Ulrich Pötter Wintersemester 2007/2008
Statistische Techniken der Sozialforschung I

Arbeitsblatt 2

1) Die Welle 2003 des PSID-Datensatzes kann mit

```
library(foreign)
dat <- read.spss("D:/Stattech/daten/pequiv_2003.sav",
to.data.frame=T, use=F)
```

eingelesen werden. Wir verschaffen uns einen ersten Überblick über einige Variable:

```
Geschlecht: D11102LL
Alter: D11101_2003
Arbeitsmarktbeteiligung: E11102_2003
(Lohn-) Einkommen: I11110_2003
```

Dies sind die Namen, die im Codebuch vergeben worden sind. Wie heißen die entsprechenden Variablen in den eingelesenen Daten `dat`? (Benutzen Sie `names(dat)`). Was würde passieren, wenn auch noch die Daten für das Jahr 2001 eingelesen würden?

2) `summary(dat$D11102LL)` gibt einen ersten Überblick über Mittelwerte, Quantile und fehlende Werte.

`mean(dat$D11102LL, na.rm=T)` berechnet den Mittelwert. Mit `na.rm` werden vor der Berechnung fehlende Werte ausgeschlossen. Was ist der Frauenanteil (in Prozent)?

Häufigkeitstabellen (contingency tables) lassen sich mit `table()` erstellen:

```
table(dat$D11102LL)
```

Hier werden fehlende Werte automatisch ausgeschlossen. Sollen auch die fehlenden Werte ausgezählt werden, kann man

```
table(dat$D11102LL, exclude=NULL)
```

benutzen.

`barplot(table(dat$D11102LL))` liefert ein Säulendiagramm, `pie(table(dat$D11102LL))` einen Kuchen. Alle Plot-Befehle können mit entsprechenden weiteren Parametern versehen werden, etwa:

```
barplot(table(dat$D11102LL), names.arg=c("Männer", "Frauen"),
main="Anteil Männer und Frauen", sub="PSID 2003",
col=c("lightblue", "lightsalmon1"), cex.main=1.5)
```

3) Man muss nicht immer die Form `dat$D11102LL` benutzen, um auf die Variablen eines Datensatzes zu verweisen. Die Variablennamen in `dat` sind nach einem `attach(dat)` auch direkt zugreifbar. Diese Bindung lässt sich mit `detach(dat)` wieder rückgängig machen.

```
attach(dat)
summary(D11102LL)
detach(dat)
```

Genaugenommen macht sich R, nachdem es auf einen Variablen- oder Funktionsnamen gestoßen ist, nach diesem in den durch `search()` einsehbaren Paketen und Objekten auf die Suche. Mit `attach(dat)` wird die Variable `dat` (hier: der eingelesene Datensatz) zu dieser Liste hinzugefügt, mit `detach(dat)` wieder entfernt. Z.B.:

```
D11102LL
search()
attach(dat)
search()
D11102LL
detach(dat)
search()
```

Eine Auflistung der selbstdefinierten Objekte und Funktionen erhalten Sie mit `ls()`. Um Objekte/Variablen aus dem Speicher zu entfernen, benutzen Sie `rm()` (gleichbedeutend mit `remove()`):

```
a <- c(1,2,3)
ls()
rm(a)
ls()
```

4) Zur einfacheren Handhabung kann es sinnvoll sein, die Variablen umzubenennen.

```
names(dat)
names(dat)[1]
names(dat)[1] <- "PID"
names(dat)
```

```
names(dat)[2] <- "HHID"
names(dat)[39] <- "GENDER"
names(dat)[38] <- "AGE03"
```

```
table(AGE03)
```

Ein derart modifizierter Datensatz kann bspw. mit
`write.table(dat, "D:/Stattech/daten/p03.dat", row.names=FALSE)`

in eine Datei geschrieben und mit
`Y <- read.table("D:/Stattech/daten/p03.dat", header=TRUE)`
wieder eingelesen werden.

5) Verschaffen Sie sich einen ersten Überblick über die Verteilung der Variablen `AGE03`. Benutzen Sie dazu zunächst `summary()`. `table()` ist nun nicht mehr sehr gut lesbar. Benutzen Sie daher zunächst `barplot(table())`. Schlagen Sie im Codebuch die Besonderheiten der Kodierung der Variablen nach.

Eine weitere Möglichkeit der Darstellung sind Kernschätzer. Probieren Sie `plot(density(AGE03, na.rm=T))`.

Man kann nun auch die Altersverteilung der Männer und Frauen vergleichen:

```
plot(density(AGE03[GENDER==1], na.rm=T), col="blue",
main="Altersverteilung", xlab="Alter", ylab="", lwd=2)
lines(density(AGE03[GENDER==2], na.rm=T), col="red", lwd=2)
legend(60,0.017,c("Männer","Frauen"), lty=c(1,1),
col=c("blue","red"), lwd=c(2,2))
```

6) Schlagen Sie im Codebuch den Namen und die Definition der Variable 'Individual Labour Earnings' nach. Berechnen Sie deren Durchschnitt. Wieviele fehlende Angaben gibt es?

Berechnen und zeichnen Sie einen Kern-Schätzer für das Einkommen. Hinweis: Bei der Berechnung des Dichte-Schätzers lässt sich das zu berücksichtigende Intervall einschränken:
`plot(density(I11110_2, na.rm=TRUE, to=100000))`

7) Die Berechnungen lassen sich auch auf Fälle beschränken, bei denen der Wert einer anderen und/oder derselben Variablen gewisse Bedingungen erfüllt:

```
mean(I11110_2[GENDER==1], na.rm=TRUE)
mean(I11110_2[I11110_2>0], na.rm=TRUE)
mean(I11110_2[I11110_2>0 & GENDER==2], na.rm=TRUE)
```

Das ist automatisierbar:

```
tapply(I11110_2, GENDER, mean, na.rm=TRUE)
```

führt `mean(I11110_2, na.rm=TRUE)` einzeln für die durch die unterschiedlichen Werte in `GENDER` definierten Gruppen aus.

8) Der ungefähre Speicherverbrauch eines R-Objekts kann mit `object.size()` ausgegeben werden, bspw. `object.size(dat)`. Der momentane und der bisherige maximale Speicherverbrauch der R-Sitzung wird mit `gc()` angezeigt.

Für den Speicherverbrauch ist auch zu beachten, dass `attach(dat)` eine Kopie von `dat` erstellt, auf der dann operiert wird. Das ist natürlich sicherer, weil man sich nicht unbeabsichtigt Daten überschreiben kann, erhöht aber den Speicherbedarf:

```
object.size(dat)
gc()
detach(dat)
gc()
```