

Einige SPSS Befehle

Hinweise: SPSS-Befehle können sich über mehrere Zeilen erstrecken und werden mit einem Punkt abgeschlossen. Ein neuer Befehl muss auf einer neuen Zeile beginnen. Die meisten Befehle, Unterbefehle und Schlüsselwörter können auf die ersten drei oder vier Buchstaben abgekürzt werden. SPSS kennt eine Vielzahl an Operatoren und Funktionen, die in Argumenten (*exp*) verwendet werden können. Darunter befinden sich: arithmetische Operatoren wie +, -, *, /, ** (Exponent); arithmetische Funktionen wie *ln(arg)*, *exp(arg)*; relationale Operatoren wie =, >, <, >=, <=, <> (nicht gleich, auch: ~ =) (auch: *eq*, *gt*, *le*, *ge*, *le*, *ne*); logische Operatoren wie *and*, *or*, *not* (auch: *&*, *|*, *~*); Missing-Value-Funktionen wie *missing(arg)* (wahr falls Wert fehlend), *value(arg)* (benutzerdefinierte Missing-Values werden als gültig behandelt); etc.

Anmerkungen zur Notation auf diesem Blatt: *var* steht für eine bestehende Variable, *varlist* für *eine oder mehrere* Variablen (z.B. „v1 v2 v3“, „v1 to v3“, „all“ [alle Variablen] ...); *newvar* für eine neue oder schon bestehende Variable (die ggf. überschrieben wird); *num* für eine Zahl, *numlist* für *eine oder mehrere* Zahlen (z.B. „1 2 3“, „1 thru 3“ ...); *exp* für einen mathematischen, logischen oder sonstigen Ausdruck (z.B. „3*4“, „var<>9“, „range(var, 1, 8)“ ...)

Literaturhinweis: Wittenberg, R., H. Cramer (2000). Datenanalyse mit SPSS für Windows. Stuttgart: Lucius&Lucius.

Bemerkungen

**Text.*

Bemerkungen zwischen Befehlen anbringen.

... /* *Text* */ ...

Bemerkungen innerhalb eines Befehls anbringen.

Daten-Handling

```
get file='Laufwerk:\pfad\datei.sav'  
  /keep=varlist /drop=varlist.
```

Lesen von SPSS-Daten, Variablen können dabei behalten (keep) oder ausgelassen (drop) werden (falls keep/drop fehlt, werden alle Variablen gelesen).

```
save out='Laufwerk:\pfad\datei.sav'  
  /keep=varlist /drop=varlist.
```

SPSS-Daten speichern.

```
save translate out=datei.dat  
  /type=typ /keep=varlist  
  /drop=varlist /fieldnames  
  /map /replace.
```

Speichern der Daten in einem durch *typ* spezifizierten Format (z. B. *tab* für tab-delimited ASCII). Der optionale Unterbefehl *fieldnames* schreibt die Variablennamen in die erste Zeile der Daten, *map* liefert eine Auflistung der gespeicherten Variablen, und *replace* erlaubt das Überschreiben einer bestehenden Datei mit gleichem Namen.

```
data list list (separator)  
  file=datei.dat  
  /varlist (formatlist).
```

Import von Daten in freiem ASCII-Format mit *separator* als Trennzeichen für die Variablen (z.B., *tab* oder „“). Mit dem optionalen Befehl *formatlist* können die Anzeigeformate der Variablen spezifiziert werden.

```
compute newvar=exp.
```

Bildung einer neuen Variable.

<code>recode varlist (numlist=num) ... into newvarlist.</code>	Umcodierung der Werte einer Variable (bzw. mehrerer Variablen). Bei Verwendung von <code>into</code> werden die umcodierten Daten in eine neue Variable geschrieben.
<code>if exp1 newvar=exp2.</code>	Bildung von Variablenwerten gemäss <code>exp2</code> für Fälle, für die <code>exp1</code> wahr ist.
<code>do if exp1. ... Transformationsbefehle ... else if exp2. ... mehr Transformationsbefehle ... end if.</code>	<i>Transformationsbefehle</i> werden ausgeführt für Fälle, für die <code>exp1</code> wahr ist; <i>mehr Transformationsbefehle</i> für Fälle, für die <code>exp2</code> wahr ist.
<code>do repeat platzhalter=varlist. ... Transformationsbefehle ... end repeat print.</code>	<i>Transformationsbefehle</i> werden der Reihe nach für alle Variablen in <code>varlist</code> ausgeführt.
<code>missing values varlist (num).</code>	Bestimmung des Wertes <code>num</code> als Missing-Value. Hinweis: <code>num</code> kann ein Wertebereich sein (z.B. <code>lowest thru 0</code> um alle negativen Werte und 0 auf missing zu setzen).
<code>variable labels var1 'label1' /var2 'label2'</code>	Variablen beschriften.
<code>value labels varlist x'label1' y'label2' z'label3'... .</code>	Werte einer Variable beschriften.
<code>add value labels</code>	Zusätzliche Werte einer Variable beschriften (die schon bestehenden Beschriftungen der Variable bleiben erhalten).
<code>formats varlist (fw.d).</code>	Anzeigeformate spezifizieren (wobei <code>w</code> der Gesamtbreite und <code>d</code> der Anzahl Dezimalstellen entspricht).
<code>select if exp.</code>	Auswahl von Fällen, für die <code>exp</code> gilt (z.B. <code>sel if sex=1</code> oder <code>sel if age>=25 and age<40</code>). Alle anderen Fälle werden aus dem Arbeitsspeicher gelöscht.
<code>temporary. sel if exp.</code>	Temporäre Auswahl von Fällen (gültig für <i>eine</i> nachfolgende Prozedur).
<code>filter by ivar. ... Prozeduren ... filter off.</code>	Auswahl von Fällen, für die <code>ivar=1</code> gilt (<code>ivar</code> ist eine binäre Indikatorvariable). Nach <code>filter off</code> stehen wieder alle Fälle zur Verfügung.
<code>sort cases by var.</code>	Aufsteigende Sortierung der Daten nach den Werten von <code>var</code> .
<code>split file by var. ... Prozeduren ... split file off.</code>	Unterteilung der Daten in Gruppen gemäss den Werten von <code>var</code> (die Daten müssen nach <code>var</code> sortiert sein). Die nachfolgenden Prozeduren werden für die Gruppen getrennt durchgeführt. <code>split file off</code> hebt die Teilung auf.
<code>weight by var.</code>	Gewichtung der Fälle mit den Werten von <code>var</code> .

`execute.`

Befehl um Transformationsbefehle abzuschliessen (nur selten benötigt, da der Befehl bei statistischen Prozeduren automatisch vorgeschaltet wird).

Univariate und bivariate Statistik, Grafiken

`fre varlist.`

Häufigkeitstabellen für die Variablen in *varlist*.

`fre varlist /hist /bar /pie
/sta=mean median mode std.`

Häufigkeiten inkl. Histogramm, Säulendiagramm, Kreisdiagramm und ausgewählte univariate Statistiken.

`graph`

Befehl zur Erzeugung von Grafiken.

`desc varlist.`

Univariate Statistiken (arith. Mittel, Standardabweichung, Min., Max., Anzahl Fälle).

`means var1 by var2.`

Mittelwerte für *var1* nach Subgruppen.

`examine var1 by var2 /plot=box.`

Boxplots für *var1* nach *var2*.

`cross var1 by var2
/cells=count tot col row.`

Kreuztabelle mit absoluten (count), relativen (tot) und bedingten Häufigkeiten (col und row).

`cross var1 by var2 /sta=Masse.`

Kreuztabelle mit Zusammenhangsmassen. Nominal: `chi cc phi lambda uc` (χ^2 , Kontingenzkoeff., ϕ /Cramer's V, Guttman's λ , Unsicherheitskoeff.); ordinal: `gamma d btau` (Goodman und Kruskal's γ , Sommer's d , Kendall's τ_b); metrisch: `corr` (lineare und Rangkorrelation); gemischt: `eta`

`corr varlist /missing=listwise.`

Korrelationsmatrix mit linearen Korrelationskoeffizienten. Durch `missing=listwise` werden nur Fälle mit gültigen Werten für alle Variablen in *varlist* berücksichtigt.

`nonpar corr varlist /print=both`

Korrelationsmatrix mit Rangkorrelationen und Kendall's τ_b .

`t-test /var=var1
/groups=var2(num1 num2).`

T-Test auf Mittelwertsdifferenzen für *var1* nach *var2* (mit Ausprägung *num1* und *num2*).

Regression

`regression /dependent=var
/method=enter varlist.`

Schätzung eines linearen Regressionsmodells (OLS).

`plot /format=reg
/plot var1 with var2.`

Streudiagramm mit Regressionsgerade.

`logistic regression
var with varlist.`

Schätzung einer logistischen Regression.