Sissejuhatus

Kuigi on andmeanalüüsi ja statistikameetodeid, mille kasutamiseks pole arvutit vaja, on enamik andmeanalüüsi käigus kasutatavaist meetoditest sellised, mille kasutamiseks läheb tarvis arvutit. Arvuti aga ilma tarkvarata on üsna kasutu kast. Kursuse läbiviijana olen sellel aastal valinud kasutatavaks tarkvaraks R-i. R on vabavara, ta on üheks parimaks ja enamkasutatavaks statistikatarkvaraks maailmas ja õppejõu isiklik lemmik.

Kuidas saada endale R?

Uusima versiooni programmist võib maha laadida R-i koduleheküljelt **http://www.r-project.org/** (*Download-CRAN* – <vali server> – *Download R for Windows* – *base*). Samuti võib R-i koduleheküljelt leida ka ingliskeelse raamatu algajaile (*Help* – *Manuals* – *An Introduction to R*) ja spetsiaalsete ülesannete jaoks loodud statistikamooduleid, mis vaikimisi koos R-iga ei installeeru (hetkel saadaval rohkem kui 500 lisamoodulit). Lisamooduleid saab internetiühendusega arvutisse lisada vastavalt vajadusele kas programmi R menüüvalikut (*Packages -> Install Packages*) kasutades või vastavat käsku kasutades. Programmi R on võimalik kasutada Windowsi, Linuxi, Apple'i (Mac) OS X operatsioonisüsteemide (ja nii mõnegi muu operatsioonisüsteemi) peal. Standardinstallatsioon võtab umbes 85 Mb ruumi, koos täiendavate lisamoodulitega märgatavalt rohkem

Paljud kasutajad armastavad lisaks programmile R paigaldada oma arvutisse mõne R-i kasutamist hõlbustava kasutajakeskonna, nagu näiteks RStudio (soovi korral vaata www.rstudio.com).

Kirjandust

Põhjalikumat ülevaadet kui käesolev sissejuhatav material pakkuda suudab võib leda djärgmistest allikatest:

- www.r-project.org alt on võimalik leida algajaile mõeldud raamatut (*Help* – *Manuals* – *An Introduction to R*). Soovitav on iseseisvalt läbi proovida peatükis *Sample session* toodud käsud.
  R´i koduleheküljelt on võimalik kätte saada ka mitmeid teisi asjalikke raamatuid, vaata menüüd *Documentation-Contributed*.
- Peter Dalgaard (2002). *Intrductory Statistics with R*. Springer-Verlag. Heas stiilis raamat ühelt R-i loojatest.
- http://www.ms.ut.ee/mart/Rnaide/ – näiteid põhistatistikute, tabelite ja graafikute joonistamisest R-is (esialgu vaid ühe tunnuse jaoks), samuti on toodud ära programminäited peamiste andmemanipulatsioonide jaoks.

Alustuseks

R kui kalkulaator

R oskab arvutada. Seda väidet saab kontrollida, sisestades näiteks järgmine käsk (viipa „>“ pole tarvis sisestada, rea lõpus vajutage ENTER-klahvile):

```
> (2+3)*6
[1] 30
```

Tehteid saab teha (ja paljusid funktsioone kasutada) ka tervete arvujadadega ehk vektoritega korraga. Järgmise käsuga palume ruutjuured arvudest ühest kümneni:

```
> sqrt(1:10)
 [1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751 2.828427
 [9] 3.000000 3.162278
```

Näiteid käskudest ja funktsioonidest, mida R tunneb:

`> 2**8 või 2^8` – astendamine, 2^8;

`> sin(0.5*pi)` – näide trigonomeetrilise funktsiooni kasutamisest;

`> log(exp(10))` – funktsioon log leiab naturaallogaritmi arvust;

Muutujad

Ka kalkulaatorit kasutades tekib peatselt vajadus meeles hoida arvutuste tulemusi. R´is, nagu paljudes teistes programmeerimiskeelteski, võib kasutada sümboleid või sõnu (objektide nimesid) väärtuste hoidmiseks. Näiteks saab omistada *x*-le väärtuse 3 järgmise käsu abil:

```
> x=3
```

Selle käsu saamisel teostab R omistamise. Kui kõik läheb hästi, ei ilmu sealjuures ekraanile midagi. Edaspidi saab juba oma töös kasutada *x*-i samamoodi kui numbrit 3:

```
> x
[1] 3
> x+5
[1] 8
```

Meelde jätta võib muudki, kui vaid ühte arvu. Nime *x* taga võib peidus olla näiteks terve arvude vektor (kümne hiire kaalud); andmestik (saja katsehiire nimed, kaalud, pikkused, geneetiliste markerite väärtused,...); statistilise analüüsi tulemused, mis tallele pandud hilisemaks kasutamiseks jne.

Seda, milliseid nimesid oled objektide tähistamiseks kasutanud, saab vaadata käsu *ls()* abil:

```
> ls()
[1] "x"
```

Kasutu või tüütu objekti võib kustutada käsuga *rm()*:

```
> rm(x)
```

Kõik loodud objektid saab korraga kustutada käsuga `rm(list=ls())`.

R lubab omistamisel kasutada ka tema enda käskude nimesid. Juhul, kui R'i mõni käskudest ümber defineeritakse, võib mõnes situatsioonis tekkida segadus. Sestap tuleks võimaluse korral vältida R´i käskude („*c*“, „*t*“, „*lm*“, „*ls*“ jne) kasutamist muutujanimedena. Muutujate nimedeks sobivad hästi näiteks eestikeelsed salvestatud objekti kirjeldavad sõnad (RebasteArv, mudel2 jne).

Tähtis! R teeb vahet suurte ja väikeste tähtede vahel. Seega on „x“ ja „X“ kaks erinevat objekti. Samuti annab käsk *SQRT*(2) veateate, sest ruutjuure leidmise funktsioon on *sqrt* (väikesed tähed!).

Abiinfo (help)

Juhul, kui tead küll funktsiooni nime, mida kasutada soovid, kuid sooviks siiski täiendavat informatsiooni tema kohta, tipi „?“ ning sind huvitav funktsioon sinna järgi:

```
> ? median
```

R näitab seepeale antud käsu süntaksit ja näiteid sisaldavat ekraani.

Käsk `apropos("mean")` näitab aga kõiki R-i käske (ja ka sinu poolt loodud objekte) mille nimi sisaldab tähekombinatsiooni „mean“. Võib osutuda kasulikuks kui mäletad käsunime vaid osaliselt.

Programmi kirjutamine ja tehtud töö dokumenteerimine

Kuigi lihtsamaid käske ja arvutusi võib sisestada R-is otse käsurealt, on enamasti targem kirjutada käsud/kommentaarid eelnevalt valmis ja alles seejärel kirja pandud käsud käivitada. See võimaldab ühelt poolt töö käigus paratamatult tekkivaid vigu kergesti parandada, teiselt poolt võimaldab tehtavat tööd paremini dokumenteerida (saates andmeanalüüsi teostamiseks koostatud programmi teisele teadlasele, saab teine teadlane sama analüüsi täpselt korrata ja vajadusel otsida ka vigu, mida võisite töö käigus kogemata teha).

Programmi kirjutamiseks võib näiteks kasutada R-i enda editori (*File -> New Script*) või ka mõnda teist tekstiredaktorit (näiteks notepad'i). Juhul, kui kasutad R'i enda editori, saab soovitud käske kergesti käivitada – vali vaid programmilõik mida käivitada soovid ja vajuta CTRL+R (Windows-arvutil) või COMMAND+Enter (Apple).

Kõiki töö käigus R-le antud käske saab salvestada valides menüüst *File -> Save History* (aken „R Console“ peab olema aktiivne). Peale salvestamist tekib *.Rhistory* – lõpuga tekstifail, mida saab tekstieditoriga (näiteks notepad) avada ja redigeerida.

Antud hetkeni R-le antud käsud koos vastustega saab tekstifaili salvestada aktiviseerides akna „R Console“ ja valides menüüst *File -> Save to File*.

Kõiki töö käigus tehtud muutujaid, andmestikke jms saab korraga salvestada valides menüüst *File -> Save Workspace*. Kogu tekitatud muutujate, andmestike ja muu töökeskkonna saab siis hiljem taastada valiku *Load Workspace* abil.

Vaikimisi salvestatakse ja hakatakse faile otsima töökataloogist. Mõistlik on teha iga suurema projekti jaoks oma alamkataloog ja tööd alustades seada töökataloog viitama mainitud kataloogile. Seda saab teha kas valides menüüst käsu *File -> Change dir* või kasutades käsku *setwd*.

Ülesanded

Arvuta R’i abil:

a) $\frac{1}{1+0{,}00022} - (1 - 0{,}00022) =$ .....

b) Leia avaldise $x+x^x+\sin(x)$ väärtus kui x=1,79451
(Tee arvutus nii, et numbrit 1,79451 poleks vaja 4 korda sisestada....)

Vastus: .....

Esmatutvus andmestikuga.

Statistilise analüüsi tegemiseks läheb tarvis andmeid. Hiljem tutvume erinevate võimalustega sisse lugeda erinevat tüüpi andmeid R'i, aga alustame kõige lihtsamast – juba R'ile meelepäraseks tehtud andmete sisselugemisega.

Järgmine käsk loeb sisse kõik R'i andmestikud/objektid, mis olemas ühele veebilehele paigutatud failis andmefail.RData. Print-käsk sunnib R'i väljatrükkima kõigi sisseloetud andmestike/objektide nimed:

```
print(load(url("http://www.ms.ut.ee/mart/biomeetria2014/andmed.RData")))
```

Kui veateadet pole, siis on andmefailis peidus olevad andmestikud (ja muud objektid) edukalt R-i loetud. Juurdetekkinud objektide nimesid võid vaadata ka `ls()`-käsu abil.

Üks sisseloetud objektidest kannab nime tudengid. See ongi meie andmestik, mida üritame edaspidi veidi lähemalt uurida. Tegemist on Tartu Ülikooli (arstiteaduskonna) tudengite küsitlemisel saadud andmestikuga.

Andmestikus olevate tunnuste nimesid saab näha `names`-käsu abil:

```
names(tudengid)
```

Sisseloetud andmestiku võib soovi korral ka ekraanile trükkida (pole enamasti kuigi hea mõte):

```
tudengid
```

Targem on ehk vaadata andmestiku päist või viimaseid ridu. Päist saab näha käsuga `head`:

```
head(tudengid)
```

sabaotsa aga käsuga `tail`:

```
tail(tudengid)
```

ja esmast ülevaadet kõigi andmestikus olevate tunnuste kohta saab näiteks tellida `summary`-käsu abil:

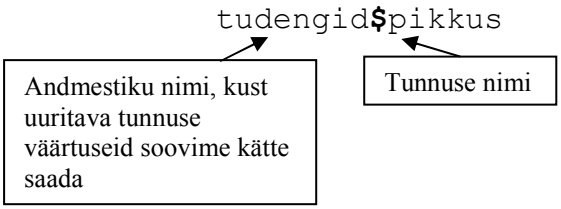
```
summary(tudengid)
```

Kogenumatele meeldib sageli ka käsu str abil antav ülevaade andmestikust (töötab ka muude objektide korral):

```
str(tudengid)
```

Millegi asjalikuma tegemiseks peame aga oskama andmestikust välja noppida meile vajaminevaid väärtuseid.

Ühe tunnuse väärtuseid – näiteks tudengite pikkuseid – võime kätte saada kirjutades andmestiku nime, dollarimärgi ($) ja seejärel tunnuse nime:



Kättesaadud väärtustega võime juba teha seda mida tahame, näiteks arvutada nende keskmise:

```
mean(tudengid$pikkus)
```

Sellist kohmakat võimalust tunnuse väärtuste nägemiseks kasutatakse eelkõige siis, kui tarvis läheb töötada mitme andmestikuga korraga. Aga praegu huvitume vaid ühestainsast andmestikust. Sellisel juhul on mõistlik R'le öelda, et nüüd töötame justnimelt andmestikuga tudeng. Sellisel juhul ei ole meil enam kohustust enam andmestiku nime tunnuse nime ette kirjutada. Töötamiseks kasutatavat andmestikku saab valida attach-käsu abil:

```
attach(tudengid)
```

ja nüüd piisab tudengite keskmise pikkuse või pikkuste mediaani leidmiseks lihtsalt käskudest

```
mean(pikkus)
median(pikkus)
```

Andmestiku „lahtiühendamiseks" on viisakas peale töö lõpetamist anda detach-käsk:

```
detach(tudengid)
```

Peale detach-käsu andmist lühike pöördumisviis enam ei tööta (annab veateate), pikk variant (andmestiku nimi$tunnuse nimi) jääb aga tööle:

```
mean(pikkus)
mean(tudengid$pikkus)
```

Kuna jätkame tööd sellesama tudengite andmestikuga, siis *attach*-i andmestik tudeng uuesti. Veendumaks, et `attach`-käsk töötas, leiame ka kõige lühema tudengi antud andmestikus:

```
min(pikkus)
```

ja joonistame ka histogrammi tudengite pikkusele:

```
hist(pikkus)
```

Veidi ilusama graafiku saaksime aga käsuga:

```
hist(pikkus, col="skyblue", xlab="Pikkus (cm)",
 ylab="sagedus", main="TÜ arstiteaduskonna tudengite pikkused")
```

Pideva või diskreetse tunnuse jaotust iseloomustavaid statistikuid ja jaotuse kuju iseloomustavaid graafikuid saab leida järgmiste funktsioonide abil (enamkasutatavad näitajad/joonised):

Statistikud

`min(pikkus)`	– miinimum
`max(pikkus)`	– maksimum
`range(pikkus)`	– miinimum ja maksimum
`mean(pikkus)`	– keskmine
`median(pikkus)`	– mediaan
`sd(pikkus)`	– standardhälve
`var(pikkus)`	– dispersioon (kasutatav ka kovariatsiooni leidmiseks)
`quantile(pikkus, 0.25)`	– 0.25-kvantiil (kas tead, mis see on?)
`summary(pikkus)`	– lühiiseloomustus (miinimum ja maksimum, alumine ja ülemine kvantiil, mediaan)
`length(pikkus)`	– vaatluste arv (vektori pikkus – arvesse lähevad ka tudengid, kellel uuritava tunnuse väärtus on puudu ehk NA)
`sum(!is.na(pikkus))`	– reaalselt olemasolevate vaatluste arv (mõõtmistulemuste arv ilma puuduvate väärtusteta)

Graafikud

`hist(pikkus)`	– pikkuste histogramm
`boxplot(pikkus)`	– karp-vurrud diagramm pikkustele

Puuduvad väärtused

Puuduva väärtuse tähis R-is on NA (Not Available). R käsitleb kohati puuduvaid väärtuseid peaaegu segavalt korrektselt:

```
> mean(kaal)
[1] NA
```

Tulemuseks on puuduv väärtus, sest osade tudengite kaal pole teada, ja seega pole võimalik leida ka tudengite keskmist kaalu – keskmiseks kaaluks saab R seega puuduva väärtuse ehk NA.

Otsitava statistiku väärtuse saame leida vaid olemasolevaid mõõtmistulemusi kasutades kui lisame lisaparameetri *na.rm=T*:

```
> mean(kaal, na.rm=T)
[1] 63.10427
```

Tunnusest väärtuste väljanoppimine

Tunnusest saab välja noppida ka üksikuid meid huvitavaid vaatluseid. Kui käsk

```
pikkus
```

trükib välja kõigi tudengite pikkused, siis esimese tudengi pikkuse saaksime kätte käsuga

```
pikkus[1]
```

ja kolme esimese tudengi pikkuseid näeksime käsu

```
pikkus[1:3]
```

abil.

Ülesanne: uuri mida saame kätte järgmise käsu abil:

```
pikkus[c(1:4, 10, 12:13)]
```

Vaatluseid võime välja noppida ka mingi tingimuse abil. Näiteks nende tudengite pikkused, kelle tervislik seisund on halb (tunnuse tervis väärtuseks on „halb“), saame kätte näiteks nii:

```
pikkus[tervis=="halb"]
```

Halva tervisega tudengite keskmise pikkuse saame leida aga käsuga

```
mean(pikkus[tervis=="halb"])
```

Kõrvalepõige:
Miks kautasime topeltvõrdust ülaltoodud näites? Proovi läbi järgmised käsud ja seleta, mida teeb ühekordne võrdusmärk ja mida kahekordne võrdusmärk:

```
x=1
x

x==2
x
```

Ülesanne

Tunnus olu näitab seda, mitu pudelit õlut tudeng nädala aja jooksul joob. Milliseid väärtuseid sellel tunnusel esineb ja kui palju ühte või teist väärtust esines saab teada `table` käsu abil:

```
> table(olu)
olu
ei joo     <1    1-5   5-12    >13
   266    265     92     30      7
```

Sinu ülesanne:
Leia õlut mittejoovate tudengite keskmine pikkus, õlut veidi joovate tudengite keskmine pikkus jne. Millised tulevad tulemused, mis võiks olla nähtu põhjuseks?

Andmestikust väljavõtte tegemine

Ka andmestikust saab nurksulgude abil väärtuseid välja noppida. Andmestiku puhul aga peame nurksulgudes määrama nii väljanopitavad tunnused kui ka vaatlused, kasutatav käsu süntaks on järgmine: andmestiku nimi[soovitavad objektid – näiteks tudengid, soovitavad tunnused]. Näiteks järgmine käsk näitab nelja esimese tudengi kolme esimese tunnuse väärtuseid:

```
tudengid[1:4, 1:3]
```

Kui soovime kas kõiki tunnuseid või kõiki objekte (tudengeid), siis tuleb vastav koht lihtsalt tühjaks jätta. Näiteks kolme esimese tudengi kõigi tunnuste väärtused saame käsuga:

```
tudengid[1:3,]
```

ja kahe esimese tunnuse väärtused kõigi tudengite jaoks:

```
tudengid[,1:2]
```

Kasutada võib ka sarnaseid tingimusi, nagu seda tegime eelmisel leheküljel. Näiteks võtame välja õlut armastavate (13 või enam pudelit nädalas joovate) tudengite andmed:

```
tudengid[olu==">13",]
```

Saadud andmestikuga võib juba midagi soovikohast ette võtta, näiteks võime salvestada meestudengite (tunnuse sugu kodeering: 1-naine, 2-mees) andmed eraldi andmestikku:

```
mehed = tudengid[sugu==2,]
```

Nüüd on meie käsutuses ka teine andmestik (mehed), mis sisaldab ainult meestudengite andmeid. Võrdle näiteks:

```
mean(tudengid$pikkus)
mean(mehed$pikkus)
```

Joonistame ka paari viimase tunnuse jaoks nn hajuvusgraafikud nii kõigi tudengite pealt kui ka ainult meestudengeid sisaldava andmestiku pealt.

Joonis kõigi tudengite pealt kasutades tunnuseid 17-20 (pikkus, kaal, SVR- süstoolne vererõhk ja DVR-diastoolne vererõhk):

```
plot(tudengid[,17:20])
```

Sarnane joonis ainult meestudengite pealt:

```
windows()
plot(mehed[,17:20])
```

Ülesanded

1. Millise käsuga saame kätte seitsme esimese tudengi andmed?

2. Mida teeb järgmine käsk:

```
tudengid[sugu==1 & pikkus>180,]
```

3. Kuidas (millise käsuga) saab leida 170cm-st lühemate meeste andmed?

4. Kui palju naisi ja kui palju mehi on lühemad kui 175cm?

5. Joonista tunnusele *olu* (mis antud andmestikus on esitatud kui järjestustunnus) tulpdiagramm. Kuidas tulpdiagrammi joonistada, seda uuri soovitatud materialide lehelt: http://www-1.ms.ut.ee/mart/Rnaide/

Andmete sisestamine programmi abil

Vahel soovime mõningaid väärtuseid ka käigupealt sisestada. Lihtsaim viis vaatluste (või arvujada) sisestamiseks on kasutada funktsiooni *c*. Loome vektori *h* mis sisaldab viie puu kõrguseid:

```
> h = c(20, 12, 14, 16, 33)
> h
[1] 20 12 14 16 33
```

Andmed, mida vektoris hoitakse, ei pea olema arvulised. Me võime tekitada ka vektori, mis sisaldab uuritud puude liigilist kuuluvust näitavaid andmeid (kasuta jutumärke, kui tahad sisestada või kasutada tekstina antavat informatsiooni):

```
> liik = c("Kuusk", "Kask", "Kask", "Kask", "Kuusk")
> liik
[1] "Kuusk" "Kask"  "Kask"  "Kask"  "Kuusk"
```

Üks võimalus andmemaatriksit luua on teha seda kasutades olemasolevaid vektoreid ehk üksiktunnuste väärtuseid. Üksiktunnused saab andmestikuks kokku panna *data.frame* käsu abil:

```
> minuandmed=data.frame(liik, h)
> minuandmed
   liik  h
1 Kuusk 20
2  Kask 12
3  Kask 14
4  Kask 16
5 Kuusk 33
```

ja vaid kaskede andmete nägemiseks võime näiteks kasutada käsku

```
minuandmed[liik=="Kask",]
```

Väärtuste automaatne genereerimine

Vahel tekib vajadus kiiresti genereerida tunnuse väärtused mingi kindla skeemi kohaselt. Näiteks järjestikustest arvudest koosnevat vektorit saab tekitada järgmise käsu abil:

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

Näiteid kasutamisest:

```
h[1:3]
for(i in 1:10) {print("R on kole keeruline")}
```

Käsuga seq saab tekitada järjestikuseid arve mingi etteantud sammuga:

```
> seq(6,10,0.5)
[1]  6.0  6.5  7.0  7.5  8.0  8.5  9.0  9.5 10.0
```

või saab tema abil tekitada etteantud pikkusega vektori:

```
> seq(0,1,length=4)
[1] 0.0000000 0.3333333 0.6666667 1.0000000
```

Käsuga rep saab esimest argumenti korrata soovitud arv kordi:

```
> rep(3,10)
 [1] 3 3 3 3 3 3 3 3 3 3
> rep(c(1,7,9),2)
[1] 1 7 9 1 7 9
> rep(c("Tallinn","Tartu"),c(3,2))
[1] "Tallinn" "Tallinn" "Tallinn" "Tartu" "Tartu"
```

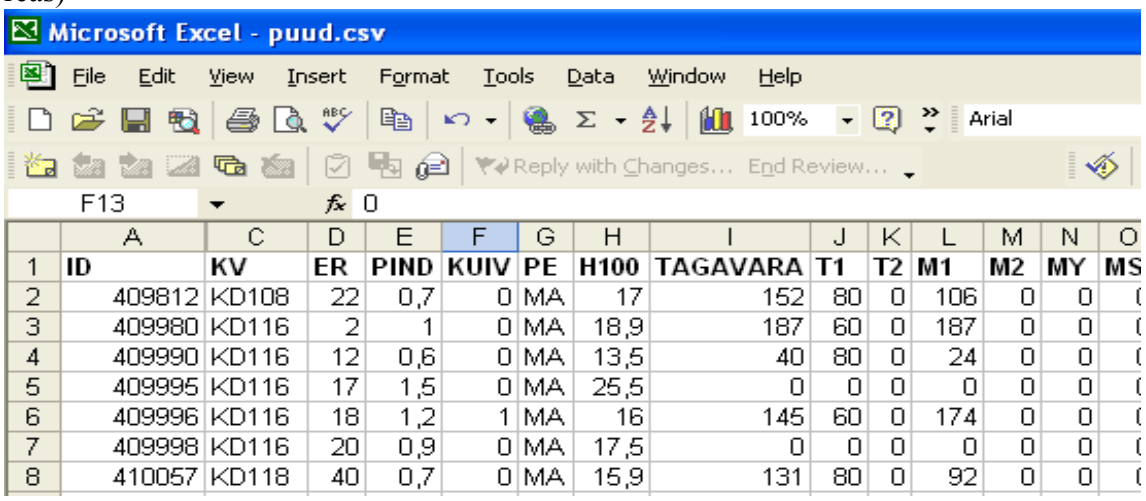
Ülesanded

1. Pika, 10 aastat kestnud uuringu käigus püüti lõksudega metsast kinni 123 halli karvaga hiirt, 156 täpilise karvaga hiirt ja 23 mummulise karvaga hiirt. Tekita vektor, mis sisaldaks kinnipüütud hiirte karvavärvi (iga hiire värv eraldi kirjas).

Lisa 1. Andmete importimisest

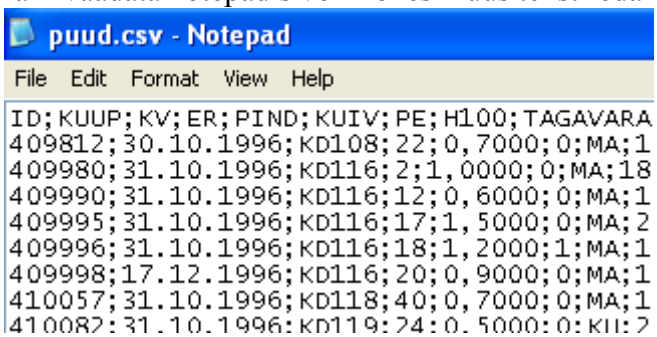
Suuremate andmestike sisestamiseks pole R just kõige sobivam vahend. Enamasti sisestataksegi andmeid kas spetsiaalselt andmete sisestamiseks mõeldud tarkvara abil või kasutades mõnda tabelarvutusprogrammi (näiteks Excelit). Andmestiku importimisel tabelarvutusprogrammist (Excelist) on soovitav andmefail esmalt salvestada CSV-formaadis (Comma Separated Values), näiteks faili "C:\puud.csv" (*File -> Save As ->* muuda *Save As type* aknas failitüüp „*CSV (Comma Delimited) (*.csv)*"-ks).

Andmestik Excelis (tunnuste nimed – lühikesed, soovitavalt ühesõnalised – esimeses reas)



	A	C	D	E	F	G	H	I	J	K	L	M	N	O
1	ID	KV	ER	PIND	KUIV	PE	H100	TAGAVARA	T1	T2	M1	M2	MY	MS
2	409812	KD108	22	0,7	0	MA	17	152	80	0	106	0	0	
3	409980	KD116	2	1	0	MA	18,9	187	60	0	187	0	0	
4	409990	KD116	12	0,6	0	MA	13,5	40	80	0	24	0	0	
5	409995	KD116	17	1,5	0	MA	25,5	0	0	0	0	0	0	
6	409996	KD116	18	1,2	1	MA	16	145	60	0	174	0	0	
7	409998	KD116	20	0,9	0	MA	17,5	0	0	0	0	0	0	
8	410057	KD118	40	0,7	0	MA	15,9	131	80	0	92	0	0	

Andmestik peale csv-faili salvestamist näeb välja järgmine (soovi korral saame csv-faili vaadata notepadis või mõnes muus tekstiredaktoris):



Salvestatud andmestiku lugemiseks R-i tuleb anda näiteks järgmine käsk:

```
puud=read.csv2("C:/puud.csv", header=T)
```

ja vaatamaks, kas andmete sisselugemine läks valutult:

```
head(puud)
names(puud)
```

Vahel tekivad probleemid – andmeid ei loeta R-i õigel kujul, saame veateateid vm. Üks põhjus – andmefail sisaldas tekstikirjeid, mis sisaldasid keelatud märke (Näiteks sümbolit „;"). Vahel aga on põhjuseks see, et erinevad Excelid võivad (erinevates masinates) teha erinevaid csv-faile. Vahel pannakse andmeväljade vahele semikooloni (;) asemel näiteks koma (,) ja kümnendkohtade eraldaja arvus kasutatakse hoopis

punkti (.) – seda näeme avades salvestatud csv-faili notepadis/tekstiredaktoris. Sellisel juhul tuleb andmete sisselugemiseks R-i anda hoopis käsk

```
puud=read.csv2("C:/puud.csv", header=T, dec=".", sep=",")
```

Pange tähele:

- sisseloetud andmestik tuleb kuhugi salvestada, kui soovime teda hiljem ka kasutada! Siin salvestasime ta andmestikuks nimega "puud".
- Failinimes on kasutatud tagurpidi kaldkriipse („/“)!
- Käsuga *sep*= määratakse sümbol, mis eristab eri tunnuste väärtuseid (Excel võib salvestamisel kasutada eraldajana nii sümbolit "," kui ka ";")
- *dec*= parameetri abil saab määratleda sümboli, mis tähistab arvus koma (Näiteks Excel kasutab kümnendkohtade tähistamiseks vahel sümbolit "." ja vahel sümbolit ",").
- Parameeter *header=T* ütleb arvutile, et tunnuste nimed on kirjas tekstifaili esimeses reas.

R-i andmestikku saab samuti salvestada tekstifaili kasutades käsku

```
write.table(andmestik, "C:/andmed/uustabel.csv",
     sep=",", dec=".", header=T, row.names=F)
```

Teine võimalus

Alternatiivne võimalus oleks installeerida lisamoodul xlsx, milles olevad funktsioonid võimaldavad nii lugeda kui kirjutada uuemate Exceli versioonide poolt kasutatavaid xlsx-faile. Kasutusnäiteid vaata lisamooduli dokumentatsioonist, http://cran.r-project.org/web/packages/xlsx/xlsx.pdf

Märkused:

- Kasutades lisamoodulit foreign saab R-i (otse) lugeda ka statistikapakettide S-Plus, Stata, Minitab ja SPSS andmefaile.

Täiendavat infot vaata R-i koduleheküljelt (*Manuals→R Data Import/Export*)