Henning Lohmann
Lehrstuhl für Empirische Sozial- und Wirtschaftsforschung
Wirtschafts- und Sozialwissenschaftliche Fakultät
Universität zu Köln

Einführung in Stata 9.0
(Mai 2007)

0. Notation und wichtige Tasten

Notation von Stata-Befehlen:

Stata-Befehle sind in anderer Schrift fett geschrieben (Beispiel: `regress`). Die meisten Befehle können auch abgekürzt werden. Unterstreichungen zeigen die jeweils kürzeste Schreibweise an (Beispiel: `reg`ress). In vielen Fällen werden Befehle sowohl in allgemeiner Form und als konkretes Beispiel aufgeführt. Das Beispiel folgt jeweils nach einem Pfeil.

save *filename* [, replace]
→ save *fppfadfptest_neu.dta***

Mehr zur Notation von Befehlen in Abschnitt 4.

Wichtige Tasten und Benennung einzelner Zeichen:

‚Bild-oben' (=Anzeigen des vorherigen Befehls im Eingabefenster)

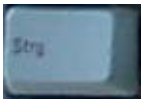
‚Bild-unten' (=Anzeigen des nächsten Befehls im Eingabefenster)

‚Escape' (=Löschen eines Befehls im Eingabefenster)


‚Backspace' (=Löschen des Zeichens links vom Cursor)

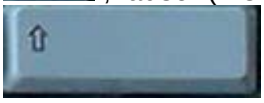
‚Entfernen'/‚Delete' (=Löschen des Zeichens rechts vom Cursor)


‚Eingabe'/‚Return'



‚Steuerung'/‚Control'

‚Pause' (=Unterbrechen eines gerade ausgeführten Befehls oder do-files)

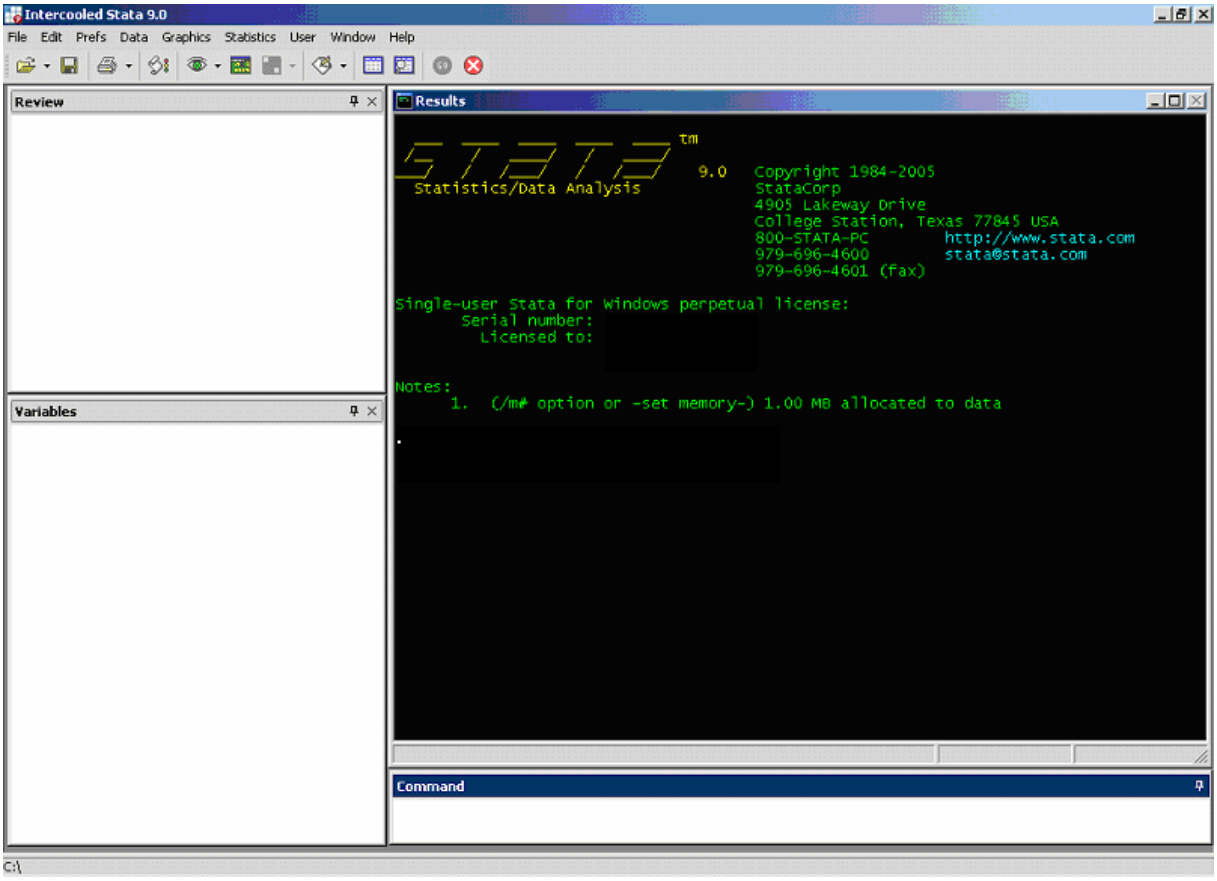

‚Groß'/‚Shift'

Zur Benennung einzelner Zeichen:
^ → Zirkumflex oder ‚Dach'
~ → Tilde oder ‚Schlange'
| → Verkettungszeichen oder ‚senkrechter Strich'
/ → Schrägstrich oder ‚Slash'
\ → umgekehrter Schrägstrich oder ‚Backslash'

1. Fenster und Menüs

Beim Öffnen von Stata sieht man vier Fenster:

- das *Eingabefenster (Kommandozeile)*, in das man Befehle eingeben und über die ‚Return'-Taste direkt ausführen kann;
- das *Ergebnisfenster*, in die aktuell ausgeführten Befehle, die Ergebnisse, Fehlermeldungen etc. angezeigt werden;
- das *Review-Fenster*, in dem alle bislang ausgeführten Befehle angezeigt werden und durch doppeltes Anklicken wiederholt werden können;
- das *Variablenfenster*, in dem die Variablen des aktuell verwendeten Datensatzes angezeigt werden.



Weiter gibt es eine *Menüzeile* und eine *Zeile mit Buttons*, über die bestimmte Befehle oder Aktionen (Speichern, Drucken usw.) ausgelöst werden können. Fast alle Funktionen, die im Menü enthalten sind, können auch in Befehlen ausgedrückt werden. Hier wird vor allem der Zugang über Befehle dargestellt, da nur so die Dokumentation der Analysen als Beleg für das Vorgehen und für spätere Wiederholungen bzw. Überarbeitungen möglich ist. Trotzdem kann man natürlich auch über die Menüs durch Ausprobieren – insbesondere unter den Punkten ‚*Graphics*' und ‚*Statistics*' - einiges über die unterschiedlichen Stata Befehle herausfinden.

Neben diesen beiden Menüpunkten ist noch der Punkt ‚*Save Windowing Preferences*' unter dem Oberpunkt ‚*Prefs*' zu nennen. Hier können Änderungen der Fenstergröße und –position (verschiebbar mit dem Mauszeiger) gespeichert werden. Mit ‚*default windowing'* kann man die ursprüngliche Ansicht wiederherstellen. Unter dem Punkt ‚*help*' findet sich wie in anderen Programmen eine Hilfefunktion.

Neben den beim Öffnen von Stata sichtbaren Fenstern gibt es noch drei weitere Fenster, die über Befehle zu öffnen sind:

- das *Datenfenster*, in dem der verwendete Datensatz in Zeilen-Spalten-Struktur angezeigt wird (**browse** – zur Ansicht, **edit** – zum Verändern)
- der Viewer, eine Art zweites Ergebnisfenster, in dem unterschiedliche Inhalte angezeigt werden können, z.B. Hilfetexte (**view**)
- das Grafikfenster, in dem Grafiken angezeigt werden (öffnet sich automatisch wenn Grafikbefehle ausgeführt werden)
- der do-file-Editor, in dem do-files – Dateien in denen mehrere Befehle gesammelt eingegeben und gespeichert werden - erstellt werden können (**doedit**)

1.1 Das Eingabefenster

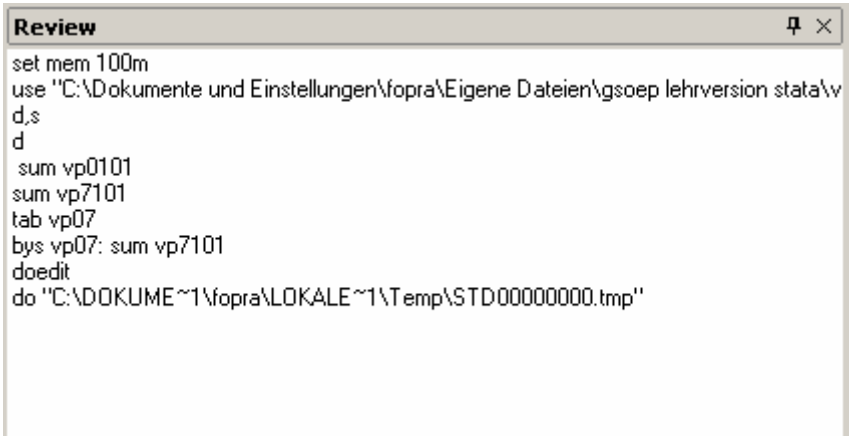
Im Eingabefenster eingegebene Befehle werden über die ‚Return'-Taste abgeschickt und sofort ausgeführt.

1.2 Das Ergebnisfenster

Hier werden sämtliche Ergebnisse angezeigt. Dabei erscheinen in der Standardeinstellung Befehle in weiß, Fehlermeldungen in rot, technische Rückmeldungen in blau (z.B. Anzahl geänderter Fälle), und Ergebnisse in grün/gelb. Im Ergebnisfenster ausgegebene Tabellen können mit dem Mauszeiger markiert werden (linke Maustaste beim Makieren gedrückt halten) und durch Drücken der rechten Maustaste in unterschiedliche Formate in die Zwischenablage kopiert werden. iüber die Tastenkombination Shift+Strg+C als Tabelle in die Zwischenablage kopiert werden. Sie können dann beispielsweise zur weiteren Bearbeitung in Excel eingefügt werden.

1.3 Das Review-Fenster

Hier erscheinen alle über das Eingabefenster ausgeführten Befehle. Befehle, die aus dem do-file-Editor abgeschickt werden, erscheinen nicht, sondern nur eine relativ kryptische Angabe zum verwendeten do-file. Durch Doppelklicken auf die entsprechende Zeile im Review-Fenster kann der Befehl bzw. das do-file wiederholt werden.



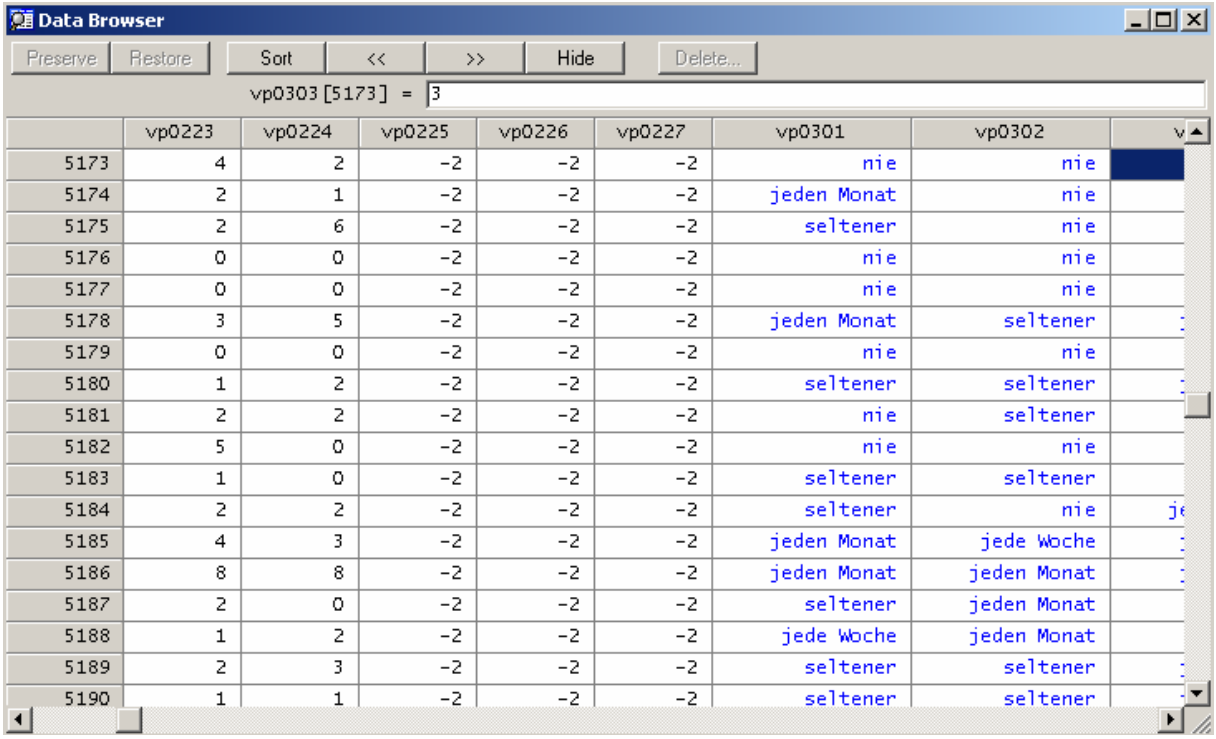
1.4 Das Variablenfenster

Im Variablenfenster werden sämtliche Variablen des verwendeten Datensatzes in der Ordnung, wie sie im Datensatz enthalten sind, angezeigt. Wenn für die Variablen

genauere Bezeichnungen vergeben wurden (‚labels') werden auch diese angezeigt. Durch Klicken auf die Variablennamen werden diese im Eingabefenster angezeigt.

1.5 Das Datenfenster (Stata Editor / Stata Browser)

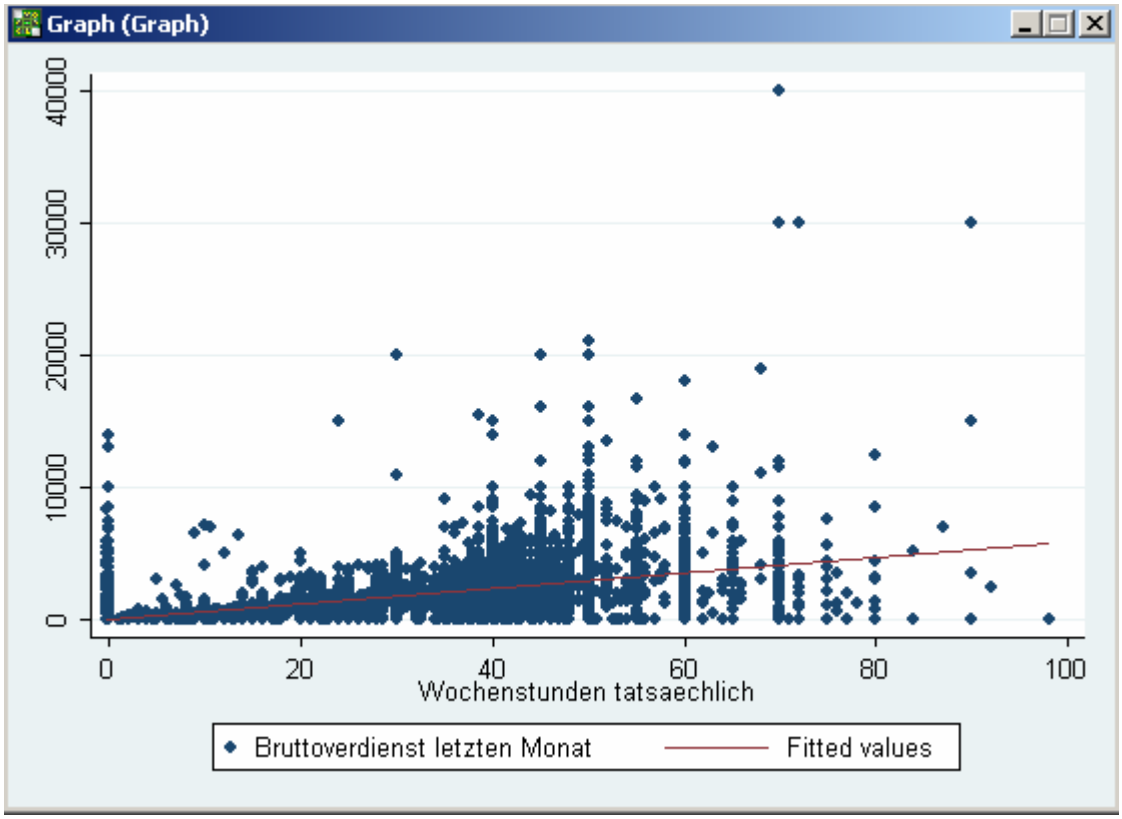
Im Datenfenster werden Datensätze in Zeilen-Spalten-Form angezeigt (zur Ansicht: Stata-Browser → **browse**, zum Verändern: Stata-Editor → **edit**).



	vp0223	vp0224	vp0225	vp0226	vp0227	vp0301	vp0302
5173	4	2	-2	-2	-2	nie	nie
5174	2	1	-2	-2	-2	jeden Monat	nie
5175	2	6	-2	-2	-2	seltener	nie
5176	0	0	-2	-2	-2	nie	nie
5177	0	0	-2	-2	-2	nie	nie
5178	3	5	-2	-2	-2	jeden Monat	seltener
5179	0	0	-2	-2	-2	nie	nie
5180	1	2	-2	-2	-2	seltener	seltener
5181	2	2	-2	-2	-2	nie	seltener
5182	5	0	-2	-2	-2	nie	nie
5183	1	0	-2	-2	-2	seltener	seltener
5184	2	2	-2	-2	-2	seltener	nie
5185	4	3	-2	-2	-2	jeden Monat	jede Woche
5186	8	8	-2	-2	-2	jeden Monat	jeden Monat
5187	2	0	-2	-2	-2	seltener	jeden Monat
5188	1	2	-2	-2	-2	jede Woche	jeden Monat
5189	2	3	-2	-2	-2	seltener	seltener
5190	1	1	-2	-2	-2	seltener	seltener

1.6 Das Grafikfenster

Im Grafikfenster (und nicht im Ergebnisfenster) werden sämtliche Grafiken angezeigt. Es wird immer nur das Ergebnis *eines* Grafikbefehls angezeigt. D.h., wenn Grafiken nicht gespeichert werden (siehe auch Abschnitt 3.4), werden sie beim Schließen des Grafikfensters oder bei Ausführung eines weiteren Grafikbefehls gelöscht.



1.7 Der do-file Editor

Im do-file-Editor können Befehlsdateien, so genannte do-files erstellt, verändert, betrachtet und abgeschickt werden. Der Editor wird mit dem Befehl **doedit** geöffnet. Do-files können jedoch auch mit jedem anderen Editor bearbeitet werden.

2. Grundlegende Dateitypen (‚files')

2.1 Log-Dateien/log-files (*.log / *.smcl)

In Log-Dateien werden alle Inhalte, die im Ergebnisfenster dargestellt werden, gespeichert. Es gibt zwei verschiedene Anzeigeformate für log-Dateien. Voreingestellt ist das so genannte smcl-Format (Stata-eigenes Format, Dateierweiterung von log-Dateien .smcl). Jedoch kann man sich log-Dateien auch in reinem Textformat ausgeben lassen, was für eine weitere Verarbeitung außerhalb von Stata sinnvoll ist. Hierzu muss man allerdings zunächst über den Befehl **set logtype text** die entsprechende Einstellung ändert. log-Dateien im Text-Format haben die Dateierweiterung .log. Unabhängig vom Format wird eine log-Datei mit dem Befehl **log using *filename*** geöffnet und mit dem Befehl **log close** geschlossen. Da Grafiken nicht im Ergebnisfenster angezeigt werden, werden sie auch nicht in log-files gespeichert.

2.2 Befehlsdateien/do-files (*.do)

Dateien (‚files'), in denen mehrere Befehle zusammen abgespeichert werden, die nacheinander ausgeführt werden, nennt man do-files (bei SPSS: Syntax-files). Alle do-files haben die Dateierweiterung .do.

2.3 Datensätze/data (*.dta)

Daten im Stata-Format werden in Datensätzen mit der Dateierweiterung .dta gespeichert.

2.4 weitere Dateitypen

Neben den drei beschriebenen Dateitypen gibt es noch weitere: ado-files (selbst oder von anderen Stata-Nutzern geschriebene Programme), hlp-files (Hilfeanweisungen, die man für eigene Programme auch selbst schreiben kann), dct-files (dictionary-files, die zum Einlesen von Daten in ASCII-Format, also unformatierte Zeichen, benötigt werden), gph.files (Grafikdateien). Näheres zu den einzelnen Dateitypen findet sich in Kohler/Kreuter 2006 (Abschnitt 3.1.8, Kapitel 10).

3. Arbeiten mit Stata

Wie bereits erwähnt können Befehle entweder über das Eingabefenster, in Form von do-files oder auch über das Menü ausgeführt werden. Aus Gründen der Dokumentation, Wiederholbarkeit und Arbeitsorganisation sollten Analysen allein über do-files durchgeführt werden.

Generell sind beim Arbeiten mit Stata folgende formale Regeln zu beachten:

- keine Umlaute oder ß (Teilweise kann Stata auch Umlaute und ß verarbeiten. Um trotzdem auftretende Probleme zu vermeiden, verzichtet man aber am besten auf beides.)
- Groß- und Kleinschreibung wird berücksichtigt
- möglichst keine Leerzeichen in Dateinamen und Verzeichnisnamen (Verzeichnisnamen mit Leerzeichen müssen in Anführungsstrichen stehen.)
- In der Standardeinstellung erfolgt die Ausgabe von Dezimalzahlen mit einem Punkt und nicht mit einem Komma (vgl. Abschnitt 6)

3.1 Ablage von Dateien

Sowohl Daten (dta-files), Befehle (do-files), Ergebnisse und Grafiken (log-files und gph-files) werden in Dateien gespeichert. Da unterschiedliche Datenaufbereitungs- und Analyseschritte in unterschiedlichen Dateien gespeichert werden sollten, kommt man bereits bei kleineren Analysen relativ schnell auf eine größere Menge von Dateien. Um nicht den Überblick zu verlieren, sollte man sämtlichen Dateien zu einem Projekt (Hausarbeit, Abschlussarbeit, Artikel, Vortrag usw.) in ein separates *Dateiverzeichnis* speichern. Neue Verzeichnisse können direkt aus Stata über das Eingabefenster angelegt werden (Befehl **mkdir**). Auch kann man über das Eingabefenster durch die komplette Verzeichnisstruktur auf einem Datenträger (Festplatte, Diskette, USB-Stick ...) navigieren (Befehl **cd**). Wer mit DOS oder UNIX vertraut ist, dem werden die Befehle bekannt vorkommen. In einer Windowsumgebung erscheint dieses Vorgehen zunächst etwas gewöhnungsbedürftig. Dieses Vorgehen macht jedoch trotzdem Sinn: So kann jeweils das aktuelle Arbeitsverzeichnis eingestellt werden und man spart sich bei der weiteren Arbeit das Eingeben langer Pfadnamen. Insbesondere für do-files, in denen auf Datensätze oder andere do-files zugegriffen wird ist dies hilfreich. Wird zu Beginn des do-files das Arbeitsverzeichnis definiert ist das do-file ohne weitere Änderungen lauffähig.

3.2 Interaktives Arbeiten: Eingabefenster

Zum Ausprobieren von Befehlen oder anderen Schritten, die nicht dauerhaft gespeichert werden müssen, können Befehle in das Eingabefenster eingegeben werden. Mit der ‚Return'-Taste werden Befehle abgeschickt und sofort ausgeführt. Mit dem Befehl **#review** kann man sich bis zu 100 der letzten Befehle im Ergebnisfenster anzeigen lassen und ggf. dokumentieren. Mir den Tasten ‚Bild-oben' und ‚Bild-unten' kann man durch die bislang eingegebenen Befehle navigieren.

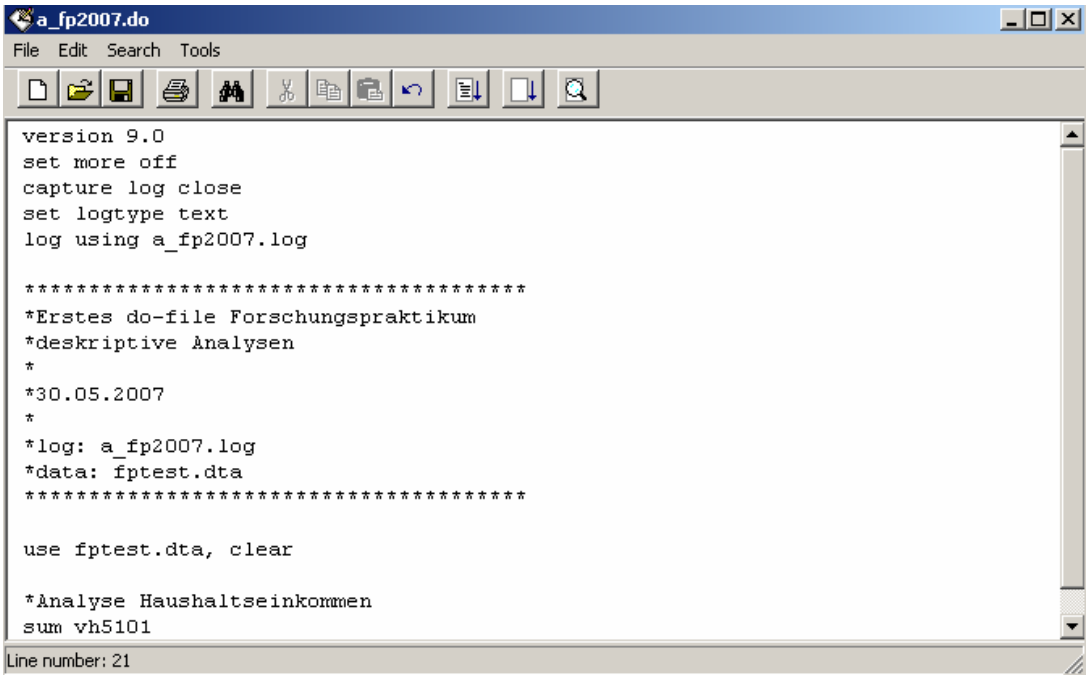
Weiter können über den Befehl **do** ***filename*** bzw. **run** ***filename***, do-files ausgeführt werden, ohne diese zuvor im do-file-Editor zu öffnen.

3.3 Arbeiten mit do-files

In do-files wird eine Abfolge von Befehlen gespeichert. Jedes do-file sollte zu Beginn die folgenden Elemente enthalten:

- die Angabe für welche Stata-Version das do-file geschrieben wurde (z.B. **version 9.0**)
- den Befehl zum Schließen evtl. geöffneter log-Dateien: **capture log close**
- die Einstellung **set more off**, um zu Verhindern, dass die Ergebnisausgabe stoppt, wenn der Bildschirm „vollgeschrieben" ist (vgl. auch Abschnitt 6)

- die Einstellung **`set logtype text`** bzw. **`smcl`**, um das Ausgabeformat für log-Dateien einzustellen (vgl. Abschnitt 2.1)
- den Befehl zum Öffnen einer log-Datei (vgl. Abschnitt 2.1)
- eine Überschrift bzw. ein Infoblock, um auch nach längerer Zeit noch zu wissen, zu welchem Projekt und welchem Analyseschritt das do-file gehört (Zeile, an deren Anfang ein **`*`** steht werden nicht als Befehl erkannt, sondern als reiner Kommentar. Mit **`/*`** (Anfang) und **`*/`** (Ende) kann man einen ganze Bereich als Kommentar kennzeichnen.)



Am Ende des do-files sollte der Befehl **`log close`** stehen, um die log-Datei wieder zu schließen.

3.4 Arbeiten mit Stata-Grafiken

Grafiken werden standardmäßig im Grafikfenster angezeigt und nicht in Log-Dateien gespeichert. Grafiken können auf zwei unterschiedliche Arten gespeichert werden. Entweder können Grafiken in die Zwischenablage kopiert werden (Tastenkombination Strg+C) und dann beispielsweise in einen Text eingefügt werden (Tastenkombination Strg+V). Oder Grafiken können über einen Befehl gespeichert werden und dann wieder in Stata geöffnet werden.

`graph save` ***`filename`*** (Speichern)
`graph use` ***`filename`*** (Öffnen)

3.5 Hilfe

Die Hilfe enthält sehr umfangreiche Informationen zu sämtlichen Stata-Befehlen. Mit dem Befehl **`help`** wird der Hilfetext direkt im Ergebnisfenster angezeigt. Natürlich kann man auch die Hilfefunktion im Menü benutzen.

`help` ***`befehlsname`***
→ **`help`** ***`graph`***

4. Struktur von Stata-Kommandos

Die Begriffe Stata-Befehl und Stata-Kommando sind bislang nicht unterschieden worden. Im Folgenden soll mit ‚Befehlen' der eigentliche Befehl bezeichnet werden (z.B. **doedit**) während als Stata-Kommando ein Ausdruck bezeichnet werden soll, der noch weitere Elemente enthält, z.B. den Dateinamen des entsprechenden do-Files (z.B. **doedit *filename***). Neben Befehlen und Dateinamen gibt es allerdings eine Reihe weiterer Elemente in Stata-Kommandos:

- Befehl (z.B. **summarize)**
  Ein Befehl muss in jedem Stata-Kommando enthalten sein. Mit dem Befehl wird grundsätzlich ausgedrückt welche Form der Analyse, der Datenverwaltung/ –veränderung oder welche anderen Schritte durchgeführt werden sollen. Der Befehl **summarize** gibt beispielsweise an, dass ein Mittelwert (und einiges mehr) berechnet werden soll.
- Variablenliste (***varlist***)
  Die meisten Befehle beziehen sich auf bestimmte Variablen. Dies können eine oder auch mehrere Variablen sein. Dies wird durch die Abkürzung ***varlist*** ausgedrückt, die im Folgenden immer kursiv geschrieben wird, um sie von den Befehlen zu unterschieden.
- Spezielle Variablenlisten (z.B. ***depvar***)
  In manchen Befehlen kann man zwischen verschiedenen Typen von Variablen unterschieden, z.B. unabhängige und abhängige Variablen, alte und neue Variablen.
- In-Bedingung (**in #1/#2**)
  Über die in-Bedingung wird angeben auf welche Fälle der Befehl angewendet werden soll. Die in-Bedingung steht immer hinter dem Befehl und der Variablenliste. So sollen bei dem Befehl **list *hhnr vh5101* in 1/50** die Variablen hhnr und vh5101 für die ersten 50 Fälle in einer Liste angezeigt werden. Die Nummerierung der Fälle bezieht sich dabei auf eine interne Zählung, die von der Sortierung des Datensatzes abhängt (vgl. Abschnitt 5.8).
- If-Bedingung
  Die if-Bedingung steht i.d.R. am Ende des Befehls, aber vor den Optionen. Der voranstehende Befehl wird nur für die Fälle ausgeführt, für die nachstehende Bedingung wahr ist. Z.B. wird das Kommando **list *hhnr vh5101* if hhnr<100**
  nur für die Fälle ausgeführt, bei denen die Variable hhnr kleiner als 100 ist.
- Gewichtungsanweisung
  Gewichtungsanweisungen werden in eckigen Klammern am Ende des Befehls, aber vor den Optionen angeführt. Stata unterscheidet vier Typen von Gewichten. In der Gewichtungsanweisung muss angegeben werden, welche Art von Gewicht verwendet wird (Beispiel: **[aw=weight]**).
- Optionen
  Zu den meisten Befehlen gibt es Reihe von Optionen. Diese werden zumeist am Ende eines Befehls nach einem Komma genannt. In der im Folgenden verwendeten Notation werden Befehlsoptionen in eckigen Klammern nach dem Befehl aufgeführt, z.B. **[, clear].**
- Dateinamen (***filename***)
  Z.B. beim Öffnen und Speichern von Datensätzen werden Dateinamen angegeben.
- Befehls-Präfixe (z.B. **by**)
  Über das by-Präfix wird ein Befehl für die unterschiedlichen Ausprägungen einer bestimmten Variable getrennt durchgeführt. Mit dem Befehl **by *sex*: summarize *vh5101*** werden getrennt Mittelwerte für Männer und Frauen für die Variable vh5101 ausgerechnet. Dafür muss der Datensatz jedoch zuvor nach der entsprechenden Variablen sortiert werden (vgl. Abschnitt 5.8).

Weitere Elemente von Stata-Kommandos sind Anweisungen für Schleifen (foreach/forvalues) und die Nummernliste (vgl. Kohler/Kreuter 2006: Abschnitte 3.1.7, 3.2.2 und 3.2.3).

5. Stata-Befehle

5.1 Öffnen von bestehenden Datensätzen

Stata lädt die jeweils verwendeten Datensätze komplett in den Hauptspeicher des Computers. In der Voreinstellung ist der Speicherbereich relativ klein (1 MB). Die Größe des Speicherbereichs, den Stata verwenden kann, wird über den Befehl **set memory** definiert. Ist der Datensatz größer als der Hauptspeicher des Computers muss mit Teildatensätzen gearbeitet werden. Bei halbwegs aktuellen Computern sollte man jedoch bei den meisten Datensätzen nicht an diese Grenzen stoßen.

set memory #m
→ **set memory 20m**

Bestehende Datensätze können mit dem Befehl **use** in den Speicher geladen und damit geöffnet werden.

use *filename* [, clear]
→ **use *fppfad\fptest.dta***

Mit der Option **clear** werden im Speicher befindliche Daten gelöscht. Sind noch Daten im Speicher erfolgt bei Öffnen eines neuen Datensatzes ohne die clear-Option eine Fehlermeldung. **clear** kann auch als Befehl allein verwendet werden. Die aktuell verwendeten Daten werden dann aus dem Speicher gelöscht.

5.2 Speichern von Datensätzen

Datensätze können mit dem Befehl **save** gespeichert werden. Den Ausgangsdatensatz sollte man niemals verändern, sondern veränderte Datensätze unter neuem Namen abspeichern. Die Befehle, mit denen ein Datensatz verändert wurde, sollten in einem do-file dokumentiert werden (siehe Abschnitt 3.3), sodass man ausgehend von dem Ausgangsdatensatz, die Veränderungen nachvollziehen kann.

save *filename* [, replace]
→ **save *fppfad\fptest_neu.dta***

Mit der Option **replace** wird ein bereits bestehender Datensatz gleichen Namens überschrieben.

5.3 Import von Daten in Stata

Daten liegen in unterschiedlichen Formen vor. Eher selten erhält man Daten bereits als Stata-Datensatz, den man mit **use** direkt laden kann. Häufig erhält man Daten als unformatierten Text (ASCII) oder im Format eines anderen Statistikprogrammpakets (z.B. SPSS). Ist letzteres der Fall gibt es zwei Möglichkeiten: Man kann die Daten über ein zusätzliches Programm (z.B. DBMS Copy, Stat/Transfer) in Stata-Format umwandeln. Oder: Man kann die Daten in dem anderen Programm öffnen und als unformatierten Text abspeichern, den man dann in Stata importieren kann.

Zum Importieren von unformatierten Daten gibt es in Stata mehrere Möglichkeiten (Befehle: infix, insheet, infile, vgl. Kohler/Kreuter 2006: Kap. 10).

5.4 Beschriftung von Datensätzen und Variablen

Mit dem Befehl **describe** kann man sich eine kurze Übersicht über den Datensatz geben lassen.

describe [, short]
→ **describe**

```
Contains data from C:\stata\vp.dta
  obs:        10,451                          2006 GSOEP@school: vp.dta
 vars:           527                          15 Aug 2006 14:54
 size:    10,963,099 (89.5% of memory free)
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
hhnr            long   %12.0g                 Ursprungshaushaltsnummer
persnr          long   %12.0g                 Unveraenderliche Personennummer
welle           int    %8.0g                  Befragungswelle(1984 =1)
```

Angezeigt wird der Name des Datensatzes, die Anzahl der Beobachtungen, die Anzahl der Variablen, die Größe (in KB) und die Ausschöpfung des Speicherplatzes, eine Kommentierung (sofern vorhanden) und das Datum der Erstellung des Datensatzes. Dann folgt eine Liste mit allen Variablen unter Angabe des Namens und weiterer Angaben und einer Kommentierung des Variablennamens (falls vergeben). Am Ende wird angezeigt, ob der Datensatz nach einer bestimmten Variable sortiert ist.

Die meisten Datensätze erhalten eine Vielzahl von Variablen, die häufig eine Reihe von Ausprägungen haben. Die gespeicherten Variablennamen sind zumeist sehr kurz und enthalten – wenn überhaupt – keine genaue Beschreibung der Variablen. Unterschiedliche Ausprägungen werden in unterschiedlichen Werten ausgedrückt (z.B. 1=männlich, 2=weiblich). Um den Datensatz für sich selbst und evtl. für andere übersichtlicher zu gestalten, sollte man Datensätze, Variablen und Variablenausprägungen über den Befehl **label** beschriften.

Beschriftung des Datensatzes:
label data *"label"*
→ **label data "Forschungspraktikum 2007 geaendert"**

Beschriftung einer Variablen:
label variable *varname "label"*
→ **label variable *vp0101n* "Zufriedenheit Gesundheit 3 Auspr."**

Definition von Bezeichnungen für Variablenausprägungen (value labels):
label define *labelname # "label"* [# "label" ...]
→ **label define *vp0101n* 1 "(eher) unzufrieden" 2 "weder unzufrieden noch zufrieden" 3 "(eher) zufrieden"**

Zuordnung von value labels zu Variablen:
label values *varname labelname*
→ **label values *vp0101n vp0101n***

Da value labels zunächst unabhängig von einzelnen Variablen definiert werden, kann ein einmal definiertes Label bei gleichen Ausprägungen (z.B. 1=ja, 0=nein) für mehrere Variablen vergeben werden.

5.5 Datenanalyse: Erste Schritte

Mit dem Befehl **list** kann man die Ausprägungen bestimmter Variablen für einzelne Fälle betrachten.

list *varlist* [in #/#, clean]
→ **list *vhhnr persnr sex gebjahr* in 1/50, clean**

Mit dem Befehl **tabulate** werden Häufigkeitsauszählungen einzelner Variablen oder Kreuztabellen von zwei Variablen erstellt.

Häufigkeitsauszählung:
tabulate *varname* [,missing]
→ **tabulate *sex*, missing**
→ **tabulate *vp0101***
→ **tabulate *vp10***

(Option: Anzeigen von fehlenden Werten)

Kreuztabelle:
tabulate *varname1 varname2* [, row column cell expected nofreq all]
→ **tabulate *vp10 sex*, col**

(Optionen: Zeilen-/Spalten-/Zellenprozente, Erwartungswerte, nur Prozentwerte, sämtliche verfügbaren Statistiken)

Mit dem Befehl **summarize** kann man Variablen zusammenfassen, z.B. wird das arithmetische Mittel berechnet. Es ist zu berücksichtigen, dass sinnvoll interpretierbare Werte nur für Variablen mit einem entsprechenden Skalenniveau berechnet werden können; das arithmetische Mittel beispielsweise nur für mindestens intervallskalierte Variablen.

summarize *varlist* [,detail]
→ **summarize *vp0101*, detail**

(Option: Anzeige von Perzentilen inkl. Median und weiterer Details)

5.6 Erstellen von Grafiken: Erste Schritte

Mit Stata lassen sich sehr präzise Grafiken erstellen, die Befehlsstruktur ist z.T. aber nicht leicht zugänglich. Viele Möglichkeiten Grafiken zu erstellen, finden sich unter dem Befehl **graph**. Allerdings muss zusätzlich noch der Typ der Grafik definiert werden, z.B. ob ein Balkendiagram, ein Streudiagram (Scatterplot) oder eine andere Form von Grafik gebildet werden soll. Im folgenden Beispiel wird ein Streudiagram mit zwei Variablen gebildet.

graph twoway (scatter *varname1 varname2*)
→ **graph twoway (scatter *vp7102 alter*)**

Histogramme warden nicht mit dem graph-Befehl generiert, sondern mit dem Befehl **histogram**. Hier ein Beispiel:

histogram *varname* [, frequency percent]
→ **histogram gebjahr**

(Optionen: Anzeige von Häufigkeiten bzw. Anteilswerten)

Insbesondere der graph-Befehl aber auch andere Grafik-Befehle enthalten eine Vielzahl von Optionen, die hier nicht dargestellt werden können.

5.7 Erstellen und Veränderung von Variablen

Veränderungen an Variablen sollten nie an einer bereits im Datensatz vorhandenen Variablen, sondern immer an neu gebildeten Variablen erfolgen. Nur so sind Veränderungen später wiederholbar und nachvollziehbar. Vor allem werden aber auch die in der Ursprungsvariable vorhandenen Informationen nicht vernichtet. Z.B. kann man so bei Zusammenfassungen von Ausprägungen einer Variablen später auch andere Zusammenfassungen definieren bzw. mit der ursprünglichen Variable arbeiten.

Neue Variablen werden mit dem Befehl **generate** gebildet.

generate ***varname=exp***
→ **generate** ***teilzeit=.***
→ **generate** ***vp0101n=vp0101***
→ **generate** ***einktaus=vh5101/1000***

Veränderungen an Variablen können entweder mit dem Befehl **replace** oder dem Befehl **recode** vorgenommen werden. **replace** ersetzt bisherige Werte einer Variablen mit neuen Werten. **recode** verändert bisherige Werte. Da der Befehl deutlich weniger Rechenzeit in Anspruch nimmt, wird er von den meisten Nutzern häufiger angewendet.

replace ***varname=exp*** **[if exp]**
→ **replace** ***teilzeit=0*** **if** ***vp10==1***
→ **replace** ***teilzeit=1*** **if** ***vp10==2***

recode ***varlist*** **(rule) [(rule)]**
→ **recode v*p0101n* (0/4=1) (5=2) (6/10=3)**

Einzelne oder mehrere Variablen können mit dem Befehl **drop** gelöscht werden:

drop ***varlist***
→ **drop** ***vp0101n***

Mit dem Befehl **keep** werden alle nicht genannten Variablen gelöscht:

keep ***varlist***
→ **keep** ***hhnr persnr sex gebjahr***

5.8 Sonstige Befehle

Zur Verwendung der by-Option (vgl. Abschnitt 4) müssen die Daten nach der dabei verwendeten Variablen sortiert sein (Es gibt allerdings auch den Befehl **bysort**, der dies in einem Schritt erledigt). Auch zum Zusammenführen von zwei Datensätzen ist es notwendig, die Daten nach bestimmten Variablen zu sortieren. Das Zusammenführen von Datensätzen wird in diesem Skript jedoch nicht erläutert. Auch für die Betrachtung von einzelnen Fällen über **list** kann es hilfreich sein, die Daten in einer bestimmten Weise sortiert zu haben.

sort ***varlist***
→ **sort v*hhnr persnr***

Manchmal macht es Sinn, z.B. bei der Berechnung von internen Ergebnissen (vgl. Abschnitt 6), die Ausgabe von Ergebnissen über den Befehl **quietly** zu unterbinden. **quietly** kann prinzipiell mit sämtlichen Befehlen kombiniert werden.

quietly ***command***
→ **quietly** ***sum vh5101***

Der Befehl **egen** ist eine Erweiterung des Befehls **generate**. Neue Variablen können unter Verwendung einer Reihe von Funktionen gebildet werden, z.B. können Werte anderer Variablen aufsummiert werden (Funktion: **rsum(*varlist*)**). Sehr praktisch ist auch die Funktion **rmiss(*varlist*)**, mit der gezählt wird, in wie vielen Variablen missing values auftreten.

egen ***varname*****=function(*****varlist*****)**
→ **egen** ***varname*****=rmiss(*****vp7101 vp7102 vh49*****)**

6. Einstellungen, Variablentypen, Operatoren und interne Ergebnisse

Einstellungen:
Mit dem Befehl **set** (vgl. **set memory**, Abschnitt 5.1) können bestimmte Einstellungen verändert werden:

set dp comma	Anzeige von Dezimalzahlen mit Komma
set dp period	Anzeige von Dezimalzahlen mit Punkt (Standardeinstellung)
set memory 50m	Größe des verfügbaren Speichers (hier 50 MB)
set logtype text	Ausgabe von Logdateien in Textformat
set logtype smcl	Ausgabe von Logdateien in smcl-format
set more on	Stopp der Ausgabe, wenn Bildschirmseite voll ist
set more off	kein Stopp der Ausgabe

Variablentypen:
Generell werden zwei Variablentypen unterschieden: Variablen, deren Inhalt als Text behandelt wird (Strings, alphanumerische Variablen) und Variablen, deren Inhalt als Zahl behandelt wird (Reals, numerische Variablen). In fast allen Datensätzen sind die meisten Variablen numerisch. Insbesondere bei numerischen Variablen sind weitere Typen zu unterschieden (siehe Tabelle). Stringvariablen unterscheiden sich allein in der Länge. Die von uns verwendete Version von Stata kann Strings mit einer Länge von maximal 80 Zeichen verarbeiten.

Numerische Variablen (Reals)				
	bytes	Minimalwert	Maximalwert	Genauigkeit
Byte	1	-127	126	Ganze Zahl
Int	2	-32.768	32.766	Ganze Zahl
Long	4	-2.147.483.648	2.147.483.656	Ganze Zahl
Float	4	+/-10^37	+/-10^37	6*10^-8
Double	8	+/-10^99	+/-10^99	2*10^-16
Alphanumerische Variablen (Strings)				
	bytes	max. Länge		
str1	1	1		
str2	1	2		
str80	80	80		

Werden über **generate** neue Variablen gebildet, kann man direkt den Variablentyp als Option angeben. Standardmäßig werden float Variablen gebildet. In den meisten Fällen ist dies ausreichend. Ist das double Format notwendig sollte dies bei Bildung der Variablen berücksichtigt werden.

→ **generate double var1=.**

Notwendig ist die Angabe des Formats auch, wenn neue Textvariablen (Stringvariablen) gebildet werden sollen.

Sind andere Formate ausreichend (byte, int, long) ist es nicht notwendig dies zu spezifizieren. Zwar wird dabei der Datensatz unnötig groß, jedoch kann man dies mit dem Befehl **compress** beheben. Dabei wird jeweils das ‚sparsamste' Format für die vorhandenen Variablen gewählt.

Operatoren:
Zur Berechnung neuer Variablen oder in Ausdrücken wie z.B. if-Bedingungen können die folgenden Operatoren verwendet werden:

Arithmetisch	
+	Addition
-	Subtraktion
*	Multiplikation
/	Division
^	'hoch'
Logisch	
~	nicht
|	oder
&	und
Relational	
>	größer als
<	kleiner als
> =	größer oder gleich
< =	kleiner oder gleich
= =	gleich
~ =	ungleich

Interne Ergebnisse:
Zu fast jedem Befehl werden (zumindest einige) der angezeigten Ergebnisse solange intern gespeichert bis der nächste Befehl ausgeführt wird. Diese Ergebnisse können für weitere Berechnungen verwendet werden. So kann beispielsweise ein gespeicherter Mittelwert zur Berechnung einer Armutsquote verwendet werden. Hier einige der Ergebnisse, die beim Befehl **summarize** gespeichert werden:

r(N)	Fallzahl
r(mean)	arithmetisches Mittel
r(p50)	Median
r(min)	Minimum
r(max)	Maximum

Mit dem Befehl **return list** können nach Ausführen eines Befehls sämtliche gespeicherten internen Ergebnisse angezeigt werden.

Beispiel zum Rechnen mit internen Ergebnissen (Armutsgrenze, 50% des arithmetischen Mittels):

```
summarize einkommen
generate arm=0.5*r(mean)
```

7. Befehle im Überblick

Allgemein	
browse	Öffnen des Datenfensters zur Ansicht
edit	Öffnen des Datenfensters zum Verändern
view	Öffnen des Viewers
doedit *filename*	Öffnen eines do-files im do-file-Editor
log using *filename*	Öffnen einer log-Datei
log close	Schließen der aktuellen log-Datei
mkdir *verzname*	Anlegen eines Verzeichnisses
cd *verzname*	Wechsel des Verzeichnisses
cd ..	Wechsel in das vorherige Verzeichnis
**cd **	Wechsel in c:\ (oder d:\, e:\ …)
pwd	Anzeige des aktuellen Verzeichnisses
#review	Anzeige der zuletzt eingegebenen Befehle
do *filename*	Ausführen eines do-files
run *filename*	Ausführen eines do-files
help *befehlsname*	Anzeigen der Hilfe zu einem Befehl
Öffnen, speichern und Import von Datensätzen	
use *filename* [, clear]	Öffnen eines Datensatzes (Option: Löschen bisheriger Daten im Speicher)
save *filename* [, replace]	Speichern eines Datensatzes (Option: Überschreiben eines Datensatzes gleichen Namens)
infix	Import von Daten im festen Format
insheet	Import von Daten im Spreadsheet-Format
infile	Import von Daten
Betrachtung und Beschriftung von Datensätzen	
describe [,short]	Anzeigen von Informationen zum Datensatz und den enthaltenen Variablen (Option: nur Info zum Datensatz)
label data *"label"*	Beschriftung des Datensatzes
label variable *varname "label"*	Beschriftung einer Variablen
label define *labelname # "label" [# "label" ...]*	Definition von Bezeichnungen für Variablenausprägungen (value labels)
label values *varname labelname*	Zuordnung von value labels zu Variablen
Datenanalyse	
list *varlist* [in #/#, clean]	fallweise Betrachtung bestimmter Variablen (Optionen: Auswahl von Fällen, Anzeige ohne Rahmen)
tabulate *varname* [,missing]	Häufigkeitsauszählung (Option: Anzeige von fehlenden Werten)
tabulate *varname1 varname2* [, row column cell expected nofreq all]	Kreuztabelle (Optionen: Zeilenprozente, Spaltenprozente, Zellenprozente, Erwartete Werte, keine Häufigkeiten, Anzeige aller statistischen Maßzahlen)
summarize *varlist* [,detail]	Mittelwert und andere Angaben (Option: weitere Angaben, z.B. Median)
Erstellen von Grafiken	
graph twoway (scatter *varname1 varname2*)	Erstellen eines Streudiagrams
histogram *varname* [, frequency	Erstellen eines Histograms

percent]	
Erstellen und Verändern von Variablen	
generate *varname=exp*	Erstellen einer neuen Variable
egen *varname*=rmiss(*varlist*)	Erstellt neue Variable, die missing values in ***varlist*** zählt
replace *varname=exp* [if exp]	Ersetzen eines Wertes einer Variablen [Option: Bedingung)
recode *varlist* (rule) [(rule)]	Rekodierung einer Variablen
drop *varlist*	Löschen einer Variablen
keep *varlist*	"Behalten" von Variablen (alle übrigen werden gelöscht)
sort *varlist*	Sortierung des Datensatzes
Einstellungen	
set dp comma	Anzeige von Dezimalzahlen mit Komma
set dp period	Anzeige von Dezimalzahlen mit Punkt (Standardeinstellung)
set memory 50m	Größe des verfügbaren Speichers (hier 50 MB)
set logtype text	Ausgabe von Logdateien in Textformat
set logtype smcl	Ausgabe von Logdateien in smcl-format
set more on	Stopp der Ausgabe, wenn Bildschirmseite voll ist
set more off	kein Stopp der Ausgabe
Sonstiges	
quietly *command*	Unterdrückung der Ausgabe von Ergebnissen bei Durchführung eines Befehls
return list	Anzeigen von gespeicherten internen Ergebnissen nach Durchführung eines Befehls

8. Weitere Informationen

Literatur:

Acock, Alan C. (2006) A Gentle Introduction to Stata. College Station, Texas: Stata Press.

Hamilton, Lawrence C. (2006): Statistics with Stata (Updated for Version 9). Brooks/Cole.

Kohler, Ulrich/Kreuter, Frauke (2006): Datenanalyse mit Stata. Allgemeine Konzepte der Datenanalyse und ihre praktische Anwendung. München/Wien: Oldenbourg

Stata Corp. (2005): Getting Started with Stata for Windows. Release 9. College Station, Stata Corp.

Stata Corp. (2005): Stata Base Reference Manual. Release 9 (Vol. 1-3). College Station, Stata Corp.

Stata Corp. (2005): Stata Graphics Reference Manual. Release 9. College Station, Stata Corp.

Stata Corp. (2005): Stata User's Guide. Release 9. College Station, Stata Corp.

Websites:

Datensätze und weitere Informationen zu Kohler/Kreuter 2006
http://www.stata-press.com/data/kkd.html

Stata-Homepage
http://www.stata.com
interessant insbesondere folgende Seiten:
http://www.stata.com/support/faqs/ (frequently asked questions)
http://www.stata.com/links/resources1.html (resources for learning Stata)

Stata learning modules der UCLA
http://www.ats.ucla.edu/stat/stata/