
pnorm( c(0,1,2))
```

```
[1] 0.500 0.841 0.977
```

```
y <- pnorm(seq(-20,20,by=.1))  
pnorm(1,mean=3,sd=1.2)
```

```
[1] 0.0478
```

```
#ZADATAK 15: Prosječna potrošnja goriva automobila marke Renault je 7.3 l benzina  
#na 100 km sa standardnom devijacijom 2.2 l. Kupili samo polovni automobila marke Renault.  
#Kolika je vjerojatnost da naše vozilo troši manje od 6.8 l benzina na 100 km puta?
```



**4.3.2.2 Studentova t distribucija** Omogućava nam zaključivanje o tome koliko je sredina slučajnog uzorka udaljena od poznate ili prepostavljene populacijske sredine  $\mu$  u jedinicama standardne greške izračunate iz uzorka. Testiranje po t distribuciji izvodi se kada ne znamo standardnu devijaciju populacije. Posebno je važna kod malih uzoraka ( $df < 30$ ). Ukoliko je uzorak veći od 30 slična je standardnoj normalnoj distribuciji što je vizualno i prikazano na slici 89. Način na koji je slika napravljena slika 89i sami isprobajte nacrtati nove t-distribucije u usporedbi s normalnom. Primijetite da se najveća razlika između normalne i t distribucije događa u području njihovih "repova", slika 90.

```
#kritične vrijednosti za 90, 95, 99% intervale pouzdanosti za sredine (CI)
qt(c(.95, .975, .995), df=9)

## [1] 1.83 2.26 3.25

pt(-2.1, 11)      # daje nam vjerojatnosti[t < -2.1] za t distribuciju s 11 stupnjava slobode

## [1] 0.0298

#uzimamo slučajene uzorke veličine 50 iz t disttibucije s 11 stupnjeva slobode (df)
tSamp = rt(50, 11)
#uspoređujemo različite t distribucije, t distribucije s različitim
#stupnjevima slobobe sa standardnom mormalnom distribucijom
#područje distribucije
xs = seq(-5,5,.01)

#crtamo
plot(xs, dnorm(xs), type="l", lwd=2, col="black", ylab="gustoća",
      main="Usporedba t distribucije i standardne normalne", cex.main=0.8)
#dodajemo pojedine t distribucije
lines(xs, dt(xs, 10), lwd=2, col="darkblue")
lines(xs, dt(xs, 4), lwd=2, col="red4")
lines(xs, dt(xs, 1), lwd=2, col="darkgreen")
#dodajemo legendu
legend("topright", col=c("black","darkblue","red4","darkgreen"),
       legend=c("N","t, df=10","t, df=4","t, df=1"), lty=1)
```

Primjer korištenja t-tablica dan je na slici 91.

Tražimo vrijednost t statistike za 10 stupnjeva slobode (veličina uzorka  $n=11$ ), a da je površina (vjerojatnost) 0.05:

### Stupnjevi slobode

U statistici, broj stupnjeva slobode je broj vrijednosti u konačnom izračunu statistike koja se može slobodno razlikovati, varirati.

Procjene statističkih parametara mogu se temeljiti na različitim količinama podataka. Broj neovisnih podataka koji ulaze u procjenu parametra nazivaju se stupnjevi slobode (df). Općenito, stupnjevi slobode procjene parametra jednaki su broju nezavisnih rezultata koji ulaze u procjenu minus broj parametara koji se koriste kao međukoracima u procjeni samog parametra, primjerice kada procjenjujemo varijancu populacije iz uzorka.

Matematički, stupnjevi slobode su dimenzija domene slučajnog vektora, ili bitno broj "slobodnih" komponenata: koliko se komponenti trebaju znati prije nego što je vektor potpuno određen. Koncept stupnjeva slobode mnogima je teško razumljiv na početku. Dat ćemo nekoliko primjera jednostavnim, nestatističkim rječnikom:

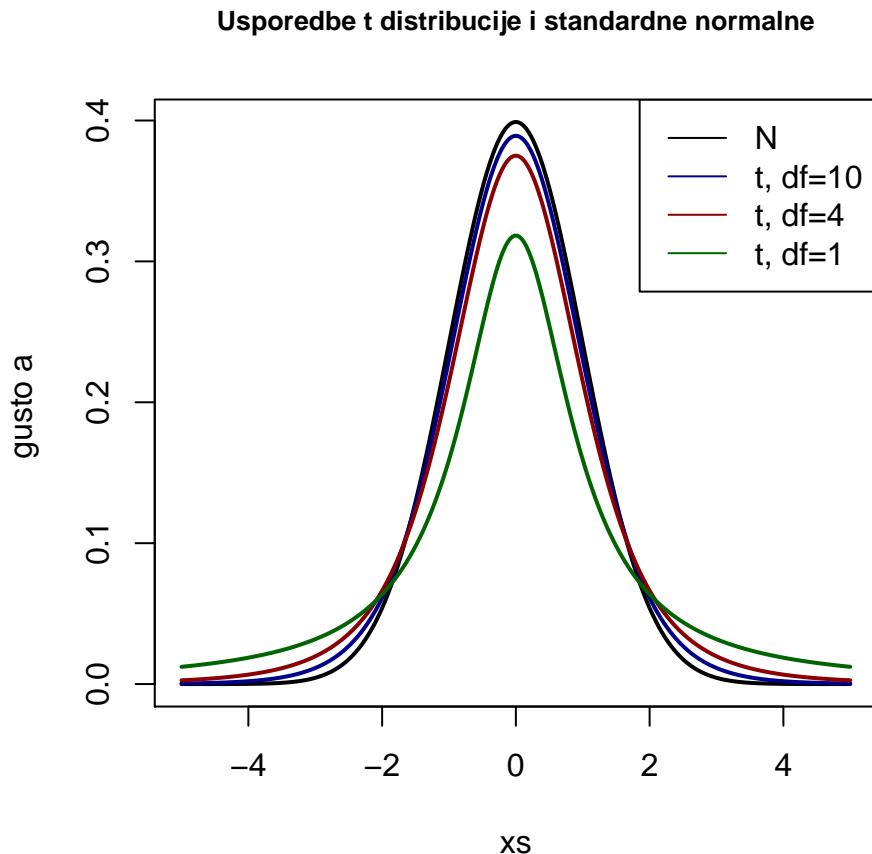
- 1) Ukoliko dobijete 10 slobodnih dana a jedan potrošite na planiranja odmora, planirati možete i trebate samo za devet dana.
- 2) Zamislite da ste trener nogometne reprezentacije te morate odrediti jedanaest igrača za različite pozicije. Ukoliko zname da iz nekog razloga igrač xy mora biti uključen u tim na vama je da odaberete još samo deset igrača.

*d, p, q i r funkcije na Studentovoj t-distribuciji*

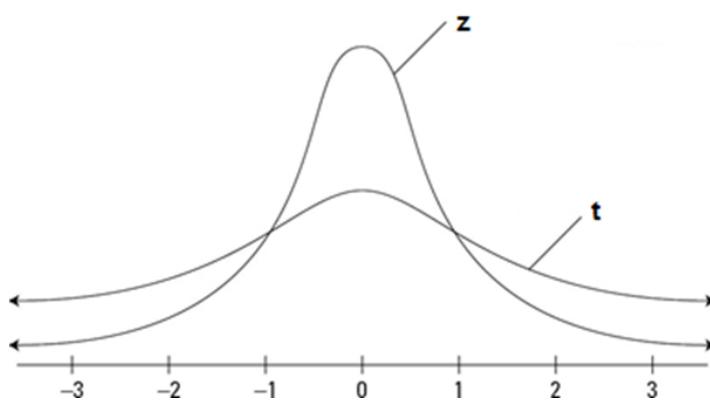
```
help(TDist)
?TDist
```

Funkcije koje slijede pandan su funkcijama koje smo upoznali za normalnu distribuciju. Razlika je u tome da prepostavljaju da su vrijednosti normalizirane na  $\mu = 0$  i  $\sigma = 1$ . Moramo specificirati stupnjeve slobose! Imena su kompatibilna, prate konvenciju tako da imamo *dt()*, *pt()*, *qt()*, i *rt()*.



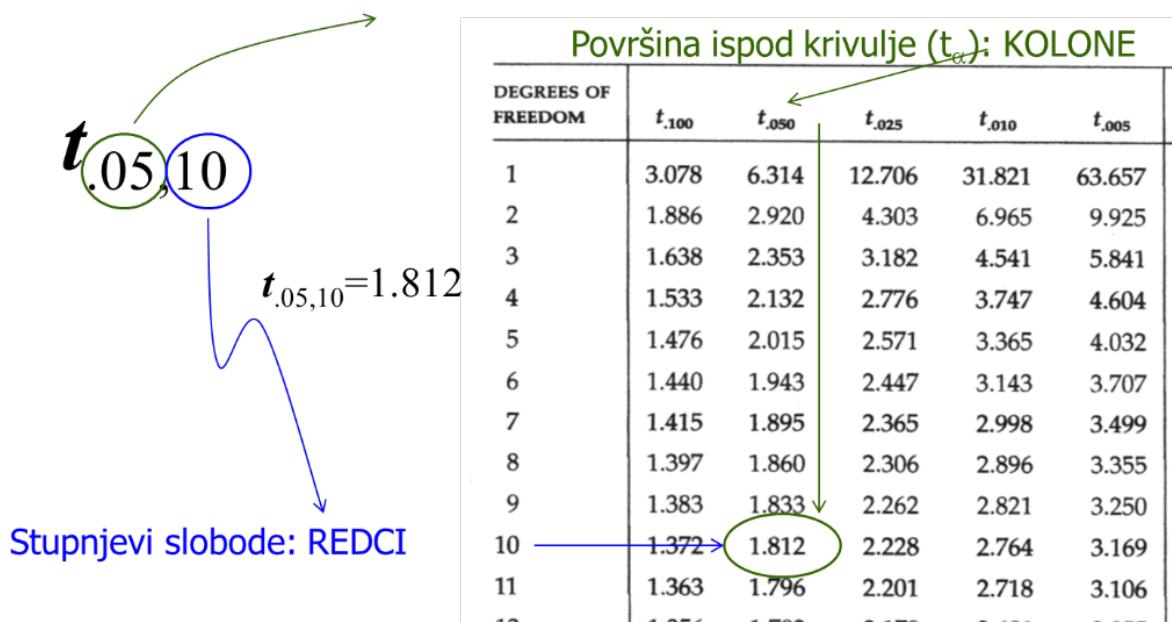


Slika 89: Usporedba nekoliko t-distribucija i standardne normalne; t-distribucija prelazi u normalnu za uzorke veće od 30



Slika 90: Najveća razlika između normalne i t distribucije je u njihovim repovima





Slika 91: Primjer korištenja t-tablica

Funkcija `dt()` pandan je već ranije predstavljenoj funkciji `pnorm()` kojom računamo gustoću funkcije na zadanom kvantilu za određenu t distribuciju.

Jednako kako smo radili s normalnom distribucijom, funkcije `d`, `p`, `q` i `r` rade i s bilo kojom drugom parametarskom distribucijom. Tako u sljedećem primjeru tražimo gustoću funkcije t sa šest stupnjeva slobode (parametar t funkcije) na x=1. Prokomentirajte tzv. parametar centraliteta t distribucije `ncp`.

```
#?dt
dt(1,df=6)
```

```
[1] 0.223
```

U sljedećem primjeru korištenjem funkcije za kumulativna distribucija vjerojatnosti prema t distribuciji `pt()` tražimo kumulativnu vjerojatnost za x=-3 za t distribuciju s 10 stupnjeva slobode:

```
pt(-3,df=10)
```

```
[1] 0.00667
```

U sljedećem primjeru korištenjem funkcije za kumulativna distribucija vjerojatnosti prema t distribuciji `pt()` tražimo kumulativnu vjerojatnost za x=-3 za t distribuciju s 10 stupnjeva slobode i parametrom `ncp` 3:

```
pt(3,3, df=10)
```

```
[1] 0.473
```

Jednako kao i kod normalne distribucije, inverzna funkcija kumulativne vjerojatnosti je `qt()`. Prokomentirajte što tražimo sljedećim naredbama:

```
qt(0.05,df=10)
```

```
[1] -1.81
```

```
qt(0.95,df=10)
```

```
[1] 1.81
```

```
qt(0.05,df=20)
```

```
[1] -1.72
```



```
qt(0.95, df=20)
```

```
[1] 1.72
```

```
v <- c(0.005, .025, .05)
qt(v, df=253)
```

```
[1] -2.60 -1.97 -1.65
```

```
qt(v, df=25)
```

```
[1] -2.79 -2.06 -1.71
```

I u konačnici, ukoliko želimo generirati slučajnu varijable prema t razdiobi koristimo generator slučajnih brojeva *rt()* i slijede primjer generiranje varijable veličine 30 iz centralne t distribucije s 20 stupnjeva slobode.

```
set.seed(1)
rt(30, df=20)
```

```
[1] -0.6241 -0.6811  0.3900 -0.3027 -0.2516  0.4448 -0.3289 -0.0144
[9]  0.7685  0.8367  0.1130  0.4393 -0.8955  0.3514 -0.0992 -0.0711
[17]  1.0942  1.0043 -0.1760 -1.0529 -0.6853  0.8031  0.2908  0.4285
[25]  0.2332 -0.1569 -0.0362  0.0327 -1.3802  0.2522
```

**4.3.2.3 Fisherova F distribucija** Oznaka  $X \sim F(n_1, n_2)$  se koristi za označavanje slučajne varijable  $X$  koja je distribuirana prema  $F$  distribuciji s parametrima  $n_1$  i  $n_2$ , što su pozitivni cijeli brojevi koji indiciraju stupnjeve slobode brojnika i nazivnika.  $F$  distribucija se još naziva i *Fisher–Snedecor distribution*. Slučajna varijabla  $X$  distribuirana po  $F$  distribuciji s  $\sim n_1$  i  $\sim n_2$  stupnjeva slobode ima funkciju gustoće (engl. *probability density function*):

$$f(x) = \frac{\Gamma((n_1 + n_2)/2)(n_1/n_2)^{n_1/2}x^{n_1/2-1}}{\Gamma(n_1/2)\Gamma(n_2/2)[(n_1/n_2)x + 1]^{(n_1+n_2)/2}} \quad x > 0$$

za  $n_1 = 1, 2, \dots$  i  $n_2 = 1, 2, \dots$

$F$  distribucija se koristi za statističko zaključivanje o omjerima varijanci dviju normalnih populacija. Također,  $F$  distribucija se koristi za statističko zaključivanje o omjerima stopa dviju eksponencijalnih populacija.

Za sve koje žele znati više o distribucijama slučajnih varijabli putem inovativnih alata neka posjete web stranicu na poveznici <http://www.socr.ucla.edu/htmls/dist/>.

#### PITANJA ZA PONAVLJANJE:

1) Izaberite tvrdnju o t-distribuciji koja je ispravna:

- To je bimodalna distribucija.
- Koristimo je kada je standardna devijacija populacije poznata.
- Sadržava veći broj vrijednosti u svojim rubnim dijelovima (repovima) od normalne distribucije.
- Veći broj vrijednosti pada u centralno područje distribucije u usporedbi s normalnom distribucijom.

Gledajući informacijsku karticu o funkciji *t.test()* označite koje su od sljedećih tvrdnji istinite:

- ima mogućnost izvođenja t-testa samo na jedom uzorku
- funkcija *t.test* standardni je dio paketa *base*
- parametar kojim određujemo smjer testa nazivamo *direction*
- y je obavezni argument funkcije
- pretpostavka na x je da je on mjerjen na intervalnoj ili omjernoj skali
- argument funkcije *alternative* prihvata jedino potpune izraze *two sided*, *greater* ili *less*
- funkcija nema mogućnost testiranja lijevog testa
- funkciji obavezno moramo dati barem dva argumenta kako nam ne bi javljala pogrešku
- kôd *t.test(x, mu=0, alternative="two sided")* identično je kôdu *t.test(x)*.

2) Što od navedenoga vrijedi za normalnu distribuciju?

- u svakoj normalnoj distribuciji približno 68% podataka dolazi između 1 *sd* od srednje vrijednosti podataka



- medijan = mod = srednja vrijednost
- opisana putem jednog parametra
- standardna devijacija normalne distribucije je horizontalna udaljenost od sredine do jedne od točaka pregiba
- nije moguće svaku normalnu distribuciju pretvoriti u standardnu normalnu distribuciju

#### 4.3.3 Distribucija statistike uzorka (engl. *sampling distribution*)

Zamislimo da imamo mogućnost mnogo puta uzeti slučajne uzorke fiksne veličine  $n$  iz iste populacije od interesa. Za svaki takav uzorak računamo statistiku koja nas zanima, npr. aritmetičku sredinu, varijancu, standardnu devijaciju ili bilo koju drugu statistiku. Sve tako dobivene vrijednosti imaju određenu distribuciju koja opisuje teoretsko ponašanje statistike. Primjerice, sredine uzorka imaju normalnu distribuciju, varijance uzorka imaju  $\chi^2$  distribuciju, omjeri varijanci dva nezavisna uzorka imaju  $F$  distribuciju i tako dalje.

**4.3.3.1 Centralni granični teorem (engl. *Central Limit Theorem - CLT*)** Govori nam o teorijska raspodjeli, distribuciji sredina uzorka.

Ukoliko je distribucija populacije iz koje uzimamo slučajni uzorak veličine  $n$  (u teoriji mnogo uzorka fiksne veličine  $n$ ) normalna,

$$\mathcal{N}(\mu, \sigma)$$

tada je distribucija sredina uzorka veličine  $n$  normalna:

$$\mathcal{N}(\mu, \sigma/n_{12})$$

Zanima nas slučaj kada je distribucija populacije iz koje uzimamo slučajni uzorak veličine  $n$  (u teoriji mnogo uzorka fiksne veličine  $n$ ) *nije normalna*, kakva je distribucija sredina uzorka?

Odgovor na pitanje daje nam Centralni granični teorem (engl. *Central Limit Theorem - CLT*) koji je najvažniji teorem u primjenjenoj statistici i čini osnovu za inferencijalnu statistiku - zaključivanje o sredini populacije na temelju uzorka.

Centralni granični teorem kaže da ukoliko je uzorak veličine  $n$ , gdje je  $n \geq 30$ , uzet iz populacije sa sredinom  $\bar{x}$  i standardnom devijacijom  $\sigma$ , tada je distribucija sredina uzorka veličine  $n$  približno normalna:

$$\mathcal{N}(\mu, \sigma/n_{12})$$

Što imamo veći uzorak to je aproksimacija bolja.

Distribucija sredina uzorka ima varijancu jednaku varijanci populacije /  $n$ , i standardu devijaciju jednaku standardnoj devijaciji populacije / drugi korijen od  $n$  ( $SD[\bar{x}] = \frac{SD[x]}{\sqrt{n}}$ ), slika 92.

Što nam kaže Centralni granični teorem?

Ukoliko uzmemo sve moguće slučajne uzorke fiksne veličine  $n$  (vrijedi samo za velike uzorke) iz bilo koje populacije koja ima sredinu  $\mu$  i standardnu devijaciju  $\sigma$ , distribucija sredina uzorka ima parametre kako slijedi:

- 1) sredina:  $\mu_x = \mu$  (distribucija sredina uzorka ima sredinu jednaku populacijskoj) ili drugačije pisano, ukoliko je očekivanje sredine uzorka jednak populačijskoj sredini:

$$E[x] = \mu \text{ tada vrijedi } E[\bar{x}] = \mu.$$

- 2) standardna devijacija: (distribucije sredina uzorka ima standardnu devijaciju manju od populačijske)

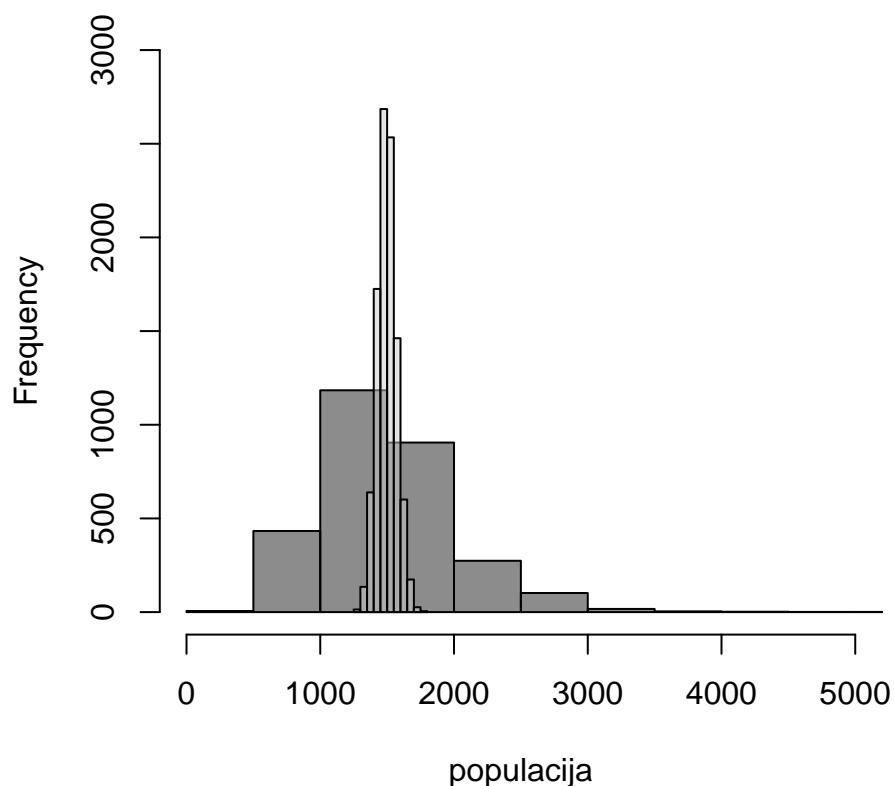
$$SD[\bar{x}] = \frac{SD[x]}{\sqrt{n}}$$

odnosno za varijancu vrijedi

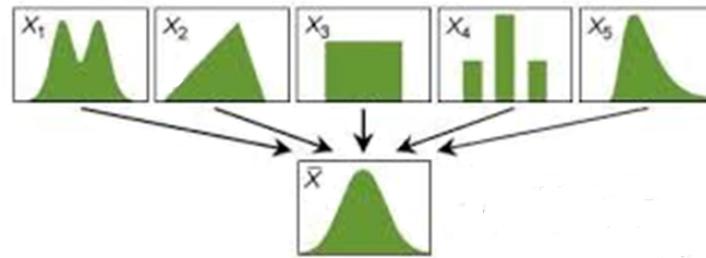
$$Var[\bar{x}] = \frac{Var[x]}{n}$$



### Centralni grani ni teorem, histogram populacije i sredina 10000 uzoraka



Slika 92: Grafički prikaz distribucije populacije i distribucije sredina 10000 uzoraka jednake veličine



Slika 93: Distribucija sredina uzoraka (Sampling distribution of the means); uzorci izvlačeni iz različito distribuiranih populacija

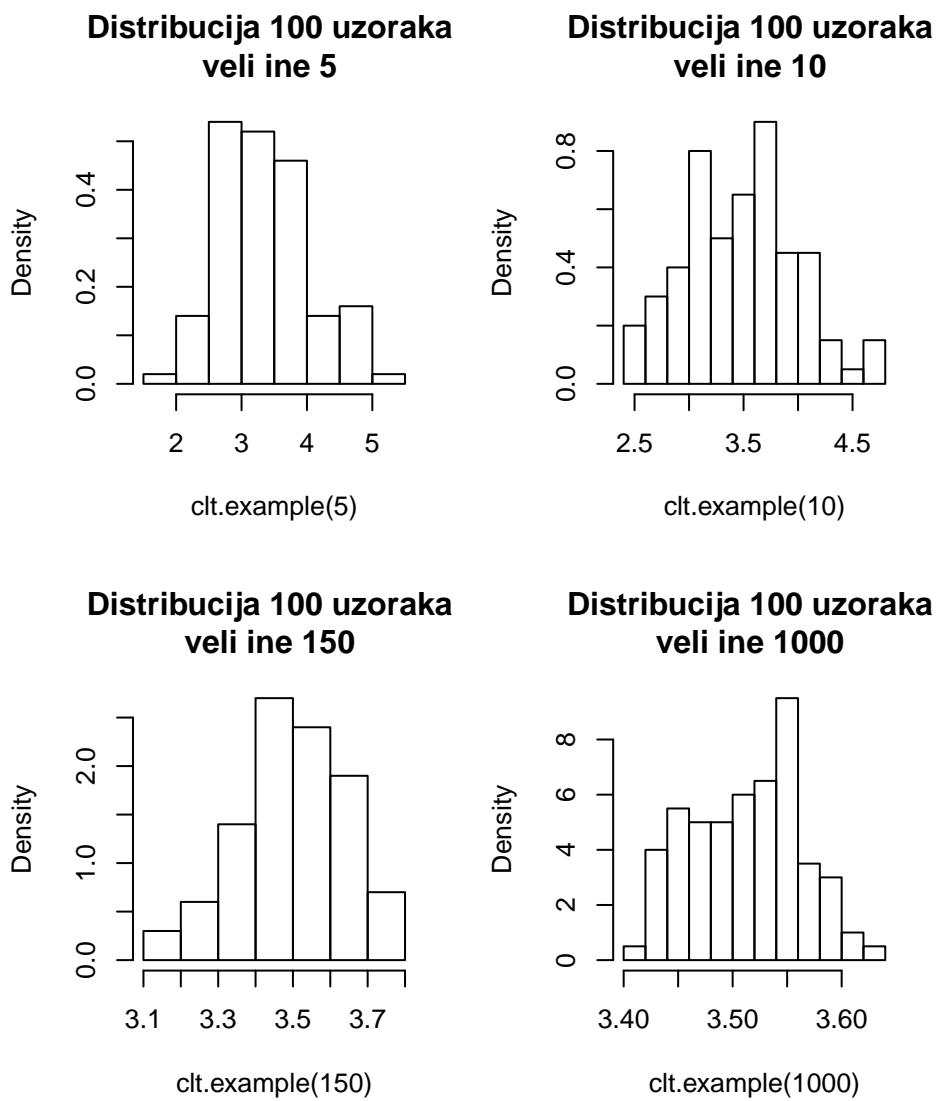
Distribucija sredina uzoraka je normalna bez obzira na distribuciju populacije iz koje smo uzorce vadili što je prikazano na slici 93.

Ipak, postoje slučajevi gdje ovaj teorem ne vrijedi ali to se odnosi na distribucije vjerojatnosti slučajne varijable o kojima na ovome tečaju neće biti govora i rijetko da ćete se sretati s njima u radu.

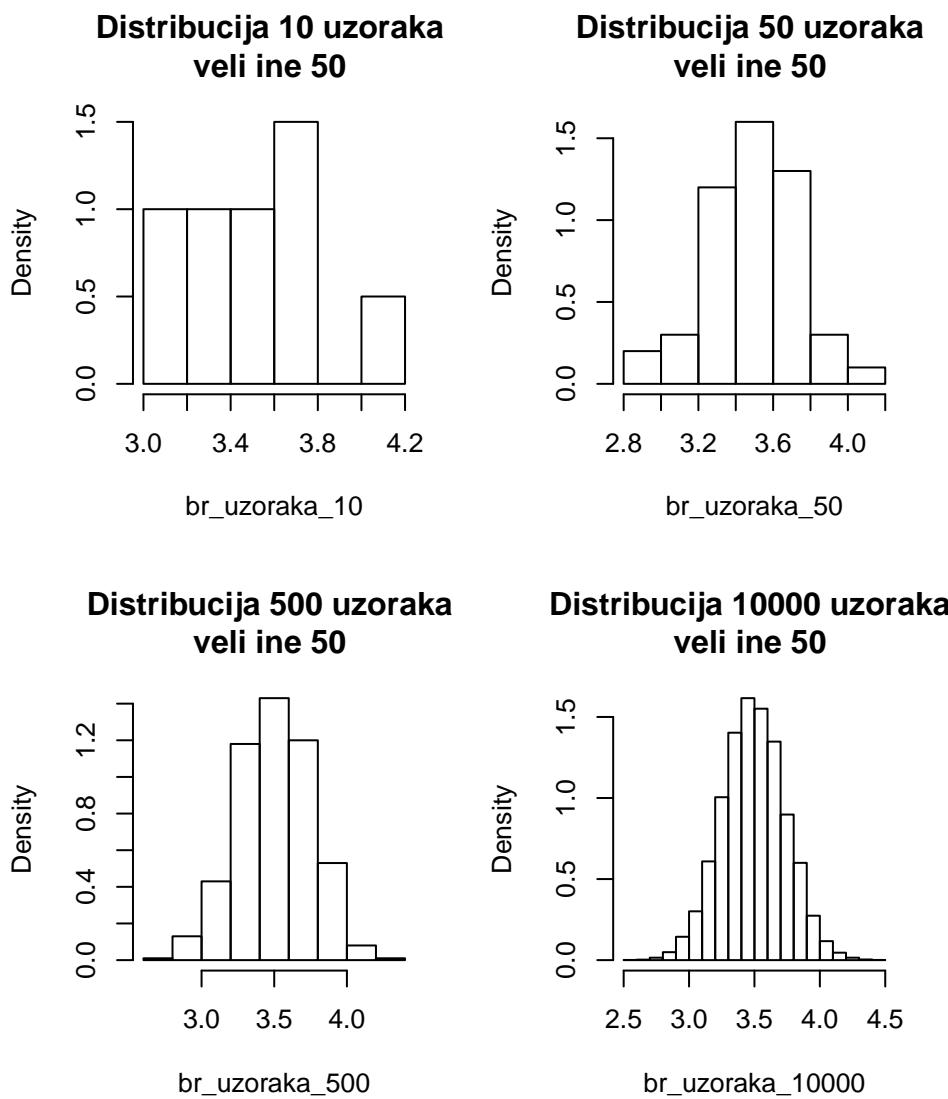
#### 4.3.3.2 Ponašanje distribucije sredina za različito velike uzorke

Ponašanje sredina uzoraka u ovisnosti o broju ponavljanja eksperimenta, proučite sliku ??.

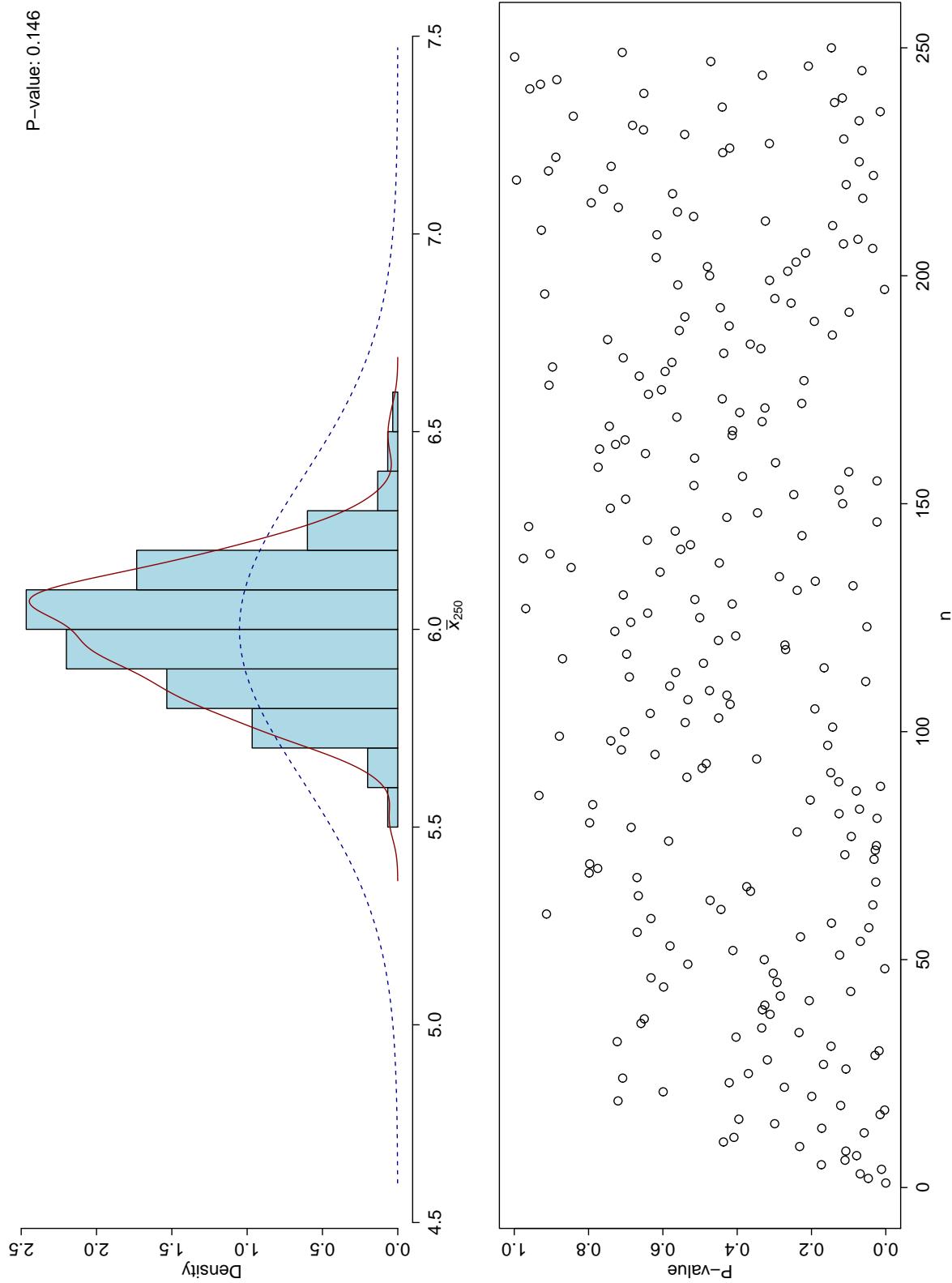
Proučite demonstraciju Centralnog graničnog teorema na poveznici: <http://vis.supstat.com/2013/04/central-limit-theorem/>. Na slici 96 prikazana je demonstracija teorema. Iako u ovome tečaju nismo stigli obraditi, kao rezultat demonstracije prikazuje i mjeru normaliteta dobivene distribucije uzoraka, p-vrijednost Shapiro-testa kojeg interpretiramo na način da daje vjerojatnost da je distribucija uzoraka normalna. Demonstracija ima svoje predodređene parametre koje možemo pogledati korištenjem `?clt.ani`. Na slici 97 prikazana je distribucija uzoraka veličine 300 uzetih iz  $\chi^2$  distribucije.



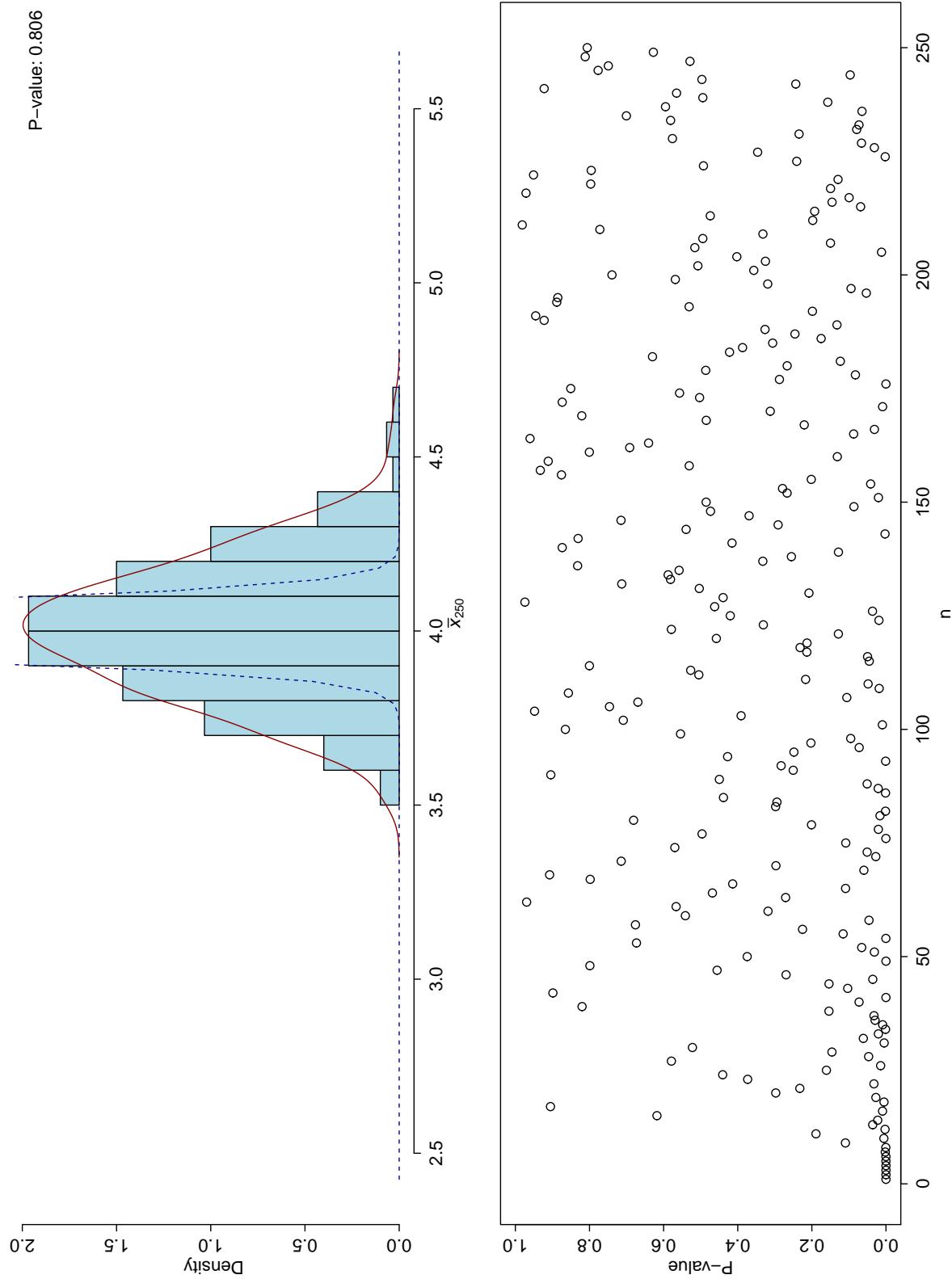
Slika 94: CLT: distribucija sredina 1000 uzoraka za različite veličine uzorka



Slika 95: CLT: distribucija sredina uzoraka jednake veličine



Slika 96: Centralni granični teorem; uzorci izvlaženi iz Poisson distribucije izvor: <https://www.rdocumentation.org/packages/animation/versions/1.1-3/topics/clt.ani>



Slika 97: Centralni granični teorem; uzorci izvlačeni iz Hi-kvadrat distribucije izvor: <https://www.rdocumentation.org/packages/animation/versions/1.1-3/topics/clt.ani>