
Einführung in die Datenanalyse mit Stata

© Prof. Dr. Stephan Huber
Version: 1.92 (9. Dezember 2021)

Vorbemerkungen

Mit dem PC-Programm Stata kann man Daten manipulieren, visualisieren und analysieren. Es funktioniert auf verschiedenen Betriebssystemen (Windows, Mac, Linux) und kann vieles besser als andere Programme (R, Eviews, SPSS, SAS, Excel, ...). Obwohl kostenpflichtig, ist Stata nicht nur unter Sozialwissenschaftlern weit verbreitet. Von den Ursprüngen als Kommandozeilen basiertes Programm hat sich Stata mittlerweile weiterentwickelt zu einem modernen Programm mit ansprechender graphischer Benutzeroberfläche. Diese erleichtert den Einstieg enorm. Nichtsdestotrotz bleibt das Erlernen der Befehlssyntax für eine fortgeschrittene Datenanalyse unerlässlich.

Folgendes kurzes Skript soll den Einstieg in Stata erleichtern, ein effizientes vorankommen ermöglichen und zum Entdecken der Möglichkeiten von Stata ermuntern. Es stellt keinen Anspruch auf Vollständigkeit und will nicht mit existierenden Lehrbüchern wie Kohler and Kreuter (2016) oder Acock (2018) konkurrieren. Statt dessen soll es den Stata-Kurs des Autors flankieren. Inhaltlich werden die Möglichkeiten von Stata überblickt, in die Bedienung des Programms eingeführt, die gängigsten Befehle genannt und zum *Kommandozeilen basierten* Arbeiten angeleitet. Die Kürze dieses Skriptes bringt es mit sich, dass ein Blick in die Stata Hilfe (`.help`) für ein ausreichendes Verständnis der Befehle unerlässlich ist. Darüber hinaus wird eine Sammlung von Übungsaufgaben geboten. Die Datensätze und Lösungen zu den Aufgaben, sowie alte Klausuraufgaben sind im Internet hier erhältlich: <https://t1p.de/stata-files> (Passwort:happystata). Hier der QR-code:



Angaben zu meiner Person findet man auf meiner Homepage: <https://t1p.de/stephanhuber>
Feedback ist willkommen.

Contact

Prof. Dr. Stephan Huber
Hochschule Fresenius für Wirtschaft & Medien GmbH
Im MediaPark 4c
50670 Cologne

 +49 221 973199-523

 drstephanhuber@yahoo.com

 www.t1p.de/stephanhuber

Inhaltsverzeichnis

1	Bedienung	4
1.1	Die Benutzeroberfläche	4
1.2	Tippen vs. klicken	4
1.3	Die Befehlssyntax	4
1.4	Beispiel	4
1.5	Log-files	6
1.6	Do-Files	6
2	Grundlegendes	7
2.1	Aktuelle Version	7
2.2	Hilfe	7
2.3	Literaturempfehlungen	7
2.4	Das Arbeitsverzeichnis	7
2.5	Updates	8
2.6	Von Benutzern geschriebene Befehle (ado-files)	8
2.7	Selbst programmieren	9
2.8	Das Betriebssystem mit Stata steuern	10
2.9	Stata Besonderheiten	10
3	Daten Management	11
3.1	Datenstruktur	11
3.2	Allgemeine Informationen zur Befehlsstruktur	11
3.2.1	Variablen	11
3.2.2	Beobachtungen	12
3.2.3	Labels	12
3.2.4	Operatoren	12
3.2.5	Die in-Bedingung	12
3.2.6	Die if-Bedingung	12
3.2.7	Makros	13
3.3	Daten einlesen	13
3.4	Daten speichern	13
3.5	Daten kombinieren	13
3.6	Daten erstellen und manipulieren	14
3.6.1	Variablen erzeugen und manipulieren mit...	14
3.6.2	Funktionen	16
4	Schleifen und sonstige Helfer	16
4.1	Für jede Beobachtung: .by(sort) und .levelsof	17
4.2	Regressieren mit .xi	17
4.3	Wenn-Dann Abfragen	17
4.4	Abzählen mit _n und _N	17
5	Datenanalyse	18
5.1	Allgemein	18
5.2	Tabellen	18
5.2.1	.tabstat und .table	18
5.2.2	Regressionen tabellieren mit .estimates table	18
5.3	Grafiken	18
5.3.1	Grafiken speichern, bearbeiten und exportieren	18
5.3.2	Das Aussehen von Grafiken	19
5.3.3	.tway	19
5.3.4	Spezielle Grafik Typen	19

6	Die Erstellung von publikationswürdigen Tabellen	22
6.1	.outreg2	22
6.2	.estout	22
6.3	.xml_tab	23
6.4	.tabout	24
6.5	outtex	24
6.6	.regsave	25
6.7	.texsave	26
6.8	.putexcel	26
7	Sonstige nützliche Befehle	26
8	Übungsaufgaben	27
8.1	Erste Schritte	27
8.2	Daten einlesen (daten_einlesen.do)	29
8.3	Sicherheitsgurte und Verkehrstote (gurte.do)	30
8.4	Bundestagswahl 2017: Die Partei und die AFD (wahl.do)	31
8.5	Bundestagswahl 2017: (wahl_sozial.do)	32
8.6	Wohnfläche (wohnen.do)	33
8.7	Poser Autos (posercar.do)	34
8.8	Income and Exams (incomeandexams.do)	35
8.9	Beauty and teaching evaluation score (beauty.do)	36
8.10	Lehrer-Schüler Verhältnis (school.do)	37
8.11	Makros und Co (makros.do)	38
8.12	Fingeruebung (finger.do)	40
8.13	Die Biersteuer und Verkehrstote (beertax.do)	41
8.14	Guns (guns.do)	42
8.15	Schleifen (schleifen.do)	44
8.16	Tabellen und Grafiken (tabellen_grafiken.do)	45
8.17	Panel Bild Alternativen (panel_pic.do)	47
8.18	Konvergenz (konvergenz.do)	48
8.19	Zipf's Law (zipf.do)	49
8.20	Das Einkommen von verheirateten Frauen (marry.do)	50
8.21	Migration in Deutschland (migration.do)	52
8.22	Migration nach Deutschland	54
	Literaturverzeichnis	56

1 Bedienung

1.1 Die Benutzeroberfläche

Menu Hier verbergen sich eine Reihe von Befehlen die über eine Benutzeroberfläche zu steuern sind.

Results Fenster Resultate von Berechnungen, aufgerufene Befehle und Fehlermeldungen.

Command Fenster Direkte Eingabe von Befehlen. Durch die Bild↑ und Bild↓ Tasten erscheinen die vorangegangenen Befehle wieder in der Kommandoingabe.

Review Fenster Dokumentation der zuletzt eingegebenen Befehle. Durch Anklicken eines bestimmten Befehls Reaktivierung im Command Fenster.

Variable Fenster Überblick über alle Variablen des Datensatzes; es kann jeweils nur ein Datensatz gleichzeitig bearbeitet werden.

Properties Fenster Ab Stata 12 gibt es dieses Fenster, hier wird die jeweils angeklickte Variable beschrieben. Eine Veränderung der Beschreibung ist auch hier möglich.

Data Editor (Browse) Betrachtung und Veränderung des Datensatzes. Wird aufgerufen über `.br` oder `.edit`. Bei letzterem hat man Bearbeitungsrechte.

Do-file Editor Führt mehrere Befehle sukzessive automatisch aus. Dient zur Dokumentation. Aufzurufen mit `.doedit`.

1.2 Tippen vs. klicken

Stata bietet die Möglichkeit entweder Befehle direkt über die Kommandozeile *einzutippen*, oder über eine graphische Benutzeroberfläche zu *klicken*. Neuere und/oder selten genutzte Befehle lassen sich nur über die Kommandozeile aufrufen. Das Gleiche gilt für viele Optionen von Befehlen. Die Erfahrung zeigt, dass es langfristig meist schneller und effizienter ist zu *tippen*. Um die Befehle kennenzulernen ist die Menü-gesteuerte Benutzeroberfläche jedoch sehr hilfreich da der Befehl den man durch das *klicken* auslöst angezeigt wird.

1.3 Die Befehlssyntax

Alle Befehle in Stata sind nach dem gleichen Prinzip aufgebaut. Die genaue Form der einzelnen Befehle ist der Hilfe zu entnehmen. Im Allgemeinen sieht sie wie folgt aus:¹

```
[by varlist1:] Befehl [varlist2] [weight] [if exp] [in #] [using file] [, options]
```

Dabei sind Elemente ohne eckige Klammern zwingend anzugeben, solche in eckigen Klammern optional möglich. Optionen folgen einem Komma. Viele Befehle können auch abgekürzt werden. In wie weit dies möglich ist, zeigt der unterstrichene Teil des Befehls an.

1.4 Beispiel

Um mit Stata Daten bearbeiten zu können, müssen diese geladen sein. Durch `.sysuse` können bspw. Datensätze geladen werden die Teil des Stata Software Packages sind. Diese Datensätze werden gerne in der Stata Hilfe zu Demonstrationszwecken herangezogen. Sobald Daten geladen sind, können Befehle über das Menü oder über die Kommandozeile angesteuert werden. Hier ein Beispiel wie man Grafiken, Tabellen oder Statistiken erstellen kann:

```
.sysuse auto, clear  
.describe
```

¹Im Folgenden werden Stata Befehle mit vorangehendem Punkt und in der Typewriter Schriftart zumeist blau angezeigt.

```

Contains data from C:\Program Files (x86)\Stata12\ado\base/a/auto.dta
  obs:          74          1978 Automobile Data
  vars:         12          13 Apr 2011 17:45
  size:        3,182        (_dta has notes)

```

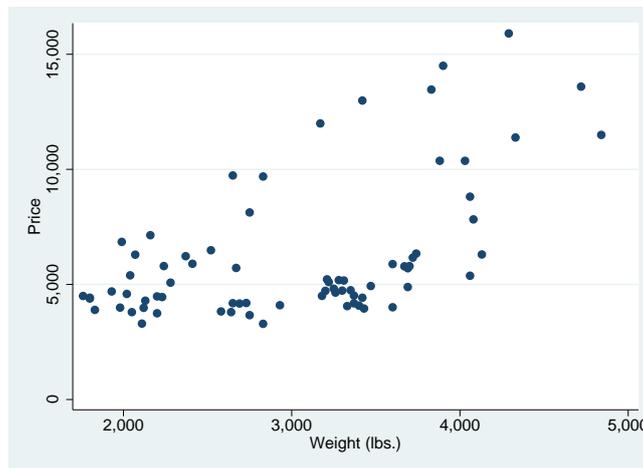
variable name	storage type	display format	value label	variable label
make	str18	%-18s		Make and Model
price	int	%8.0gc		Price
mpg	int	%8.0g		Mileage (mpg)
rep78	int	%8.0g		Repair Record 1978
headroom	float	%6.1f		Headroom (in.)
trunk	int	%8.0g		Trunk space (cu. ft.)
weight	int	%8.0gc		Weight (lbs.)
length	int	%8.0g		Length (in.)
turn	int	%8.0g		Turn Circle (ft.)
displacement	int	%8.0g		Displacement (cu. in.)
gear_ratio	float	%6.2f		Gear Ratio
foreign	byte	%8.0g	origin	Car type

```

Sorted by:  foreign

```

```
.scatter price weight
```



```
.summarize price weight
```

Variable	Obs	Mean	Std. Dev.	Min	Max
price	74	6165.257	2949.496	3291	15906
weight	74	3019.459	777.1936	1760	4840

```
.reg price weight
```

Source	SS	df	MS	Number of obs = 74		
Model	184233937	1	184233937	F(1, 72) =	29.42	
Residual	450831459	72	6261548.04	Prob > F =	0.0000	
Total	635065396	73	8699525.97	R-squared =	0.2901	
				Adj R-squared =	0.2802	
				Root MSE =	2502.3	

	price	weight	_cons	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
				2.044063	.3768341	5.42	0.000	1.292857 2.795268
				-6.707353	1174.43	-0.01	0.995	-2347.89 2334.475

Im Menü würden sich diese Befehle hier finden lassen:

- *File/Open*
- *Data/Describe data/Describe data in memory*
- *Graphics/Two-way Graph*
- *Data/Describe data/Summary statistics*
- *Statistics/Linear Models and related/Linear Regression*

1.5 Log-files

Mit `.log using log_name` wird ein log-file geöffnet, bzw. erstellt und aktiviert. Mit `.log close` wird dieses geschlossen und abgespeichert. Alles was dazwischen im Output-Fenster angezeigt wird, wird im log-file gespeichert. Damit lässt sich eine lückenlose Dokumentation der Arbeit erstellen.

1.6 Do-Files

Do-files sind eine der Stärken von Stata. Ein do-file dient der Dokumentation und der einfachen Replikation vorangegangener Arbeitsschritte. In ein Do-File lassen sich mehrere Befehle untereinander schreiben und automatisch ausführen. Grundsätzlich gibt es drei Möglichkeiten um ein do-file „laufen zu lassen“: 1. klicken des Run-, bzw. Do-Buttons, 2. gleichzeitiges tippen von Strg+D oder 3. durch den Befehl `.do do-filename`. Bei 1. und 2. muss das do-file geöffnet und aktiv sein. Zusätzlich besteht hier die Möglichkeit durch das Markieren bestimmter Zeilen ein do-file nur ausschnittsweise auszuführen. Prinzipiell führt ein do-file die in ihm enthaltenen Befehle sukzessive also Zeile für Zeile aus. Das bedeutet, jeder Zeilenumbruch wird als ein neuer Befehl verstanden. Wird jedoch `.#delimit;` eingegeben, wird ein Befehl erst dann ausgelöst wenn ein „;“ (Strich-Punkt) kommt. Dadurch können lange Befehle über mehrere Zeilen hinweg „optisch“ gegliedert werden. Durch die Eingabe von `.#delimit cr` kehrt man zur ursprünglichen Methode zurück. Soll ein Kommentar in ein do-filer geschrieben werden, kann dies entweder durch drei „///“ (Querstriche) am Ende einer Zeile geschehen, oder durch ein „*“ (Sternchen) am Anfang einer Zeile, oder es wird alles „aus-dokumentiert“ was zwischen „/*“ und „*/“ steht.

Hier ein Beispiel für ein do-file:

```
/* Dies ist ein Beispiel
fuer ein do-file*/

* das Outputfenster laeuft durch
set more off
* setzen der working directory
cd C:/stata

/**** Hier folgt die Analyse: ****/
* Daten laden:
sysuse auto, clear
* Datensatz beschreiben:
describe
* Zusammenhang von Preis und Gewicht veranschaulichen:
scatter price weight
* Bild abspeichern
graph export graphs/bild.png
* Deskriptive Statistik:
summarize price weight
* Regression: beeinflusst das Gewicht den Preis
reg price weight
* abspeichern der Datei
save data/Musterbeispiel.dta, replace
exit
```

Ab Stata 13 gibt es die Möglichkeit do-files im Rahmen von sogenannten Projekten im Do-File Editor etwas übersichtlicher zu organisieren. Dies bietet sich insbesondere dann an, wenn im Rahmen der Bearbeitung auf mehrere do-files zurückgegriffen wird.

2 Grundlegendes

2.1 Aktuelle Version

Stata liegt mittlerweile in der 17. Version vor. Wenngleich jede Version neue Möglichkeiten bietet, hat sich die wesentliche Bedienbarkeit seit der Version 9 nicht verändert. Hin und wieder ändern sich die Funktionsweisen von Befehlen, dies wird jedoch zumeist von Stata angezeigt. Manchmal können ältere Versionen Datensätze nicht lesen, welche in neueren Versionen abgespeichert wurden. Soll dieses Problem vermieden werden, empfiehlt es sich mit dem Befehl `.saveold` anstatt mit `.save` zu arbeiten. Eine stets aktuelle Quelle zu den Veränderungen findet man hier: www.stata.com/new-in-stata/

2.2 Hilfe

Stata verfügt über eine sehr ausführliche und im Gegensatz zu anderen Programmen wirklich hilfreiche Hilfe und eine ausführliche sowie vollständige Beschreibung der in Stata implementierten Software (StataCorp, 2015). Zu erreichen ist die Hilfe, über das Menü →Help oder durch den Befehl `.help`. Weiß man schon zu welchem Befehl man nähere Informationen benötigt, kann man diesen Befehl dem `.help` folgen lassen. Will man allgemeiner suchen ist dies mit `.search`, `.net search` und `.findit` möglich. Für ein effizientes Arbeiten und ein ordentliches Verständnis, ist es unerlässlich die Hilfe zu benutzen. Die zahlreichen Anwendungsbeispiele hierin erleichtern den Zugang zu den Befehlen wesentlich. Pragmatisch und oftmals hilfreich ist es nach Hilfe zu „googlen“.

2.3 Literaturempfehlungen

Da Anzahl der Lehrbücher und Hilfestellungen zur Benutzung von Stata ist enorm. Besonders übersichtlich und umfangreich ist die Sammlung an Links und Lehrmaterial der UCLA (www.ats.ucla.edu/stat/stata) und Princeton (<http://dss.princeton.edu/training>). Eine stets aktuelle Übersicht zu Bücher über Stata findet man hier: <http://www.stata.com/bookstore/books-on-stata> und hier: www.stata.com/bookstore/stata-press-books/. Eine anschauliche Einleitung mit hilfreichen Links ins Internet findet man hier: <http://dss.princeton.edu/training/StataTutorial.pdf> Hier eine Auswahl empfehlenswerter Bücher:

- Hamilton, Lawrence C. 2013. *Statistics with Stata: Version 12*. Cengage, 8 ed
- Kohler, Ulrich and Frauke Kreuter. 2016. *Datenanalyse mit Stata: Allgemeine Konzepte der Datenanalyse und ihre praktische Anwendung*. Oldenbourg: De Gruyter, 5. ed
- Acock, Alan C. 2018. *A Gentle Introduction to Stata*. Stata Press, 6 ed
- Cameron, A. Colin and Pravin K. Trivedi. 2010. *Microeconometrics Using Stata*, vol. 5. Stata Press, revised ed
- Mitchell, Michael N. 2012. *A Visual Guide to Stata Graphics*. Stata Press, 3 ed
- ———. 2020. *Data Management Using Stata: A Practical Handbook*. Stata Press, 2 ed
- ———. 2021. *Interpreting and Visualizing Regression Models Using Stata*. Stata Press, 2 ed

2.4 Das Arbeitsverzeichnis

Um in Stata Daten aufzurufen, muss man entweder das genaue Verzeichnis nennen oder das Arbeitsverzeichnis aus welchem Stata Daten abspeichert und aufruft ändern. Folgende Befehle erweisen sich hier als nützlich:

`.pwd` In welchem Verzeichnis bin ich gegenwärtig? (present working directory)

`.cd` Ändern des gegenwärtigen Verzeichnisses (change directory)

`.dir` Anzeigen aller Dateien im gegenwärtigen Verzeichnis

`.dir *.xxx` Anzeigen aller Dateien mit der Endung xxx im gegenwärtigen Verzeichnis

`.sysdir` Anzeigen der von Stata intern verwendeten Verzeichnisse. Veränderung hier können mit `.sysdir set` vorgenommen werden.

`.erase file.xxx` Löschen der Datei file.xxx

`.mkdir` Erstellen von neuen Ordnern

Wird Stata auf verschiedenen Rechnern benützt, ist es empfehlenswert mit relativen Pfaden zu arbeiten oder den Pfad durch ein globales Makro aufzurufen.

Arbeiten mit relativen Pfaden:

```
cd "C:\Eigene_Dateien"
use data\daten.dta, clear
[...]
save results\Results.dta, replace
```

Pfad mit globalen Makro ansprechen

```
global root="C:\Eigene_Dateien"
cd $root
use $root\data\daten.dta, clear
[...]
save $root\results\Results.dta, replace
```

2.5 Updates

Durch den Befehl `.update [all]` wird Stata upgedated. Mit `.adoupdate` werden von Benutzern geschriebene Befehle upgedated.

2.6 Von Benutzern geschriebene Befehle (ado-files)

Ein Befehl in Stata ist letztlich eine Datei mit der Endung *.ado (automatic do-file). Alle Statabefehle sind in bestimmten Ordnern auf der Festplatte abgelegt. Wo sich diese befinden, lässt sich mit `.sysdir` herausfinden. Neue Befehle kann man einfach in einen dieser Ordner ablegen und benutzen.² Eine Vielzahl von nützlichen benutzergeschriebenen Befehle findet man online und insbesondere im ssc Archive (siehe `.help net_mnu`). Die dort befindlichen Befehle lassen sich komfortabel mit

```
.ssc install Befehl
```

installieren. Alternativ kann man einen Befehl auch mit `.findit` suchen, finden und per Klick installieren. Mit `.ssc hot, n(100)` können die hundert beliebtesten von Benutzer geschriebenen Befehle betrachtet werden. Durch `.ssc new` kann man sich die neuesten Befehle anschauen. Wie ein Befehl in Stata aufgebaut ist und wie man selbst programmiert, wird in Abschnitt 2.7 gezeigt.

Ein gutes Beispiel für einen wirklich nützlichen benutzergeschriebenen Befehl ist `.distinct`. Dieser lässt sich durch `.ssc install distinct` installieren.

Achtung: Existieren keine Benutzerrechte für diese Verzeichnisse kann kein Befehl von Stata aus installiert werden. Dies ist erst möglich, nachdem man das Verzeichniss, in dem Stata die benutzergeschriebenen Befehle speichert, verändert hat. Dies ist bspw. wie folgt möglich: `.sysdir set PLUS e:\Eigene_Dateien` Hierbei wird davon ausgegangen, dass auf dem Laufwerk „e:\“ Schreibrechte existieren.

²Sollte eine ado-file nicht gefunden werden, führen Sie den Befehl `.discard` aus.

Ich selbst habe folgende Befehle geschrieben:

- Huber, Stephan and Christoph Rust. 2016. Calculate travel time and distance with OpenStreetMap data using the Open Source Routing Machine (OSRM). *The Stata Journal* 16 (2):416–423

Der Befehl `.osrmtime` von (Huber and Rust, 2016) ermöglicht die Länge von Reiserouten und die Fahrzeiten berechnen lassen. Google Maps kann dies auch, jedoch erlaubt `.osrmtime` dies offline und sehr effizient zu tun (>300 Abfragen pro Sekunde). Aktuelle Entwicklungen und Beschreibungen zur Installation finden sich hier: <https://github.com/christophrust/osrmtime>

- Huber, Stephan. 2017a. EXPY: Stata Module to Calculate the EXPY-Index as Proposed by Hausmann et al. (2007). Statistical Software Components S458328, Boston College Department of Economics
- ———. 2017d. simcadi: Similarity Indices for Categorical Distributions. Tech. rep., SSRN. DOI: 10.2139/ssrn.2870834
- ———. 2017c. PRODY: Stata Module to Calculate Factor Intensity and Sophistication Indicators. Statistical Software Components S458329, Boston College Department of Economics
- ———. 2020. SXPOSE2: Stata Module to Transpose String and Numeric Variable Dataset Including Variable Names and Labels. Statistical Software Components, Boston College Department of Economics. URL <https://ideas.repec.org/c/boc/bocode/s458854.html>

2.7 Selbst programmieren

In Stata lassen sich relativ einfach eigene Abfolgen von Befehlen selbst programmieren. Um z.B. ein Programm zu schreiben welche von einer Variable den Mittelwert abzieht kann dies wie folgt geschehen:

```
clear all
sysuse auto
program define demean
    foreach var of local 0 {
        quietly sum `var'
        replace `var'=`var'-r(mean)
    }
end
demean price mpg
```

Mit der Eingabe von `.demean price` werden von allen Werten der Variable price der Mittelwert von price abgezogen. Soll dieses demeaning für mehrere Variablen gleichzeitig geschehen, kann wie folgt vorgegangen werden:

```
clear all
sysuse auto
program define demean
    foreach var of local 0 {
        quietly sum `var'
        replace `var'=`var'-r(mean)
    }
end
demean price mpg
```

Die „0“ bedeutet hier, dass jedes Argument in der Variablenliste gemacht werden soll. So wird für `.demean price mpg` das ‘demeaning’ für price und mpg durchgeführt. Will man verhindern, dass der Befehl abbricht, wenn beim Erzeugen einer string variable ein Fehler auftaucht, so kann dies wie folgt

geschehen:³

```
clear all
sysuse auto
program define demean
    foreach var of local 0 {
        capture confirm numeric variable `var'
        if _rc==0 {
            quietly sum `var'
            replace `var'=`var'-r(mean)
        }
        else di `var' is not a numeric variable and cannot be demeaned."
    }
end
demean price mpg
```

Um ein Programm zu schreiben welches einem anzeigt wie viele verschiedene Ausprägungen eine Variable beinhaltet, kann man wie folgt vorgehen:⁴

```
program def varcount, rclass
    args varname
    sort `varname'
    by `varname' : gen var_count_lfd=_n
    qui count if var_count_lfd==1
    return scalar N=r(N)
    display as result "sum of distinct values of `varname' is: `r(N)'"
    drop var_count_lfd
    exit
end
sysuse auto, clear
varcount make
```

Ein einmal geladenes Programm ist nach dem schließen von Stata nicht mehr aufrufbar. Dieses muss erneut geladen werden. Soll ein Befehl dauerhaft zur Verfügung stehen, so kann dieser als automatic-do-file mit der endung `.ado` in einen der Ordner gespeichert werden die Stata jederzeit nach Befehlen durchsucht (siehe Abschnitt 2.6). Will man ein geladenes Programm in einer Sitzung neu definieren, so kann dieses Programm durch `.program drop [program name]` gelöscht werden.

Wer einen ausführlichen Einblick in die Welt des Programmierens mit Stata haben will, dem sei die Lektüre von Baum (2016) empfohlen.

2.8 Das Betriebssystem mit Stata steuern

Mit dem dem Befehl `.shell` kann man das aktive Betriebssystem steuern. Linux Benutzer sind es vielleicht eher gewohnt, über die Kommandozeile ihr Betriebssystem zu steuern, aber auch Windows und Mac Benutzern sei diese Möglichkeit empfohlen, um die vom Betriebssystem bereitgestellten Befehle zu benutzen.

2.9 Stata Besonderheiten

Missings Stata stellt fehlende Datenpunkte so genannte *missings* als „(Punkt)“ dar. Speichert diese jedoch intern als eine sehr große Zahl ab. Missings werden von vielen Befehlen teilweise etwas intransparent gehandhabt. Bei der Generierung von (Dummy-)Variablen etwa sollte anstatt `.gen y=1 if x>0` besser `.gen y=1 if (x>0 & x!=.)` angegeben werden, um zu vermeiden, dass missings fälschlicherweise den Wert 1 zugewiesen bekommen.

case-sensitive Stata ist “case sensitive”! Sprich: Groß und Kleinschreibung bedeuten Unterschiedliches.

Abspeicherung der Daten Stata speichert Daten sehr effizient, jedoch in binärer Form. Leider besitzt nicht jede Zahl eine exakte binäre Repräsentation. Wenn man dies nicht bedenkt kann das zu schwerwiegenden Missverständnissen führen. Details zu dieser Besonderheit können im

³Hier wird die `if{...}else{...}` Bedingung verwendet. Diese ist insbesondere wenn es um das schreiben von Programmen angeht sehr hilfreich. Näheres hierzu findet sich hier: `.help ifcmd`

⁴Vergleichen Sie dieses Programm einmal mit `.distinct` und `.unique` .

Abschnitt „Precision and problems therein“ des Kapitels „Functions and expressions“ des Stata User Guides 14 (StataCorp, 2015, S.177f) nachgelesen werden. Folgendes Beispiel soll das Problem darstellen und eine Lösung präsentieren:

```
clear
set obs 10
gen lfd=1.3
count if lfd==1.3
count if lfd==float(1.3)
```

3 Daten Management

3.1 Datenstruktur

Um Daten analysieren zu können, muss die Struktur der Daten klar sein: Welche Informationen stecken in den Daten und welche identifizierenden Merkmale kennzeichnen die Daten? Grundsätzlich stehen Daten in einem Tableau mit Zeilen und Spalten. Bevor mit diesen Daten gearbeitet werden kann, sollte man sich stets bewusst sein, welche Struktur die Daten besitzen. Sprich: Durch welche Variablen werden die Beobachtungen identifiziert? Hierzu ist klarzustellen, welche Variable eine **identifizierende Variable** ist. Diese Variablen identifizieren die einzelnen Zeilen eines Datensatzes. In einem Querschnittsdatsatzes ist dies bspw. das Land. In einer Zeitreihe ist dies bspw. das Jahr und in einem Panel Datensatz die Variablen Land und Jahr. Um herauszufinden, welche Variablen einen Datensatz identifizieren kann der Befehl `.duplicates` sehr hilfreich sein. Variablen die nicht identifizierend wirken, enthalten verschiedene Informationen. In einem Querschnittsdatsatz bspw. das BIP. In einer Zeitreihe bspw. den Goldpreis und in einem Panel Datensatz bspw. das BIP, die Zugehörigkeit zu einer Ländergruppe (Europa, Asien, Afrika, etc.).

3.2 Allgemeine Informationen zur Befehlsstruktur

Es gibt verschiedene häufig wiederkehrende Strukturen bzw. Komponenten in Stata. Im folgenden sollten die gängigsten kurz angesprochen werden. Detailliertere Informationen und Beispiele sind der Stata Hilfe zu entnehmen. Die Anwendung ist oft sehr intuitiv, dennoch ist es notwendig sich die Funktionsweise mit Beispielen klarzumachen.

3.2.1 Variablen

Grundsätzlich unterscheidet Stata zwischen zwei Arten von Variablen. Zum Einen *String-Variablen* und zum Anderen *numerische Variablen*. Erstere enthalten Buchstaben und Zahlen, letztere ausschließlich Zahlen. Eine dazwischen liegende Variablenform ist die der gelabelten numerischen Variable. Wie diese erzeugt wird ist dem Abschnitt 3.6.1 zu entnehmen. Vielfach muss für einen Befehl spezifiziert werden, auf welche Variable oder welche Variablen er angewendet werden soll. Wird keine Variable oder Variablenliste spezifiziert, so wird der Befehl–falls möglich–für alle Variablen im Datensatz ausgeführt. Einige Möglichkeiten Variablen verkürzt aufzulisten sind:

<code>x</code>	Befehl für Variable x
<code>x a y</code>	Befehl für Variablen x, a und y
<code>x-z</code>	Befehl für Variablen x-z
<code>f*</code>	Befehl für alle Variablen, die mit “f” beginnen
<code>*t</code>	Befehl für alle Variablen, die mit “t” enden

Der Befehl `.order varlist` ermöglicht es die Variablen in einem Datensatz in eine bestimmte Reihenfolge zu bringen. Bspw. werden durch `.order *, alphabetical` die im Datensatz enthaltenen Variablen in eine alphabetische Reihenfolge gebracht. Durch `.rename old_varname new_varname` können Variablen umbenannt werden. Einige elegante Möglichkeiten zum Umbenennen von Variablen bietet der Befehl `.renvars`.

3.2.2 Beobachtungen

Jede Zelle einer Variable beinhaltet eine Beobachtung. Diese Beobachtungen können durch `.sort` und `.gsort` sortiert und durch `.count` gezählt werden. Mit `.assert` kann abgefragt werden, ob und wie viele Beobachtungen bestimmte Bedingungen erfüllen. Bspw. wird durch `.assert x>22` erfragt, ob alle Ausprägungen der Variable `x` größer als 22 sind. Zusätzlich wird die Anzahl der Beobachtungen angezeigt welche diesen Wert annehmen. Durch `.set obs` kann die Anzahl der Beobachtungen gesetzt bzw. erweitert werden. Beobachtungen in String-Variablen werden mit Anführungszeichen angesprochen. Mit `.list` können alle Beobachtungen im Outputfenster betrachtet werden.

3.2.3 Labels

Häufig sind Datensätze groß und unübersichtlich. Stata bietet daher die Möglichkeit sogenannte labels zu vergeben (siehe `.help label`). Unter anderem können Variablen (`.label variable varname`) oder der komplette Datensatz (`.label data "Data Name"`) gelabelt und damit beschrieben werden. Der Vorteil von gelabelten Variablen ist, dass bspw. an den Achsen einer Grafiken nicht der in Stata häufig verkürzte und wenig aussagekräftige Variablenname angezeigt wird, sondern das Label. Also bspw. anstatt `gdppp` kann durch Eingabe von `.label gdppp "Gross Domestic Demand per person in 1000 US-$"` das Label *Gross Domestic Demand per person in 1000 US-\$* angezeigt.

3.2.4 Operatoren

Ausdrücke können in Stata algebraischer oder logischer Art sein. Sie werden an die Stelle von `exp` im Syntaxdiagramm geschrieben und setzen sich aus Operatoren, Funktionen, Ziffern, Platzhaltern und strings zusammen. Das einfache “=”-Zeichen gibt es in Stata auch, dieses wird aber nicht als mathematischer Operator verwendet. Vielmehr ist es zu verstehen als ein “setze das gleich”. Zum Beispiel bedeutet der Befehl `.generate x=1 if y==3` generiere eine Variable `x` und setze diese gleich eins wenn die Variable `y` den Wert drei annimmt.

<u>Arithmetisch</u>		<u>Logisch</u>		<u>Relational</u>	
+	Addition	~	nicht	>	größer als
-	Subtraktion	!	nicht	<	kleiner als
*	Multiplikation		oder	>=	größer gleich
/	Division	&	und	<=	kleiner gleich
^	Potenz			==	gleich
				~=	nicht gleich
				!=	nicht gleich

Bei der Analyse der Zeitreihendaten verwendet man oft die Zeitreihenoperatoren. Um diese zu generieren, muss zunächst der Zeitreihendatensatz als solcher deklariert werden. Dies kann mit dem Befehl `.tsset` getan werden. Ist der Datensatz als Zeitreihendatensatz deklariert, können Zeitoperatoren verwendet werden. Näheres hierzu, siehe <https://www.stata.com/manuals/u11.pdf> (Abschnitt 11.4.4)

3.2.5 Die in-Bedingung

Diese Bedingung schränkt den Befehl auf einen bestimmten Beobachtungsbereich ein. Dabei ist die Sortierung des Datensatzes entscheidend.

`in 5` nur die fünfte Beobachtung
`in 1/20` nur die ersten 20 Beobachtungen
`in -5/-1` nur die letzten 5 Beobachtungen

3.2.6 Die if-Bedingung

Diese Bedingung schränkt den Befehl auf die Beobachtungen ein, für die der anschließende Ausdruck wahr ist. Eine Verknüpfung von Bedingungen lässt sich erzeugen durch “&” oder “|”. Ersteres bedeutet

ein logisches *und*, Zweiteres ein logisches *oder*. Diese Kombinierten Verbindungen müssen mit einfachen Klammern eingefasst werden.

```
if x==1      nur solche Beobachtungen, für die x 1 ist
if z<=5      nur solche Beobachtungen, für die x kleiner 5 ist
if t~=.      nur solche Beobachtungen, für die t kein missing aufweist
```

3.2.7 Makros

Makros sind Container, in denen beliebige Information zum späteren Gebrauch gespeichert werden können. Lokale Makros gelten nur innerhalb eines spezifischen Programms/Do-Files, globale Makros überall. Beim Verlassen von Stata gehen alle Makros verloren. Lokale Makros werden gespeichert mit `.local Makroname Makroinhalt`. Resultate werden per “=” zugewiesen: `.local a = 1+1`. Globale Makros werden gespeichert mit `.global Makroname Makroinhalt`. Wieder angesprochen werden lokale Makros mit ‘Makroname’, d.h. eingeklammert in einen accent grave (‘) und ein einfaches Anführungszeichen (’). Ersteres ist auf der Tastatur anzusprechen, indem man auf Shift und zugleich auf die Taste rechts neben dem Fragezeichen klickt und danach das Leerzeichen antippt. Letzteres indem man Shift und die Taste mit der Raute # klickt. Zumindest ist dies auf den meisten deutschen Tastaturen so angeordnet. Globale Makros werden angesprochen mit einem vorangestellten Dollarzeichen: (`$Makroname`). Durch Eingabe von `.macro list` werden alle gespeicherten Makros angezeigt.

Viele Stata-Kommandos speichern ihre Ergebnisse in Makros ab. Diese können weiterverwendet werden. Diese Container werden allerdings von neuen Kommandos überschrieben, und sollten daher, falls nötig, gesichert werden (z.B. in Makros). Die Bezeichnungen der Container werden angezeigt, indem man nach Statistikbefehlen `.return list` und nach Modellbefehlen `.ereturn list` eingibt. Die daraufhin angezeigten Makros können, je nach Klasse (Skalare, Matrizen, Makros), wie oben beschrieben weitergehend verwendet werden. Ein Beispiel wie eine lokale Variable in einer Schleife verwendet wird, findet sich auf Seite 16.

3.3 Daten einlesen

Daten werden in Stata durch `.use`, `.sysuse` oder `.webuse` aufgerufen. Mit `.clear` wird der Arbeitsspeicher von Stata geleert. Oft liegen liegen nicht im gewünschten Stata-Format (*.dta) vor. Stata bietet verschiedene Möglichkeiten, um diese zu importieren. Über die Menüleiste `→File→Import` lassen sich die verschiedenen Einleseprozeduren komfortabel steuern. Stata kann die gängigsten Formate effizient lesen. Es gibt auch externe Datei-Konvertierungsprogramme, bspw. *Stat Transfer*. Generell ist zu betonen, dass beim Import von Datensätze Vorsicht geboten ist. Wenn möglich, sollte der Importprozess genau überwacht werden und replizierbar sein.

Oft können Daten per *copy-and-paste* in den Datenbrowser übertragen werden oder händisch in den Daten-Editor eingetragen werden. Dies ist jedoch weniger zu empfehlen, da sich hier schnell Fehler einschleichen und diese Fehler oft nicht auffallen, auch weil eine Replizierung des Prozedere nur beschränkt möglich ist. Folgende Befehle helfen beim einlesen von Daten: `.xmluse`, `.insheet`, `.infix`, `.infile`, `.import excel`. Für genauere Informationen siehe `.help import`.

3.4 Daten speichern

Um geladene Daten zu speichern, ist der Befehl `.save` zu benutzen. Mit Hilfe der Option `replace` wird ein bereits vorhandener Datensatz überschrieben. `.saveold` ist zu verwenden, um den Datensatz auch mit früheren Versionen als Stata 12 lesen zu können. Für ein effizientes speichern ist es sinnvoll `.compress` einzugeben, dadurch werden alle Variablen mit der kleinstmöglichen Genauigkeit gespeichert. *Kleinstmöglich* bedeutet hier: Nachkommastellen die keine Information beinhalten werden gelöscht.

3.5 Daten kombinieren

Will man zwei Stata Datensätze kombinieren, gelingt dies mit `.append`, wenn zu dem geladenen Datensatz zusätzliche Beobachtungen hinzugefügt werden sollen, oder mit `.merge`, wenn neue Variablen zu dem geladenen Datensatz hinzugefügt werden sollen.

3.6 Daten erstellen und manipulieren

3.6.1 Variablen erzeugen und manipulieren mit...

...**drop und keep** Mit `.drop` /`.keep` kann man ganze Variablen löschen/behalten, oder auch einzelne Beobachtungen. Für Letzteres müssen noch in- oder if Bedingungen spezifiziert werden. Der Befehl `.dropmiss` löscht alle Variablen, welche ausschließlich missings enthält (das kommt tatsächlich vor). Dieser Befehl muss allerdings erst installiert werden (`.ssc install dropmiss`).

...**generate, egen, egenmore, replace und recode**: Mit `.generate` erstellt man neue Variablen. Unter `.egen` und `.egenmore` finden sich eine ganze Reihe von fortgeschrittenen Arten Variablen nach einer bestimmten Logik zu generieren. Mit `.replace` kann man den Inhalt von existierenden Variablen verändern. `.recode` ermöglicht das auch, jedoch in effizienterer Weise. Diese Befehle machen starken Gebrauch der in- und if-Bedingung (siehe Abschnitt 3.2.5 und 3.2.6).

...**mvencode und mvdecode**: Mit `.mvdecode` und `.mvencode` werden missings in bestimmte Zahlen und vice versa verändert. Bspw. überschreibt `.mvencode _all, mv(999)` alle missings eines Datensatzes mit 999.

...**encode und decode** Mit `.encode` lassen sich gelabelten numerischen Variable erzeugen. Bspw. würde durch `.encode x, gen(x_n)` eine numerische Variable `x_n` erzeugt, welche jedem Wert der String Variable `x` eine Zahl zuweist. Die Zuweisung kann durch `.label list x_n` eingesehen werden. Rückgängig kann dies mit `.decode` gemacht werden.

...**tostring und destring** Mit `.tostring` (/ `.destring`) kann eine numerische-(/string-) Variable in eine string-(/numerische-) Variable konvertiert werden.

...**xi**: Eine extrem schnelle Methode viele Variablen zu erzeugen bietet der Befehl `.xi [, prefix(string) noomit]: any_stata_command varlist_with_terms`. Durch Eingabe von `.xi i.year, prefix(_DY)` wird bspw. für jede Ausprägung der Variable `year` eine Dummy Variable mit entsprechendem Wert und dem Prefix „_DY“ erzeugt.

...**tabulate**: Ähnlich wie mit `.xi` lassen sich mit `.tabulate` eine Reihe von Dummyvariablen automatisch erzeugen. Durch Eingabe von `.tabulate year, generate(_DY)` wird bspw. für jede Ausprägung der Variable `year` eine Dummy Variable mit entsprechendem Wert und dem Prefix „_DY“ erzeugt.

...**xpose**: Ein Datensatz kann mittels des Befehls `.xpose` transponiert werden. Das heißt, dass Spalten und Zeilen vertauscht werden.

...**collapse**: Oftmals soll einen Datensatz komprimiert werden, indem man für verschiedene Gruppen bestimmte statistische Kennwerte berechnet. Der entsprechende Befehl lautet `.collapse`. Durch `.collapse` können deskriptive Statistiken berechnet werden. Der Befehl überschreibt dabei die bestehenden Daten im Speicher. Folgendes Beispiel überschreibt den im Speicher geladenen Datensatz und erzeugt genau eine Beobachtung die in der Variable `mpg` gespeichert wird: den Mittelwert der Variable `mpg`.

```
.sysuse auto
.collapse (mean) mpg
```

Der `collapse` Befehl eignet sich insbesondere um Tabellen in andere Formate (Excel, LaTeX, ...) zu transformieren, da die Möglichkeiten groß sind mit denen *.dta-File exportiert werden können, siehe Menü→File→Export und Abschnitt 6.6. Folgendes Beispiel veranschaulicht die Funktionsweise des `.collapse` Befehls weiter:

```
.sysuse auto
.collapse (mean) mpg trunk (sum) price, by(foreign)
```

```
.list
```

foreign	mpg	trunk	price
0	19.82692	14.75	315766
1	24.77273	11.40909	140463

...**reshape**: Mittels dem Befehl `.reshape` kann zwischen dem long- (Jede Beobachtung steht in einer separaten Zeile) und dem wide-Format (Ausprägungen derselben Beobachtung werden in dieselbe Zeile verschoben) hin und her gewechselt werden.

...**sxpose2**: `.sxpose2` lässt sich durch `.ssc install sxpose2` installieren und transponiert die Daten. Sprich er macht die Variablen zu Beobachtungen und die Beobachtungen zu Variablen. Anders als `.sxpose` erlaubt er, dass die Variablenamen behalten werden. Beispiel:

id	var1	var2
a	d	g
b	e	h
c	f	i

```
.sxpose2, clear firstnames varlabel varname
```

_varname	_varlabel	a	b	c
var1	VAR-1	d	e	f
var2	VAR-2	g	h	i

...**fillin** Mit `.fillin` wird ein Datensatz derart erweitert, dass alle Kombinationen der aufgelisteten Variablen aufgelistet sind. Beispiel:

```
.webuse fillin1
```

```
.list
```

sex	race	age_group	x1	x2
female	1	1	20393	14.5
male	1	2	32750	12.7
female	2	3	39399	14.2

```
.fillin sex race age_group
```

```
.list
```

sex	race	age_group	x1	x2	_fillin
female	1	1	20393	14.5	0
female	1	2			1
female	1	3			1
female	2	1			1
female	2	2			1
female	2	3	39399	14.2	0
male	1	1			1
male	1	2	32750	12.7	0
male	1	3			1
male	2	1			1
male	2	2			1
male	2	3			1

...**split**: Teilt eine String Variable in einzelne Bestandteile auf. Dies ist für allem praktisch wenn bspw. eine Jahreszahl aus einer Stringvariable 'extrahiert' werden soll, wie das folgende Beispiel zeigt:

```
. webuse splitxmpl3, clear
```

```
. list
```

	date
1.	January 21, 1952
2.	July 11, 1948
3.	May 31, 1971
4.	October 7, 2000

```
. split date, parse(" ", " ") gen(ndate)
```

```
variables created as string:
```

```
ndate1 ndate2 ndate3
```

```
. list
```

	date	ndate1	ndate2	ndate3
1.	January 21, 1952	January	21	1952
2.	July 11, 1948	July	11	1948
3.	May 31, 1971	May	31	1971
4.	October 7, 2000	October	7	2000

3.6.2 Funktionen

Mathematische Funktionen Verschiedenste mathematische Funktionen lassen sich mit Stata erstellen. Mit `.gen varXY= round(gear_ratio), 1` wird bspw. die Variable gear_ratio auf ganze Zahlen gerundet. Mit `gen varXY=log(mpg)` wird bspw. der Logarithmus von mpg erzeugt. Eine genauere Auflistung findet sich hier: [.help functions](#).

String Funktionen Stata bietet eine Reihe von String Funktionen wodurch String Variablen auf unterschiedliche Weise erzeugt und manipuliert werden können (siehe [.help string funtions](#)). Beispielsweise erzeugt `.gen varXY= strlen(make)` eine Variable die indiziert wie viele Buchstaben die Variable make enthält. Oder, `.gen varXY=substr(make, 2,5)` extrahiert die Stellen 2 bis 5 der String Variable make.

4 Schleifen und sonstige Helfer

Schleifen sind eine mächtige Sache und für effizientes programmieren und arbeiten unerlässlich. Wer die Kunst beherrscht Schleifen zu schreiben, der spart sich viel Arbeit und kann scheinbar unmöglich erscheinende Operationen ganz leicht lösen. Prinzipiell funktionieren Schleifen ganz einfach. Sie postulieren zuerst irgendeine Bedingung unter welche dann die folgenden Befehle—eingeklammert durch geschweifte Klammern—immer wieder auszuführen sind. Am besten versteht man diese Schleifen indem man die Beispiele in der Stata Hilfe betrachtet.

Die `.forvalues` ist womöglich die einfachste Schleife. Diese führt für eine zu definierende Menge an Werten eine Reihe von Befehlen aus.

Bspw. ergibt diese Schleife:

```
forvalues x=1/4 {
display "Anfang 'x'"
local loc='x'+1
display 'x'
display "Ende 'x' und 'loc'"
}
```

Folgenden Output:

```
Anfang 1
```

```
1
Ende 1 und 2
Anfang 2
2
Ende 2 und 3
Anfang 3
3
Ende 3 und 4
Anfang 4
4
Ende 4 und 5
```

Ähnlich funktioniert die `.foreach`-Schleife, jedoch kann diese mit noch mehreren Arten von Daten umgehen. Bspw. kann für alle Variablen einer Variablenliste, für eine festgelegte Reihe von Nummern oder Wörtern die eingeklammerte Kette von Befehlen durchgeführt werden.

Die wohl mächtigste Schleife ist die `.while` Schleife. Mit `.while exp stata_commands` führt man eine beliebige Reihe von Befehlen sukzessive und immer wiederholend aus, bis die Bedingung nicht mehr erfüllt ist. Dazu definiert man vorab zumeist eine lokale Variable welche man dann innerhalb der Schleife immer größer oder kleiner werden lässt, bis eine kleiner(<) oder größer(>) Bedingung nicht mehr erfüllt ist und die Schleife ein Ende findet. Gute Beispiele siehe `.help while`.

4.1 Für jede Beobachtung: `.by(sort)` und `.levelsof`

Um für jede Beobachtung einer Variable bestimmte Befehle durchführen zu können, kann der Befehl `.by` bzw. `.bysort` verwendet werden. Mit `.by x: summarize y` wird der Befehl `.summarize` für jede Ausprägung der Variable x ausgeführt. Um jede Beobachtung in ein Makro zu speichern kann der Befehl `.levelsof` Verwendung finden. Durch folgendes:

```
.sysuse auto
.levelsof rep78, local(levels)
```

wird bspw. eine lokale Variable erzeugt die in Schleifen (foreach) verwendet werden kann (siehe `.help levelsof` und `.help foreach`).

4.2 Regressieren mit `.xi`

Wie bereits in Abschnitt 3.6.1 thematisiert können mit `.xi` Dummies schnell erzeugt werden. `.xi` kann aber auch benutzt werden um Dummies speziell für eine Regression zu erstellen:

```
.sysuse auto
.xi: reg price mpg i.foreign i.rep78
```

4.3 Wenn-Dann Abfragen

Ähnlich wie in vielen Programmiersprachen kann man auch in Stata Wenn-Dann Abfragen tätigen. Dies geschieht mit `.if...else...`. Für eine genaue Anleitung siehe `.help ifcmd`.

4.4 Abzählen mit `__n` und `__N`

`__n` gibt die laufende Nummer einer Beobachtung im Datensatz an, `__N` die größte Zahl von `__n`, also die Zahl der Beobachtungen. Durch zusätzliche Bedingungen und das Präfix `bysort` können sie auch für Untergruppen des Datensatzes ausgewiesen werden und so z.B. zur Erstellung von Indizes herangezogen werden. Diese kleinen Systemvariablen können unglaublich flexibel und vielfältig eingesetzt werden. Folgendes Beispiel listet die Beobachtungen, welche für jede Ausprägung von `foreign` die geringste Ausprägung in der Variable `price` besitzt:

```
.sysuse auto
.sort foreign price
.by foreign: gen lfd=__n
```

```
.list if lfd==1
```

Weitere Beispiele finden sich bei Kohler and Kreuter (2016) und in Abschnitt 6.6.

5 Datenanalyse

5.1 Allgemein

Die Befehle `.summarize`, `.codebook`, `.describe`, `.inspect` bieten einen ersten Einblick in die Daten. Die deskriptiven und analytischen Möglichkeiten in Stata sind vielfältig. Praktisch alle gängigen statistischen und ökonometrischen Methoden sind in Stata implementiert. Einen guten ersten Einstieg bietet das Menü →Statistics. Über die Stata Hilfe gelangt man jedoch zu weiteren Befehlen, siehe `.help estimation commands`. Darüber hinaus gibt es eine Reihe von Befehlen die von Benutzern geschrieben wurden. Diese müssen erst noch installiert werden bevor sie in der Hilfe auftauchen und benützt werden können.

Einzig auf die Möglichkeit von Stata Panel Datensätze zu handhaben soll hier hingewiesen werden. Stata hält unter dem Prefix `.xt` eine ganze Batterie an Methoden bereit um die Struktur von Panel Befehlen handhaben zu können. Dazu muss mit `.tsset` erst einmal die Struktur der Daten festgelegt werden.

5.2 Tabellen

5.2.1 `.tabstat` und `.table`

Die zwei gängigsten Befehle um Tabellen zu erstellen sind `.tabstat` und `.table`. Ein guter Einstieg liefert hier die Eingabemaske welche durch das Menü zu erreichen ist. Erweiterte Optionen findet man in der Stata Hilfe.

5.2.2 Regressionen tabellieren mit `.estimates table`

Die Ergebnisse von Regressionen können mit `.estimates store Speichername` gespeichert werden und mit `.estimates table Speichername(n)` tabelliert werden.

```
reg price mpg trunk weight
est store m1
reg price mpg trunk weight if price>5000
est store m2
est table m1 m2, star
est table m1 m2, b(%9.2f) stats(N r2_a)
```

5.3 Grafiken

Einen gelungenen, intuitiven und reichlich bebilderten Einstieg in die Welt der Stata Grafiken bietet Mitchell (2021).

5.3.1 Grafiken speichern, bearbeiten und exportieren

Gespeichert werden kann die aktuell aktive Grafik im Stata-Format `*.gph` mit `.graph save filename.gph` oder durch die Option `saving(filename)`. Exportiert werden können Grafiken durch `.graph export filename.XXX`. Wobei anstatt XXX hier eine Reihe von Formaten möglich sind, bspw. `.pdf`, `.eps`, `.png`, `.wmf` oder `.jpg`. Erwähnenswert ist der Grafik-Editor von Stata. Dieser lässt sich bei geöffneter Grafik durch klicken des entsprechenden Symbols aufrufen und ermöglicht ein einfaches manipulieren des Aussehens der Stata Grafik. Die im Grafik-Editor durchgeführten Schritte lassen sich jedoch nicht durch Befehle replizieren. Deswegen empfiehlt sich die Verwendung nur zur Finalisierung von Grafiken.

5.3.2 Das Aussehen von Grafiken

Das voreingestellte Aussehen von Stata Grafiken wird oft (und zurecht) kritisiert (siehe Bischof and Zurich, 2017). Es gibt schönere Alternativen. Aufgrund der Tatsache, dass sich etwa 9% aller Männer und etwa 0,8% der Frauen von einer Farbenfehlsichtigkeit betroffen sind empfiehlt es sich die Grafiken so zu gestalten, dass diese Personen ebenso gut Ihre Grafiken lesen können. Hierzu kann man entweder auf Darstellungen in Graustufen zurückgreifen oder die von Bischof and Zurich (2017) erstellten ‘graphic schemes’ (`.ssc install blindschemes`). Ich empfehle folgende ‘graphic schemes’: lean2, 538, plotplain, plotplainblind, plottig, plottigblind. Die meisten alternativen ‘graphic schemes’ müssen erst installiert werden. Ich empfehle, die jeweiligen ‘graphic schemes’ mit `.search` zu suchen und zu installieren. Durch Eingabe von `.set scheme lean2, permanently`, lässt sich das voreingestellte ‘graphic scheme’ dauerhaft verändern.

5.3.3 .twoway

Mit `.twoway` können mehrere Grafiken gleichen Typs in ein Bild subsumiert werden. Dazu sind die einzelnen Befehle durch Einklammerung (`()` oder `||`) voneinander zu trennen. Bitte beachten sie die vielfältigen twoway-graph optionen. Bspw. wird durch

```
.twoway (scatter y x) (lfit y x)
```

oder

```
.twoway scatter y x ||lfit y x eine Grafik erzeugt welche einen Scatterplott und die Regressionsgerade in einem Bild darstellt.
```

5.3.4 Spezielle Grafik Typen

Einige wenige spezielle Grafik Typen sollen hier kurz gesondert vorgestellt werden, da sie nicht über das Stata Menü zugänglich sind. Eine schöne Übersicht über Benutzer geschriebene Stata Befehle findet sich hier:

<http://www.survey-design.com.au/tipsgraphs.html>

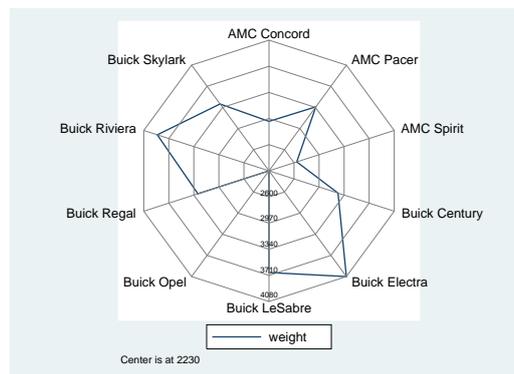
<http://www.survey-design.com.au/tipsusergraphs.html>

<https://www.stata.com/support/faqs/graphics/gph/stata-graphs/>

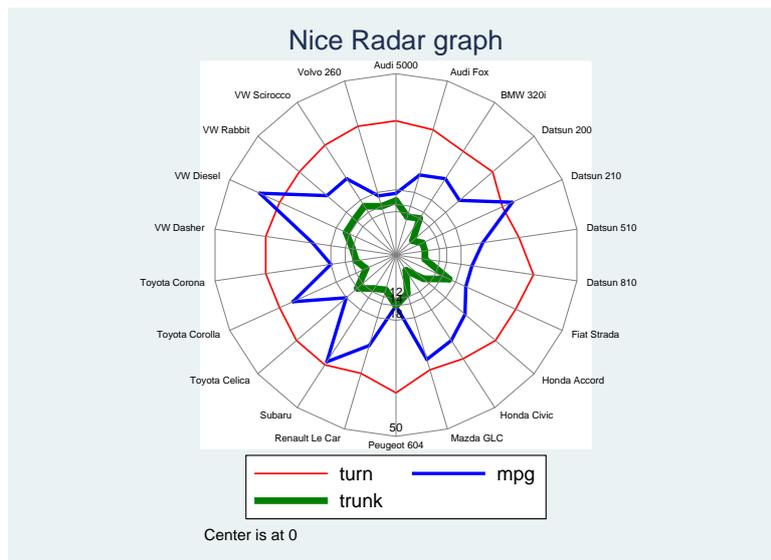
`.radar` Sogenannte Radar oder auch Spider Plots lassen sich mit `.radar` zeichnen:

```
sysuse auto
```

```
radar make weight if foreign
```

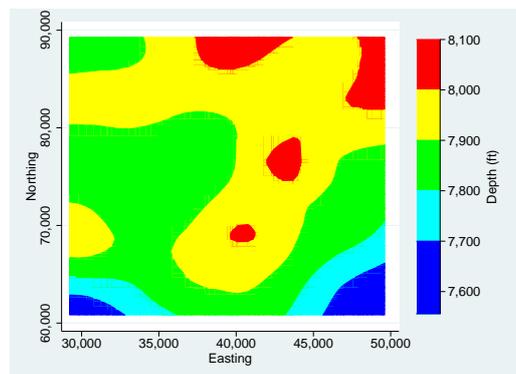


```
radar make turn mpg trunk if foreign, title(Nice Radar graph) lc(red blue green) lw(*1 *2 *4) r(0 12 14 18 50) labsz(*.5)
```



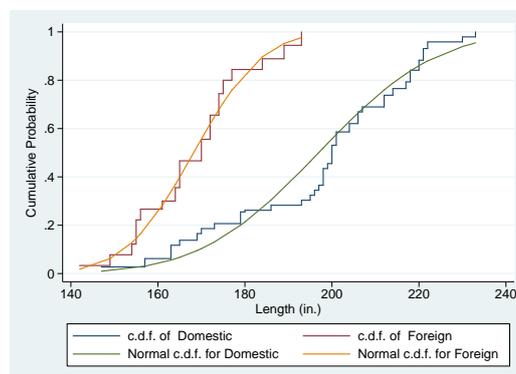
.contour

```
.sysuse sandstone
.twoway contour depth northing easting
```



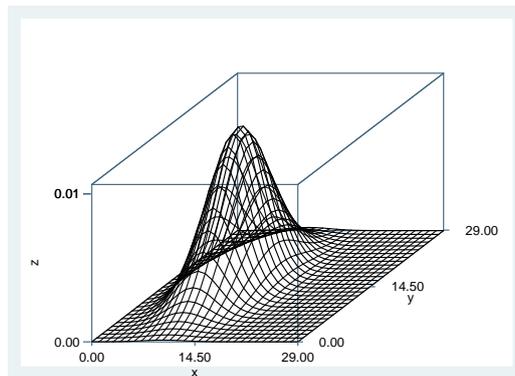
.cdfplot

```
.sysuse auto
.cdfplot length [fw=rep78], by(foreign) norm saving(mygraph,replace)
```



.surface

```
.set obs 900
.gen x = int((_n - mod(_n-1,30) - 1) / 30)
.gen y = mod(_n-1,30)
.gen z = normalden(x,10,3)*normalden(y,15,5)
.surface x y z
```

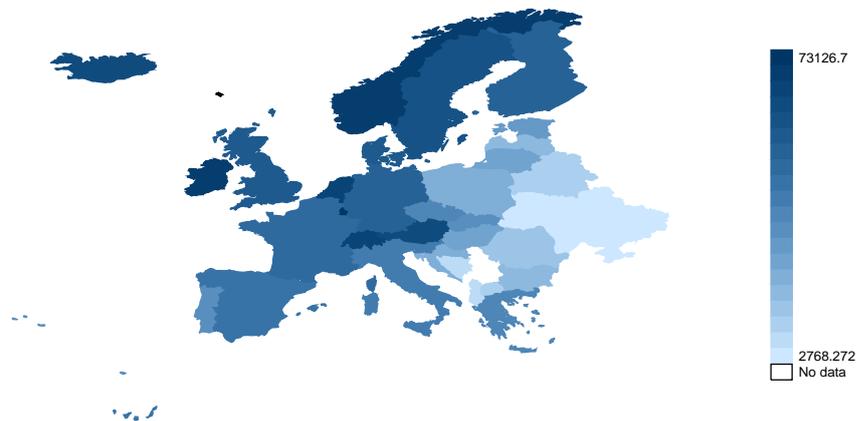


.spmap Oft will man die Situation von Regionen visualisieren. Dies funktioniert mit Stata durch den Befehl `.spmap`. Eine hilfreiche Anleitung hierzu findet sich hier:

<https://huebler.blogspot.de/2005/11/creating-maps-with-stata.html> und etwas aktueller hier:

<https://medium.com/the-stata-guide/maps-in-stata-ii-fcb574270269>

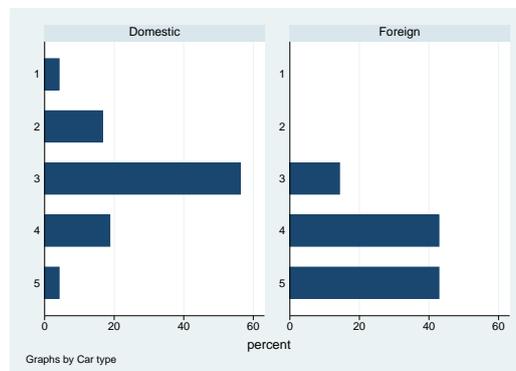
GDP per capita
average value for the time span 1995–2010



.catplot

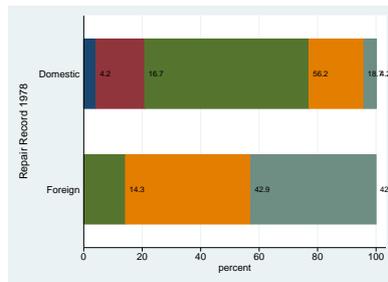
```
.sysuse auto, clear
```

```
.catplot rep78, by(foreign) percent(foreign)
```



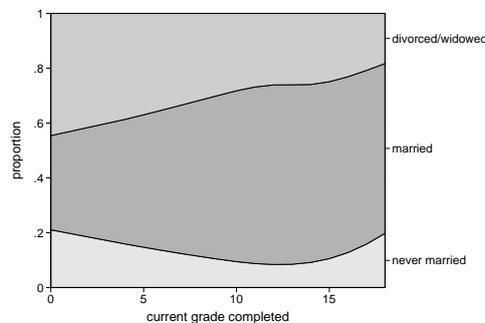
```
.sysuse auto, clear
```

```
.catplot rep78, over(for) stack asyvars perc(for) blabel(bar, format(
```



.proprcspline

```
.ssc install proprcspline
.sysuse nlsw88, clear
.gen marst = cond(never_married, 1, ///
cond(married, 2, 3)) if !missing(married, never_married)
.label define marst 1 "never married"2 "married"3 "divorced/widowed"
.label value marst marst
.proprcspline marst grade, xlab(0(5)15)
```



6 Die Erstellung von publikationswürdigen Tabellen

Stata bietet die Möglichkeit Ergebnisse jeder Art in andere Textverarbeitungsprogramme zu transformieren. Dies geschieht entweder indem gewünschte Zeilen im Output-Fenster markiert und per Copy-Paste in das entsprechende Textverarbeitungsprogramm (MS-Office, Open-Office, LaTeX) kopiert wird, oder indem spezielle Befehle eingesetzt werden. Hier nur eine Auswahl:

6.1 .outreg2

Der wohl bekannteste Befehl um Regressionen in schöne Word, Excel und LaTeX Tabellen zu konvertieren. Auch dieser muss erst durch `.ssc install outreg2` installiert werden. Die Möglichkeiten der manipulation des Grundbefehls sind groß, siehe `.help outreg2`. Die folgende Befehlsreihe erzeugt mehrere Dateien, welche in Word, Excel und LaTeX eingebunden werden können.

```
.sysuse auto,clear
.regress mpg foreign weight headroom trunk length turn displacement
.outreg2 using myfile, replace
.regress mpg foreign weight headroom
.outreg2 using myfile, see word excel tex dta
```

6.2 .estout

Sehr flexibel und mächtig sind die Befehle `estout` packages. Diese werden hier wunderbar erklärt: <http://repec.sowi.unibe.ch/stata/estout/>

6.3 .xml_tab

Dieser Befehl muss erst mit `.ssc install xml_tab` installiert werden. Mit diesem Befehl lassen sich einfach Tabellen von Stata in ein *.xml Format transformieren. Dieses kann von Excel gelesen werden. Hier einige Beispielscodes mit dem Ergebniss (in Latex):

```
.ssc install xml_tab
.sysuse auto, clear
.regress price mpg headroom trunk if foreign==0
.estimates store reg1
.regress price mpg headroom trunk turn if foreign==1
.estimates store reg2, title(Only foreign cars)
.heckman mpg weight length, sel(foreign = length displ) nolog
.estimates store heck1, title(Selection model)
.xml_tab reg1 reg2, replace
```

	reg1		Only foreign cars	
	coef	se	coef	se
Mileage (mpg)	-317.948***	107.091	-170.114*	88.874
Headroom (in.)	-783.737	565.529	356.170	991.252
Trunk space (cu. ft.)	142.072	140.221	73.566	162.119
Turn Circle (ft.)			516.143	414.700
_cons	12,752.583***	3,631.245	-9,447.506	15,631.491

note: *** p<0.01, ** p<0.05, * p<0.1

```
.xml_tab reg1 reg2, replace sheet("Table 2") stats(N r2_a) title("price regressions by car type") long tstat right
```

price regressions by car type				
	reg1		Only foreign cars	
	coef	t	coef	t
price				
Mileage (mpg)	-317.948***	-2.969	-170.114*	-1.914
Headroom (in.)	-783.737	-1.386	356.170	0.359
Trunk space (cu. ft.)	142.072	1.013	73.566	0.454
Turn Circle (ft.)			516.143	1.245
_cons	12,752.583***	3.512	-9,447.506	-0.604
Number of observations	52		22	
Adjusted R2	0.240		0.366	

note: *** p<0.01, ** p<0.05, * p<0.1

```
.xml_tab reg1 reg2, replace sheet("Table 2") stats(N r2_a) title("price regressions by car type") long pvalue below
```

price regressions by car type		
	reg1	Only foreign cars
	coef/p-value	coef/p-value
price		
Mileage (mpg)	-317.948*** (0.003)	-170.114* (0.056)
Headroom (in.)	-783.737 (0.166)	356.170 (0.719)
Trunk space (cu. ft.)	142.072 (0.311)	73.566 (0.650)
Turn Circle (ft.)		516.143 (0.213)
__cons	12,752.583*** (0.000)	-9,447.506 (0.546)
Number of observations	52	22
Adjusted R2	0.240	0.366
note: *** p<0.01, ** p<0.05, * p<0.1		

6.4 .tabout

Dieser Befehl muss erst mit `.ssc install tabout` installiert werden. Hier nur ein Beispielscodes mit dem Ergebniss (in Latex):

```
.sysuse auto
.tabout foreign using table1.txt, c(mean mpg min mpg max mpg) sum replace style(tex)
```

Car type	Mean mpg	Min mpg	Max mpg
Domestic	19.8	12.0	34.0
Foreign	24.8	14.0	41.0
Total	21.3	12.0	41.0

```
.sysuse voter, clear
.tabout inc candidat using table2.txt, c(mean pfrac) f(1) clab(%) sum replace style(tex) bt font(bold) cl1(2-5)
topstr(12cm) botstr(voter.dta)
```

Family Income	Candidate voted for, 1992			
	Clinton	Bush	Perot	Total
	%	%	%	%
<\$15k	8.3	3.2	2.5	4.7
\$15-30k	10.8	8.4	4.8	8.0
\$30-50k	12.3	11.4	6.3	10.0
\$50-75k	8.0	8.4	3.6	6.7
\$75k+	4.7	6.2	2.1	4.3
Total	8.8	7.5	3.9	6.7

6.5 outtex

Eine weitere Möglichkeit Regressionstabellen für LaTeX zu erstellen bietet der von Benutzer geschriebene Befehl `.outtex`.

6.6 .regsave

Eine flexible, einfache und äußerst effiziente Möglichkeit ist das abspeichern von Ergebnissen mittels `.regsave`. Hier werden alle ausgewählten Ergebnisse in eine `*.dta`-File gespeichert. Mit der Option `detail(all)` werden alle Informationen der Regression gespeichert. Diese `*.dta`-File kann dann weiter manuell in Stata bearbeitet werden. Selbstverständlich besteht die Möglichkeit `*.dta` files in andere Dateiformate zu konvertieren, siehe Menü→File→Export. `.texsave` etwa bietet eine komfortable Weise der Konvertierung in eine LaTeX oder Scientific Word Tabelle. Durch `.outsheet` kann ein `*.dta`-File in eine Excel Tabelle transformiert werden.

Das schöne an `.regsave` ist, dass man sich a priori noch nicht über die in einer Tabelle enthaltenen Regressionen im Klaren sein muss, da die einzelnen Ergebnisse auch erst später zusammengeführt werden können ohne die Regressionen nochmals durchlaufen lassen zu müssen, wie bspw. bei `.outreg2`.

Hier nun ein einfaches und ein ausgefeilteres Beispiel wie `.regsave` arbeitet. Versuchen sie die Unterschiede zu verstehen.

```
sysuse auto,clear
regress price mpg trunk headroom length
regsave using results1, table(reg1, parentheses(stderr) asterisk(10 5)) replace
regress price mpg trunk headroom length if foreign==1
regsave using results1, table(reg2, parentheses(stderr) asterisk(10 5)) append
use results1, clear
list
```

var	reg1	reg2
mpg_coef	-173.9506531*	-55.64738083
mpg_stderr	(87.75118256)	(66.96059418)
trunk_coef	80.90940094	130.5377197
trunk_stderr	(119.8175812)	(111.6765823)
headroom_coef	-680.289856	454.9372253
headroom_stderr	(486.0063477)	(719.7601929)
length_coef	23.28607941	135.2952271**
length_stderr	(27.02926254)	(31.79295158)
_cons_coef	6416.948242	-17718.53516**
_cons_stderr	(6036.935059)	(6839.852539)
N	74	22
r2	.2503406703	.7287294865

```
sysuse auto,clear
regress price mpg trunk headroom length
regsave using results1, pval nose table(reg1, order(regvars r2 ) format(%5.1f) asterisk(10 5)) replace
regress price mpg trunk headroom length if foreign==1
regsave using results2, pval nose table(reg2, order(regvars r2 ) format(%5.1f) asterisk(10 5)) replace
use results1, clear
merge 1:1 var using results2.dta, nogenerate
replace var = "p-valueif strpos(var,"_pval")!=0
replace var = substr(var,"_coef",,,)
gen lfd=_n
replace lfd=100 if var=="N"
replace lfd=101 if var=="r2"
replace var=Öbservationsif var=="N"
replace var="R-squaredif var=="r2"
sort lfd
drop lfd
list
```

var	reg1	reg2
_cons	6416.9	-17718.5**
p-value	0.3	0.0
headroom	-680.3	454.9
p-value	0.2	0.5
length	23.3	135.3**
p-value	0.4	0.0
mpg	-174.0*	-55.6
p-value	0.1	0.4
trunk	80.9	130.5
p-value	0.5	0.3
Observations	74	22
R-squared	0.3	0.7

6.7 .texsave

Wer LaTeX verwendet wird mit dem Befehl `.texsave` seine Freude haben. `.texsave` transportiert den existierenden Datensatz in eine Tabelle.

6.8 .putexcel

In Stata 13 findet man den Befehl `.putexcel` dieser transportiert Stata Output in eine Excel-Datei.

7 Sonstige nützliche Befehle

- `.capture` Stellt man `.capture` einem beliebigen Stata Befehl voran, so wird im Falle einer Fehlermeldung der Befehl einfach übersprungen.
- `.duplicates` Unter `.help duplicates` bietet Stata eine Reihe von Befehlen um die identifizierenden Variablen eines Datensatzes ausfindig zu machen und eventuell doppelte Werte zu beseitigen.
- `.separate` Generiert separierte Variablen. Bspw. bildet `.separate x, by(y)` eine neue Variablen für jede Kategorie der „by-Variable“.
- `.preserverestore` Mit `.preserve` wird der aktuelle Zustand des Datensatzes gespeichert. Dieser Zustand kann dann mit `.restore` jederzeit wieder hergestellt werden. Diese Befehlskombination wird gerne in Schleifen verwendet. Es ist jedoch zu beachten, dass bei einem Abbruch aufgrund eines Syntaxfehlers oder Ähnlichem automatisch und ohne das dies angezeigt wird zu dem Zustand vor dem `.preserve` Befehl zurückgekehrt wird.
- `.quietly` Bei längeren Programmen möchte man bestimmte Teile des Outputs nicht anzeigen lassen. Dies kann mit der Voranstellung der Anweisung `.quietly` an einen Befehl erreicht werden.
- `.winexec` **und** `.shell` Mit diesen Befehl(en) kann man von Stata aus Windows Befehle ausführen.
- `.distinct` **und** `.unique` Wieviele einzigartige Ausprägungen hat eine String-Variable? Dies beiden Befehle helfen bei der Antwort. Sie wurden von Benutzern geschrieben und lassen sich, wie in Unterabschnitt 2.6 beschrieben, einfach mit `.ssc install distinct` und `.ssc install unique` installieren.

8 Übungsaufgaben

8.1 Erste Schritte

- 1) Lernen Sie sich selbst zu helfen. Hierzu folgende Tipps:
 - a) Lesen Sie das Skript.
 - b) Beachten Sie die Literaturempfehlungen auf S. 7.
 - c) Besuchen Sie folgende Internetseiten und machen Sie sich mit dem Inhalt dieser Seiten vertraut:
www.ats.ucla.edu/stat/stata/
<http://www.ats.ucla.edu/stat/stata/webbooks/reg/>
<http://www.stata.com/bookstore/books-on-stata/>
 - d) Machen Sie sich mit den Hilfsfunktionen und der PDF Dokumentation von Stata vertraut.
- 2) Richten Sie Ihren PC für den effizienten Gebrauch von Stata ein. Führen Sie folgende Schritte durch:
 - a) Legen Sie auf Ihrer Festplatte (persönlichen Laufwerk G:\) einen Ordner 'stata' an, der folgende Unterordner enthält: orig (für Rohdaten); data (für modifizierte Datensätze); dofiles (für Do-Files); logfiles (für Log-Files); graphs (für Grafiken); tabs (für Tabellen), ado (für ado-files). Laden Sie alle erforderlichen Daten hier herunter: <https://t1p.de/stata-files> (Passwort:happystata) und speichern Sie diese entsprechend.
 - b) Sollten Sie von Benutzer geschriebene Befehle (sogenannte *.ado-files) verwenden, so teilen Sie Stata mit `.adopath + G:/stata/ado` bitte mit, wo sich dieser Pfad befindet. Erklärungen zur Installation von ado-files hierzu finden sich auch in Unterabschnitt 2.6.
- 3) Öffnen Sie den 'auto' Datensatz durch Eingabe von `.sysuse auto, clear`. Lassen Sie sich die Daten durch Eingabe von `.browse` anzeigen. Warum erscheinen die Angaben in verschiedenen Farben. Erklären Sie, worin sich String-Variablen, Numerische-Variablen und gelabelte Numerische-Variablen unterscheiden. Machen Sie aus der Variable „make“ eine gelabelte Numerische-Variable und aus „foreign“ eine String-Variable. (Lösung: `.encode make, generate(make_n)` und `.decode foreign, generate(foreign_s)`).
- 4) Öffnen Sie Stata und den darin implementierten Do-file Editor. (Hinweis: Dies lässt sich durch die Eingabe von `.doedit` in der Eingabeaufforderung oder durch das Tastenkürzel 'Strg+9' erreichen.) Übertragen Sie die unten angeführte do-file in den Editor. Speichern Sie diese ab und machen sie diese 'lauffähig'. Das bedeutet, mit 'Strg+D' oder durch klicken des 'Execute (Do)' Icons im Editor kann man die Befehle des do-files ausführen.

```
/* Dies ist ein Beispiel
fuer ein do-file*/

* das Outputfenster laeuft durch
set more off
* setzen der working directory
cd C:/stata

/**** Hier folgt die Analyse: ****/
* Daten laden:
sysuse auto, clear
* Datensatz beschreiben:
describe
* Zusammenhang von Preis und Gewicht veranschaulichen:
scatter price weight
* Bild abspeichern
graph export graphs/bild.png
* Deskriptive Statistik:
summarize price weight
```

```
* Regression: beeinflusst das Gewicht den Preis
reg price weight
* abspeichern der Datei
save data/Musterbeispiel.dta, replace
exit
```

- 5) Wissen Sie, wie man aus dem Do-file Editor heraus nur einen Befehle bzw. nur einen Ausschnitt an Befehlen ausführt?
- 6) Um mit Daten sinnvoll arbeiten zu können, muss man wissen, welche Struktur die Daten aufweisen (Querschnitt, Zeitreihe, Panel) und durch welche Variablen die Beobachtungen des Datensatzes eindeutig identifiziert werden.
- a) Der häufig verwendete ‘auto’ Datensatz (`.sysuse auto, clear`) beispielsweise ist ein Querschnitt-datensatz, welcher durch die Variable ‘make’ identifiziert wird. Das bedeutet, jede Zeile stellt eine neue Beobachtung darstellt. Durch Eingabe von `.duplicates report make` kann überprüft werden, ob die Variable ‘make’ tatsächlich nur unterschiedliche Autos beinhaltet.
- b) Finden Sie heraus, durch welche Variablen folgende Datensätze identifiziert werden: cancer, voter, gnp96, educ99gdp, und xtline1.

(Hinweis: Diese Datensätze lassen sich mit `.sysuse` aufrufen. Mit dem Befehl `.duplicates` kann man herausfinden, ob eine Variable oder eine Kombination mehrerer Variablen die im Datensatz enthaltenen Beobachtungen identifizieren.)

8.2 Daten einlesen (daten_einlesen.do)

a) Speichern Sie folgende Lösungen in einer do-file mit den Namen „daten_einlesen“ im Ordner „dofiles“. Vergessen Sie nicht, als Arbeitsverzeichnis den „Stata“ Ordner festzulegen.

b) Erstellen Sie folgenden Datensatz in Stata:

einsbis3	vierbis6	siebenbis9
1	4	7
2	5	8
3	6	9

c) Erstellen Sie diesen Datensatz in Excel und speichern Sie diesen als .csv und als .xls-Datei im Ordner „orig“ ab.

d) Importieren Sie diese beiden Dateien in Stata.

e) Inspizieren Sie den Datensatz mit den gängigsten Befehlen.

f) Speichern Sie ihren Datensatz im Ordner „data“ unter dem Titel einbis9.

g) Verändern Sie alle den Datensatz derart, dass er folgendes beinhaltet:

einsbis3	vierbis6	siebenbis9
1	2	7
1	2	8
1	2	9

h) Verändern Sie die Variablennamen derart, dass folgendes angezeigt wird:

nur1	nur2	siebenbis9
1	2	7
1	2	8
1	2	9

i) Inspizieren sie die Daten deskriptiv. (Hinweis: Betrachten sie die Hilfe zu folgenden Befehlen: `.summarize` `.describe` `.codebook` `.inspect`)

j) „labeln“ Sie nun die Variablen derart, dass man aufgrund der Beschreibung erkennen kann was in der jeweiligen Variable für Werte enthalten sind.

8.3 Sicherheitsgurte und Verkehrstote (gurte.do)

In dieser Aufgabe sollen Sie untersuchen, wie die Benützung eines Sicherheitsgurtes im Auto die Zahl der Verkehrstoten beeinflusst.

1) Laden Sie die Dateien ‘main_rhs.dta’ und ‘fatality.xlsx’ herunter (<https://t1p.de/stata-files> — Passwort: happystata) und speichern Sie diese in Ihrem Arbeitsverzeichnis ab.

2) Laden Sie den Datensatz ‘main_rhs.dta’ in Stata und labeln Sie die bisher ungelabelten Variablen.

(Hinweis: Die Variable ‘x1’ enthält Beobachtungen bzgl. der Benutzungsrate des Sicherheitsgurtes und ‘x2’ enthält Beobachtungen bzgl. des Einkommens pro Kopf.)

3) Speichern Sie den Datensatz in einer neuen Datei unter dem Titel ‘fatality_ready.dta’ ab.

4) Importieren Sie den Datensatz ‘fatality.csv’ in Stata.

(Hinweis: Sollte Ihnen das nicht gelingen können Sie mit der Datei ‘gurt.dta’ fortfahren.)

5) Bringen Sie den Datensatz in das ‘Long’-Format.

(Hinweis: Sollte Ihnen das nicht gelingen können Sie mit der Datei fatality_processed.dta fortfahren.)

6) Speichern Sie den Datensatz in einer neuen Datei unter dem Titel ‘fatality_long.dta’ ab.

7) Kombinieren Sie nun die beide Datensätze ‘fatality_long.dta’ und ‘fatality_ready.dta’. Löschen Sie nicht gematchten Beobachtungen.

8) Speichern Sie den kombinierten Datensatz unter dem Titel ‘ready.dta’ ab

9) Berechnen Sie die Korelation der Variablen ‘fatality’ und ‘sblt_use’.

10) Erstellen Sie eine Variable die den Logarithmus vom Pro-Kopf Einkommen beinhaltet.

11) Erstellen Sie eine Dummy die Eins ist, wenn ein Staat ein Pro-Kopf Einkommen unter 10.000 USD hat und ansonsten Null ist.

12) Zeichnen Sie ein Punktediagramm welches die Beziehung von ‘fatality’ und ‘sblt_use’ aufzeigt. Fügen Sie sowohl die Abkürzungen der Staaten, als auch eine Regressionsgerade mit in das Bild ein.

13) Untersuchen Sie den Einfluss von ‘sblt_use’ auf ‘fatality’ anhand einer Regression.

14) Fügen Sie nun der Regression folgende Kontrollvariablen hinzu: ‘speedlim65’, ‘speedlim70’, ‘alcage21’, ‘bloodalc08’, ‘lnincpc’ und ‘age’.

15) Schätzen Sie diese Gleichung nun mit robusten Standardfehlern.

16) Nehmen Sie nun sogenannte fixe Effekte für die Staaten mit in das geschätzte Modell mit auf. Sprich: ein Dummy für jeden Staat.

17) Nehmen Sie nun sogenannte fixe Effekte für die Jahre mit in das geschätzte Modell mit auf. Sprich: ein Dummy für jedes Jahr.

18) Interpretieren Sie Ihre Schätzergebnisse kurz.

8.4 Bundestagswahl 2017: Die Partei und die AFD (wahl.do)

In den folgenden Aufgaben untersuchen Sie die Ergebnisse der Bundestagswahl 2017.

- 1) Laden Sie die Dateien 'struktur.dta' und 'kreise.dta' herunter (<https://t1p.de/stata-files> — Passwort: happystata) und speichern Sie diese in Ihrem Arbeitsverzeichnis ab. Öffnen Sie die Datei 'kreise.dta' in Stata und labeln Sie nicht gelabelten Variablen.
- 2) Durch welche Variablen werden die Daten eindeutig identifiziert.
- 3) Wie viele verschiedene Wahlkreise gibt es im Jahr 2017?
- 4) Wie heißt der Wahlkreis in dem die Partei 'DiePartei' die meisten Zweitstimmen erhalten hat?
- 5) Erzeugen Sie Variablen die anzeigen, wie viel Prozent aller gültigen Zweitstimmen die einzelnen Parteien in den Wahlkreisen erzielen konnten.
- 6) Wie heißt der Wahlkreis in dem die Partei 'DiePartei' den größten Anteil der abgegebenen Zweitstimmen erhalten hat?
- 7) Zeichnen Sie ein Kuchendiagramm welches das Wahlergebnis der Zweitstimmen 2017 veranschaulicht.
- 8) Zeichnen Sie ein Balkendiagramm welches das Wahlergebnis der Zweitstimmen 2017 veranschaulicht.
- 9) Sie sollen nun untersuchen, ob sozio-ökonomische Gründe die Wahlergebnisse erklären können. Mergen Sie hierzu 'kreise.dta' mit 'struktur.dta'. Löschen Sie *not matched* Beobachtungen und untersuchen im folgenden ausschließlich Beobachtungen des Wahlergebnisses von 2017.

(Hinweis: Sollten Sie bis hierher Probleme bei Lösung der Aufgaben aufgetreten sein, bitte ich Sie mit dem Datensatz 'merge.dta' fortzufahren. Dieser ist hier erhältlich: <https://t1p.de/stata-files> — Passwort: happystata)

- 10) Sie vermuten, das die Bevölkerungsdichte das relativ gute Wahlergebnis der AFD (Zweitstimmen in %) erklärt. Untersuchen Sie dies anhand eines Scatterplots. Zeichnen Sie in den Scatterplot auch eine Regressionsgerade mit ein. Regressieren Sie das Wahlergebnis der AFD (Zweitstimmen in %) auf die Bevölkerungsdichte.
- 11) Logarithmieren Sie die Bevölkerungsdichte und erstellen Sie hiermit einen Scatterplot mit Regressionsgerade der den Zusammenhang mit dem Wahlergebnisses der AFD (Zweitstimmen in %) darstellt. Regressieren das Wahlergebnis der AFD (Zweitstimmen in %) auf die logarithmierte Bevölkerungsdichte. Argumentieren Sie, ob diese Schätzspezifikation der vorherigen vorzuziehen ist, oder nicht.
- 12) Sie vermuten, dass die AFD in den neuen Länder besonders erfolgreich ist und zugleich diese Länder weniger dicht besiedelt sind. Überlegen Sie, wie man für diese potentielle Verzerrung begegnen kann und schätzen Sie obigen Zusammenhang noch einmal.
- 13) Des Weiteren vermuten Sie, dass die AFD vermehrt von alten Menschen gewählt wird. Fügen Sie der Schätzung die Variablen age_6074 und age_75 hinzu.
- 14) Fügen Sie der Schätzung noch die Variablen mig, gdppc, educ, child, hartz und unemp hinzu. Interpretieren Sie das Ergebnis kurz.
- 15) Die CSU hat Wahlkampf gegen die AFD gemacht. Erstellen Sie eine Dummy Variable die Eins ist für alle Wahlkreise in Bayern und ansonsten Null. Fügen Sie diese Variable der Schätzung hinzu. Interpretieren Sie kurz.
- 16) Die AFD ist in Landesverbänden organisiert. Sie vermuten, dass diese unterschiedlich gut arbeiten. Fügen Sie einen eigenen Achsenabschnitt für jedes Bundesland in die Regression ein.
- 17) Schätzen Sie nun separat für die Neuen Länder und für die Alten Länder. Interpretieren Sie die beiden Schätzergebnisse kurz.

8.5 Bundestagswahl 2017: (wahl_sozial.do)

In den folgenden Aufgaben untersuchen Sie die Ergebnisse der Bundestagswahl 2017.

- 1) Laden Sie die Dateien 'struktur.dta' und 'kreise.dta' herunter (<https://t1p.de/stata-files> — Passwort: happystata) und speichern Sie diese in Ihrem Arbeitsverzeichnis ab.
- 2) Öffnen Sie die Datei 'kreise.dta' in Stata und labeln Sie nicht gelabelten Variablen und entfernen Sie Variablen, die offensichtlich keine über die Beobachtungen variierenden Informationen beinhalten.
- 3) Durch welche Variablen werden die Daten eindeutig identifiziert.
- 4) Sie interessieren sich ausschließlich für die Wahl 2017. Löschen Sie alle nicht benötigten Daten und speichern Sie diesen Datensatz unter 'kreise_clean.dta' ab.
- 5) Sie interessieren sich ausschließlich für die aktuell im Bundestag vertretenen Parteien (CDU, CSU, SPD, AFD, FDP, Die Linken, Die Grünen). Löschen Sie Variablen, die Wahlergebnisse anderer Parteien beinhalten.
- 6) Nennen Sie die Anzahl der im Datensatz enthaltenen Wahlkreise.
- 7) Nennen Sie, in welchem Wahlkreis die sieben Parteien jeweils ihr bestes Ergebnis in Bezug auf den prozentualen Anteil der erhaltenen Zweitstimmen erzielen konnten.
- 8) Zeigen Sie an, wie viel Prozent Zweitstimmen die sieben Parteien bundesweit jeweils auf sich vereint haben. Dies kann entweder mit einer Tabelle oder einer Grafik geschehen.

Im den folgenden Aufgaben sollen Sie untersuchen, ob die Anzahl der ungültigen Zweitstimmen durch sozio-ökonomische Faktoren erklärt werden kann. Sie vermuten, dass es zwei Gründe gibt warum ein Wahlschein als ungültig gewertet werden muss. Erstens, die Wähler füllen den Wahlschein unbeabsichtigt falsch aus weil Sie unfähig sind die Instruktionen korrekt umzusetzen. Zweitens, die Wähler machen den Wahlschein absichtlich ungültig, um damit ihren Protest mit den zur Wahl stehenden Parteien Ausdruck zu verleihen. Natürlich schließen sich beide Gründe nicht gegenseitig aus. Die folgenden Aufgaben sollen diese Vermutungen untersuchen.

- 9) Laden Sie nun den Datensatz 'struktur.dta' und mergen Sie diesen mit 'kreise_clean.dta'. Löschen Sie alle Beobachtungen, die nicht miteinander verbunden werden konnten.

(Hinweis: Sollten Sie bis hierher Probleme bei Lösung der Aufgaben aufgetreten sein, bitte ich Sie mit dem Datensatz 'mergeSS18.dta' fortzufahren. Dieser ist online erhältlich (<https://t1p.de/stata-files> — Passwort: happystata).)

- 10) Erstellen Sie eine Variable 'sh_ung2'. Diese soll den Anteil der ungültigen Zweitstimmen an allen abgegebenen Stimmen enthalten.
- 11) Zeigen Sie, wie die Variablen 'sh_ung2', 'educ', 'unemp', 'density', 'mig' und 'gdppc' korrelieren.
- 12) Regressieren Sie die Variable 'sh_ung2' auf die Variablen 'educ', 'unemp', 'density', 'mig' und 'gdppc'. Interpretieren Sie die Ergebnisse kurz in Bezug auf die oben genannten Vermutungen.
- 13) Sie wissen, dass die Variablen 'educ' und 'unemp' stark miteinander korrelieren. Schätzen Sie deswegen oben genannte Schätzgleichung ohne 'educ' und ein Weiteres mal ohne 'unemp' aber mit 'educ'. Welche dieser beiden Variablen erscheint nun statistisch signifikant zu sein?
- 14) Da die unabhängigen Variablen unterschiedliche Maßeinheiten besitzen, lassen sich die Regressionskoeffizienten schwer vergleichen. Man berechnet daher zusätzlich sogenannte standardisierte Regressionskoeffizienten. Wie kann man diese standardisierten Regressionskoeffizienten im Regressionsoutput von Stata mit ausweisen lassen?

8.6 Wohnfläche (wohnen.do)

- a) Kohler and Kreuter (2016) stellt eine Reihe von Datensätzen zur Verfügung. Installieren Sie diese in dem Ordner „data“. Geben Sie hierzu `.net from http://www.stata-press.com/data/kkd/` in Stata ein. Alternativ können sie die Daten hier runterladen:
`http://www.stata-press.com/data/kkd.html`
- b) Laden Sie den Datensatz „data1.dta“. Der Datensatz enthält Informationen über die Wohnungsgröße und das Haushaltseinkommen der befragten Personen. Es wird vermutet, dass die Wohnfläche der Haushalte vom Haushaltseinkommen abhängt. Erstellen und interpretieren Sie einen Scatterplot als auch eine einfache Regression um diesen Zusammenhan zu untersuchen.
- c) Der Output der Regression in Stata besteht aus 3 Teilen: Dem Koeffizientenblock (unten), dem Anova-Block (links) und dem Modellfit-Block (rechts).
- d) Das einfache Regressionsmodell ist in diesem Falle problematisch. Welche Probleme können auftreten?
- e) Neben dem oben angesprochenen Zusammenhang erwarten wir, dass die Wohnflächen sich in Ost und West tendenziell unterscheiden, dass es Unterschiede für Mieter und Eigentümer gibt, und dass größere Familien eher in größeren Wohnungen/Häusern leben. Schätzen und interpretieren Sie dieses multiple Regressionsmodell.
- f) Schätzen Sie das multiple Regressionsmodell mit unterschiedlichen Variablenkombinationen und stellen Sie die Ergebnisse in einer Tabelle dar. Versuchen Sie in dieser Tabelle auch Informationen zu der Anzahl der Beobachtungen, verschiedenster Selektionskriterien und der Signifikanz der Koeffizienten darzustellen.
- g) Versuchen sie die Regressionsergebnisse in ein publikationswürdiges Format (Word, Excel, LaTeX) zu exportieren.

8.7 Poser Autos (posercar.do)

Öffnen Sie folgenden Datensatz:

```
.sysuse auto.dta, clear
```

- a) Stellen Sie den Zusammenhang von Preis und Gewicht eines Autos in einem Scatterplot dar. Betiteln Sie die Grafik sinnvoll und versuchen Sie, dass man erkennt um welches Auto sich bei der jeweiligen Beobachtung handelt erkennt.
- b) Speichern Sie diese Grafik im Stata Format .gph und in den Formaten .png und .pdf ab.
- c) Erstellen sie eine Variable „lp100km“ die anzeigt wie hoch der Verbrauch eines durchschnittlichen Autos in Litern pro 100 Kilometern ist. (Hinweis: Eine Gallone entspricht etwa 3,8 Litern und eine Meile etwa 1,6 Kilometern.)
- d) Erstellen sie eine Dummy Variable „larger6000“ die 1 ist, wenn der Preis eines Autos über 6000\$ liegt.
- e) Suchen Sie nun das „unvernünftigste Poser Auto“ das höchstens 6000\$ kostet. Ein unvernünftiges Auto wird definiert als ein Auto, welches teuer ist, einen großen Wendekreis besitzt, viel Kraftstoff verbraucht und oft defekt ist (rep78 klein ist). Bilden Sie hierfür zu jeder entsprechenden Variable einen metrischen Indikator der bei dem Auto den Wert 1 anzeigt welches in dieser Variable am unvernünftigsten ist und beim Wert 0 das vernünftigste Auto ist. Alle anderen Autos sollen zwischen 0 und 1 rangieren.
- f) Suchen Sie nun das beste „Poser“ Auto unabhängig vom Preis des Autos. Für Ambitionierte: versuchen Sie hierzu das Procedere der vorherigen Aufgabe als Programm zu schreiben.

8.8 Income and Exams (incomeandexams.do)

Lesen Sie diese Daten

<http://fmwww.bc.edu/ec-p/data/stockwatson/caschool.dta>

in Stata ein und machen Sie sich damit vertraut. Die Datenbeschreibung finden Sie hier:

<http://fmwww.bc.edu/ec-p/data/stockwatson/caschool.des>

- a) Calculate total expenditures by district and by county.
- b) Show the correlations between computers per student, expenditures per student, teachers per students, district average income, the percentage of English learners and average test scores. From the correlations do you expect that outcomes in test scores per district are similar or deviate a lot? Why do you think so?
- c) Plot the average test score against average income per district. Do you expect a linear relationship between the two variables?
- d) Regress average test scores on average income and separately on average income and average income squared. What do the results tell you about the relationship between average income and test scores?
- e) Develop a regression where you analyze the same relationship by only comparing within county differences (hint: fixed effects). Why do the results differ?
- f) Include other independent variables in your regression, like the computer-student ratio, the teacher-student ratio and percentage of English learners. (Use the basic regression without fixed-effects). Do the regression results confirm your expectations? Why (not)?

8.9 Beauty and teaching evaluation score (beauty.do)

Lesen Sie diese Daten

<http://www.princeton.edu/~mwatson/sw/TeachingRatings.xls>

in Stata ein und machen Sie sich damit vertraut. Die Datenbeschreibung finden Sie hier:

http://www.princeton.edu/~mwatson/Stock-Watson_3u/Students/EE_Datasets/TeachingRatings_Description.pdf

Zur Information: This question is based on the dataset ‘TeachingRatings’, which contains data on course evaluations, course characteristics and instructor characteristics for 463 courses give at the University of Texas. One of the characteristics is an index of the instructor’s “beauty” as rated by a panel of 6 judges. In this exercise, you will investigate how course evaluations are related to the professors’ beauty. Helpful commands are: rename, replace, scatter, reg, correlate, generate, log.

- a) Erstellen Sie im Kopf des do-Files eine kurze, aussagekräftige Beschreibung des Files und fügen Sie die üblichen Eingangsbefehle ein, die die Lauffähigkeit des Files sicherstellen.
- b) Lesen Sie die folgenden Daten
<http://www.princeton.edu/~mwatson/sw/TeachingRatings.dta>
in Stata ein und machen Sie sich damit vertraut.
Eine Beschreibung der Daten finden Sie hier:
http://www.princeton.edu/~mwatson/sw/TeachingRatings_Description.pdf.
- c) Benennen Sie die Variable *course_eval* um in *score* und die Variable *nnenglish* in *native*. Verändern Sie die Variable *native* dahingehend, dass sie den Wert 1 für einheimische Professoren annimmt und ansonsten 0 ist.
- d) Stellen Sie die Beziehung zwischen *score* and *beauty* in einem Scatterplot dar und legen Sie eine gefittete Gerade durch die Daten. Welcher Effekt lässt sich aus dem Scatterplot erwarten?
- e) Wie groß ist die Korrelation zwischen beiden Variablen?
- f) Regressieren Sie die Variable *score* auf *beauty* mithilfe einer linearen Einfachregression. Ist der Effekt von *beauty* statistisch signifikant?
- g) Fügen Sie die anderen Variablen als Kontrollen in Ihre Regression ein. Wie verändert dies den Zusammenhang zwischen *beauty* und *score*?
- h) Unterscheidet sich der Effekt von *beauty* zwischen den Geschlechtern? Schätzen Sie zur Beantwortung der Frage die Gleichung getrennt für Männer und Frauen, einmal ohne und einmal mithilfe einer Schleife.

8.10 Lehrer-Schüler Verhältnis (school.do)

Lesen Sie diese Daten <http://fmwww.bc.edu/ec-p/data/stockwatson/caschool.dta>
in Stata ein und machen Sie sich damit vertraut. Die Datenbeschreibung finden Sie hier:
<http://fmwww.bc.edu/ec-p/data/stockwatson/caschool.des>

Variablenbeschreibung:

str Klassengröße, d. h. das Verhältnis Schuler/Lehrer (student-teacher ratio)
testscr Mittleres Ergebnis für die Tests „Lesen“ und „Mathematik“ (test score)
read_scr Mittleres Ergebnis für den Test „Lesen“ (reading score)
math_scr Mittleres Ergebnis für den Test „Mathematik“ (math score)
enrl_tot Anzahl der Schuler im Schulbezirk (total enrollment)
teachers Anzahl der Lehrer, anteilig für Teilzeit (number of teachers)
computers Anzahl der Computer (number of computers)
expn_stu Budget pro Schuler, in \$ (expenditures per student)
el_pct Anteil der nicht-englischsprachigen Schuler (percent of English learners)
meal_pct Anteil der Schuler mit Anspruch auf verbilligtes Mittagessen (percent qualifying for reduced-price lunch)
calw_pct Anteil der Schuler mit Anspruch auf das CalWorks-Programm (percent qualifying for CalWorks)
avginc Durchschnittseinkommen im Schulbezirk, in 1000\$ (district average income)
dist_code Schlüsselnummer des Schulbezirks (district code)

- Machen Sie sich mit dem Datensatz „caschool“ vertraut. Erstellen Sie Überblicksstatistiken für die Testresultate und den Schüler/Lehrer-Schnitt. Errechnen Sie auch einige wichtige Quantile und die Korrelation der Variablen. Zeichnen Sie einen anschaulichen Scatterplot.
- Schätzen Sie eine lineare Einfachregression von *testscr* auf *str*. Interpretieren Sie das Resultat. Zeichnen Sie den Zusammenhang in der Grafik aus Aufgabe 1 ein.
- Fügen Sie ihrem Modell nun auch die Variablen *el_pct* und *expn_stu* hinzu. Sind diese signifikant? Wie ändert sich die Interpretation? Was sagt das Ergebnis über die Effizienz der Schulpolitik aus?
- Regressieren Sie die Testergebnisse auf das mittlere Einkommen. Zuerst linear, dann quadratisch, dann logarithmisch. Stellen Sie den Zusammenhang auch grafisch dar. Interpretieren Sie die Koeffizienten.
- Regressieren Sie die Testergebnisse auf (1) *str*, *el_pct* und *meal_pct*, (2) zusätzlich *avginc* (in logs), und (3) zusätzlich str^2 und str^3 . Stellen Sie die Ergebnisse tabellarisch dar und interpretieren Sie sie. Testen Sie, ob der nichtlineare Zusammenhang von STR in (3) signifikant ist. Zeichnen Sie den geschätzten Zusammenhang aus (3) im Scatterplot der Aufgabe 1 ein.

8.11 Makros und Co (makros.do)

a) Lassen Sie folgendes in Stata (und ihrem Kopf) "durchlaufen":

```
use data/data1, clear
* Variablenlisten
drop ymove ybuild
summarize income
summarize kitchen-phone
summarize np*

* Optionen
summarize income, detail
tabulate gender np9506, missing row

* Operatoren
display 2+3
display 2^3
display 2*(3+5)/(5-2)^3
display 2==3
display 3==3
display 2==3 & 2==2
display 2==3 | 2==2
display ("Regensburg" == "Princeton")

* Funktionen
display sqrt(2)
display exp(5+abs(-1))

* Die in-Bedingung
list persnr gender ybirth in 10
list persnr gender ybirth in 10/12
list persnr gender ybirth in -5/-1

* Die if-Bedingung
summarize income if gender==1
summarize income if ybirth < 1979
summarize income if ybirth <= 1979
summarize income if ybirth ~= 1979
summarize ybirth if edu >=6
summarize ybirth if edu >=6 & edu<.
tabulate edu, missing nolabel

* generate und replace
generate minor=0
replace minor = 1 if ybirth >1979 & ybirth <.
sum sqm, detail
generate size=0
replace size = 4 if sqm <.
replace size = 3 if sqm <= 113
replace size = 2 if sqm <= 81
replace size = 1 if sqm <= 57

* _n und _N
sort persnr
gen index = _n
by intrnr, sort: generate intcount = _N
list persnr intrnr intcount in 1/10
display _N

* recode
use data/data1, clear
recode ybirth (min/1979 = 0)(1980/max=1), gen(minor)
recode kitchen-phone (1=0)(2=1), gen(d1 d2 d3 d4 d5 d6 d7 d8)
```

```

* egen
egen incomemean = mean(income), by(state)
egen rowmiss = rowmiss(income state voc edu)
use data/hierarch.dta, clear
egen hhinc = total(eink), by(hhnr)

* string-Funktionen
use data/mdb, clear
d
encode party, gen(party_n)
destring period, gen(period_n)
gen cdu = 1 if party=="CDU"
gen comma = strpos(name,",")
gen str famname = substr(name,1,comma-1)

* Makros
use data/data1, clear
local l 123
local l2 1+1
local l3 = 1+1

local tr `tr' Hallo
local tr `tr' wie
local tr `tr' geht
dis "`tr'"

global g Peter und Paul
macro list
display `l'
display "$g"

* Skalare und Matrizen
scalar s = 5
scalar list s
matrix m = (1,2/3,4)
matrix list m
matrix t =m * s
matrix list t

* Ergebnisse
summarize income
return list
local mean = r(mean)
display `mean'
regress income gender
ereturn list
matrix coef = e(b)
matrix list coef

```

- b) Berechnen Sie mit der implementierten Taschenrechneroption den Wert von $\sqrt{(2^2 + 4^2)/2 + 3}$.
- c) Erstellen Sie eine Liste der Personennummer, des Einkommens und des Geschlechts für die 10 Personen mit dem höchsten Einkommen.
- d) Berechnen Sie Mittelwert und Standardabweichung der Ausbildungsdauer von verheirateten Frauen, die in Bayern wohnen.
- e) Erstellen Sie eine Kreuztabelle, in der Sie die Zahl der Beobachtungen für die einzelnen Bundesländer und das Geschlecht abtragen. Sind alle Beobachtungen in dieser Tabelle ausgewiesen?
- f) Erzeugen Sie zwei lokale Makros var1 und var2 die die Variablenlisten 'income gender ybirth' bzw. 'edu occ hhinc' enthalten. Führen Sie diese beiden Makros in einem neuen Makro var3 zusammen.

8.12 Fingeruebung (finger.do)

Folgende Fragen sollen innerhalb eines Do-Files bearbeitet werden:

- a) Versetzen Sie dieses do-file mit einem informativen Header, der es lauffähig macht (cd?, log?, version?, Datum?...).
- b) Laden Sie das file „structure.dta“ herunter (<https://t1p.de/stata-files> — Passwort: happystata), öffnen Sie die Daten in Stata und verschaffen Sie sich einen Überblick über die Daten.
- c) Verändern Sie die Variable *v* so, dass diese Exporte in US\$ anzeigt.
- d) Erstellen Sie eine Variable welche die Exporte in Euro anzeigt. Der Wechselkurs beträgt 1.30US\$ pro Euro.
- e) Erstellen sie eine Dummy Variable, die den Wert 1 für Länder mit über 20.000.000 Einwohner annimmt.
- f) Erstellen Sie mehrere Dummy Variablen, die jeweils 1 sind für die 10 Länder welche
 - die geringste Arbeitslosenquote,
 - den rigidesten Arbeitsmarkt,
 - das größte GDP per capitaim Jahr 2008 aufweisen.
- g) Erstellen Sie eine Variable *numGDP*, die 1 ist für die 10 Länder mit den höchsten GDP per capita, 2 für die 10 Länder mit dem zweithöchsten GDP per capita, usw für das Jahr 2008.
- h) Erstellen Sie Dummy Variablen für jede Ausprägung von *numGDP*.
- i) Löschen Sie die in h erstellten Variablen wieder.
- j) Versetzen Sie die Variable *numGDP* mit einem Label.
- k) Addieren Sie zu jedem Wert der Variable *numGDP* 23 dazu.
 - l) Ziehen Sie von allen ungeraden Zahlen der Variable *numGDP* 3 ab.
- m) Ziehen Sie von allen geraden Zahlen der Variable *numGDP* 3 ab.
- n) Speichern Sie den bisher erstellten Datensatz unter *structure_ich.dta*.
- o) Erstellen Sie einen Datensatz, der die Mittelwerte aller Länder und aller Variablen über die Zeit beinhaltet. Speichern Sie diesen unter *test_mean.dta* ab. Hinweis: Der Mittelwert von Dummies und String-Variablen ist sinnlos und wird deswegen verworfen.
- p) Erzeugen Sie eine Excel Datei, welche die Variablen *i3 e_pop e_ur e_gdppp* beinhaltet. Die Werte in der Excel-File sollten absteigend nach Einwohnerzahl sortiert sein. Die Werte in dem Excel-File sollten so sortiert sein, dass die Länder mit den meisten Einwohnern oben stehen.

8.13 Die Biersteuer und Verkehrstote (beertax.do)

Lesen Sie diese Daten <http://fmwww.bc.edu/ec-p/data/stockwatson/fatality.dta>

in Stata ein und machen Sie sich damit vertraut. Die Datenbeschreibung finden Sie hier:

<http://fmwww.bc.edu/ec-p/data/stockwatson/fatality.des>

Hilfreiche Befehle: `.scatter` , `.reg` , `.areg` , `.tab` , `.xi i.variable, prefix(_DY)` , `.tabulate variable, generate(_DY)` , `.xtset` , `.areg` , `.xtreg`

- a) Machen Sie sich mit dem Datensatz vertraut. Welche Variablen identifizieren den Datensatz? Welche Variablen beschreiben die Daten genauer?
- b) Die Variable `mrall` zeigt die Todesrate pro 10.000 Einwohner für einen Staat in einem gegebenen Jahr dar. Verändern sie Variable `mrall` so, dass sie die Verkehrstoten pro 10.000 Einwohner anzeigt.
- c) Berechnen Sie für jeden Staat die Wachstumsrate der Verkehrstoten. (Hinweis: Google darf benutzt werden und mit Suchbegriffen auf Englisch findet man oft mehr.)
- d) Untersuchen Sie nun wie sich eine Biersteuer auf die Anzahl der Verkehrstoten auswirkt.

8.14 Guns (guns.do)

Lesen Sie diese Daten

<http://www2.econ.iastate.edu/classes/econ371/McPhail/Guns.dta>

in Stata ein und machen Sie sich damit vertraut. Die Datenbeschreibung finden Sie hier:

<http://www2.econ.iastate.edu/classes/econ371/McPhail/Guns.pdf>

- Laden Sie den Datensatz Guns.dta und verschaffen Sie sich, auch mithilfe der Datenbeschreibung in guns.pdf einen Überblick über die Daten.
- Checken Sie, ob der Datensatz von stateid und year eindeutig identifiziert wird.
- Wenn dies nicht der Fall wäre, wie könnten Sie das bereinigen?
- Beinhalten die identifizierenden Variablen missings? Wenn ja, bereinigen Sie dies.

- Die Variablen sind nicht sauber gelabelt. Ergänzen Sie folgende fehlende Labels:

```
label variable pb1064 "percent of state population that is black, ages 10 to 64"
label variable pop "state population, in millions of people"
label variable pm1029 "percent of state population that is male, ages 10 to 29"
label variable pb1064 "percent of state population that is black, ages 10 to 64"
label variable pw1064 "percent of state population that is white, ages 10 to 64"
label variable year "Year (1977-1999)"
label variable stateid "ID number of states (Alabama = 1, Alaska = 2, etc.)"
label variable avginc "real per capita personal income in the state, in thousands of dollars"
label variable shall "=1 if the state has a shall-carry law in effect in that year=0 otherwise"
label variable density "population per square mile of land area, divided by 1000"
```

- Veranschaulichen Sie die Entwicklung der Kriminalität über die Zeit in Tabellen und/oder Grafiken. Haben Sie eine Erklärung?

- Erzeugen Sie folgende Variablen:

- Erzeugen Sie eine Variable "mean_vio", welche die über die Zeit durchschnittliche Kriminalitätsrate für alle Staaten beinhaltet.
- Erzeugen Sie eine Variable "mean_vio_id", welche die über die Zeit durchschnittliche Kriminalitätsrate für jeden Staat einzeln beinhaltet.
- Erzeugen Sie eine Dummy Variable "Davg_vio", die 1 ist für Staaten mit einer überdurchschnittlichen Kriminalitätsrate und ansonsten 0.

- „Shall-carry law“ bedeutet grob, dass Personen in einem Bundesstaat verdeckt Waffen tragen dürfen. Sie vermuten, dass dies die Kriminalität beeinflussen kann. Im Folgenden sollen Sie versuchen diesem Zusammenhang zu ergründen. Generieren Sie zunächst eine Variable „sh_years“, die ausweist, wie viele Jahre dieses Gesetz in den jeweiligen Staaten in Kraft war. Tipp: `.help egen`

- Generieren Sie eine Variable `sh_new`, die das Jahr, indem das Gesetz in Kraft trat, mit einer 1 ausweist und ansonsten 0 ist. Lösung:

```
sort stateid year
gen sh_new=1 if shall[_n]!=shall[_n-1] & stateid[_n]==stateid[_n-1]
replace sh_new=0 if implementation==.
```

- Können Sie sich die folgende Kommandozeile erklären?

```
by stateid: gen sh_ch_year = 1900 + year if sh_new ==1
mean vio mur rob , over(shall)
mean vio mur rob [aweight=pop] , over(shall)
```

- k) Bewirkt das Gesetz etwas bzgl. *vio mur rob*? Vergleichen Sie den Mittelwert von *vio mur rob* derjenigen Staaten, die das Gesetz niemals hatten, mit Staaten, die das Gesetz immer (vor 1977) hatten.
- l) Betrachten Sie die Mittelwerte von *vio mur rob* bei Staaten, die das Gesetz während der beobachteten Periode eingeführt haben vor und nach der Einführung*/
- m) Berechnen Sie die durchschnittliche Wachstumsraten „gr_*“ für *vio mur rob*.
- n) Wie ist die durchschnittliche Wachstumsrate von *vio mur rob* vor und nach der Einführung des Gesetzes?
- o) Versuchen Sie, folgende Kommandos nachzuvollziehen:
- ```

mean vio [aweight=pop] if sh_years==23, over(year)
matrix sh1 = 'e(b)'
mean vio [aweight=pop] if sh_years>0 & sh_years<23, over(year)
matrix sh2 = 'e(b)'
mean vio [aweight=pop] if sh_years==0, over(year)
matrix sh0 = 'e(b)'
svmat sh1, names(sh1_)
svmat sh0, names(sh0_)
svmat sh2, names(sh2_)
egen year2 = seq() in 1/23, from(1977) to(1999)
line sh0 sh1 sh2 year2, title(Gewaltkriminalitätsrate in den USA 1977 - 1999) ///
ytitle(Rate pro 100000 Bürgern) xtitle(Jahr) yscale(range(0)) ylabel(0(100)900, angle(horizontal)) ///
legend(label(1 SStaaten, die nie das Gesetz verabschiedet haben") ///
label(2 SStaaten, die das Gesetz vor 1977 verabschiedet haben") ///
label(3 SStaaten, die das Gesetz nach 1977 verabschiedet haben")cols(1))

```
- p) Versuchen Sie, das Ergebniss der vorhergehenden Aufgabe zu replizieren, indem sie den `.collapse` Befehl verwenden.
- q) Regressieren Sie nun (1)  $\ln(vio)$  auf *shall* und (2)  $\ln(vio)$  auf *shall*, *incarc\_rate*, *density*, *avginc*, *pop*, *pb1064*, *pw1064* und *pm1029*. Interpretieren sie die Koeffizienten von *shall*. Sind sie statistisch signifikant? Sind sie ökonomisch signifikant.
- r) Schätzen Sie Modell (2) mit fixen Effekten und vergleichen Sie die Ergebnisse. Schätzen Sie anschließend mit zusätzlichen Zeitdummies. Vergleichen Sie erneut. Was fällt auf? Nennen Sie Gründe, die weiterhin an der Validität der Ergebnisse Zweifel aufkommen lassen könnten.

## 8.15 Schleifen (schleifen.do)

- a) Machen Sie sich im Hilfemenü mit der forvalues Schleife vertraut.
- b) Machen Sie sich im Hilfemenü mit der foreach Schleife vertraut.
- c) Folgende foreach Schleife zeigt nacheinander Großbuchstaben (A,B,C) und Zahlen (1, 2, 3, 4) an. In welcher Reihenfolge werden diese Zeichen geschrieben?

```
local n=1
local x "A B C"
foreach f of local x {
local n='n'+1
dis "'n'"
dis "'f'"
}
```

- d) Machen Sie sich im Hilfemenü mit der while Schleife vertraut.
- e) Speichern Sie den Datensatz „wage.dta“ (<https://t1p.de/stata-files> — Passwort: happystata) in ihren Ordner „data“. Diese Datei dürfte Ihnen aus Ökonometrie 1 noch bekannt sein. Öffnen Sie diese Datei.
- f) Regressieren Sie nun den Lohn separat auf *iq*, *educ* und *exper*.
- g) Versuchen Sie dasselbe nun mit einer foreach-Schleife.
- h) Regressieren Sie nun für jede Ausprägung von *educ* (9-18) separat:  $wage=iq+exper$ . Versuchen sie dies mit einer foreach-Schleife. Tipp: versuchen Sie dies mit `.levelsof` .
- i) Erstellen Sie eine Variable  $D\_iq$ , die den Wert 1 für alle Beobachtungen mit einem  $iq < 90$  annimmt, 2 für ...  $iq > 110$  und 3 für ... alle dazwischenliegenden *iq* Beobachtungen.
- j) Regressieren Sie nun den Lohn separat auf *educ* und *exper*, jeweils einmal für jede Ausprägung von  $D\_iq$ . Dies ergibt insgesamt sechs Regressionen (drei für *educ* und drei für *exper*). Versuchen Sie hierzu die forvalues Schleife in eine foreach Schleife zu integrieren.
- k) Ersetzen Sie nun die forvalues Schleife mit einer while Schleife.
- l) Stellen Sie den Zusammenhang zwischen dem Stundenlohn (*wage/hours*) und *iq* graphisch dar.
- m) Stellen Sie den Zusammenhang zwischen dem Stundenlohn (*wage/hours*) und *iq* graphisch für alle Ausprägungen von *educ* dar und speichern Sie die Graphiken. Benutzen Sie für den gesamten Vorgang eine foreach Schleife.

## 8.16 Tabellen und Grafiken (tabellen\_grafiken.do)

Im Folgenden untersuchen wir, ob sich die Wohnungsgrößen von Frauen und Männern innerhalb der Bundesländer unterscheiden.

- a) Kohler and Kreuter (2016) stellt eine Reihe von Datensätzen zur Verfügung. Installieren Sie diese in dem Ordner „data“. Geben Sie hierzu `.net from http://www.stata-press.com/data/kkd/` in Stata ein. Alternativ können sie die Daten hier runterladen:  
`http://www.stata-press.com/data/kkd.html`
- b) Laden Sie den Datensatz „data1.dta“ und erzeugen Sie eine Tabelle, in der die Bundesländer und deren durchschnittliche Wohnungsgröße abgebildet sind. Versuchen Sie je eine Lösung mit `.table` und `.tabstat` zu finden.
- c) Erzeugen Sie eine Tabelle, in der die Bundesländer und deren durchschnittliche Wohnungsgröße abgebildet sind, sowie Maximum, Minimum, Median und Standardabweichung. Versuchen Sie je eine Lösung mit `.table` und `.tabstat` zu finden.
- d) Erzeugen Sie diese Tabelle nun für Männer und Frauen getrennt mit einer Befehlszeile. Versuchen Sie je eine Lösung mit `.table` und `.tabstat` zu finden.
- e) Erzeugen Sie eine Tabelle, in der die durchschnittliche Wohnungsgröße für beide Geschlechter abgebildet ist. Versuchen Sie je eine Lösung mit `.table` und `.tabstat` zu finden.
- f) Fällt Ihnen ein Unterschied auf zwischen `.table` und `.tabstat` ? Versuchen Sie „format(%10.2f)“ als Option bei `.table` anzugeben. Was verändert sich? Können Sie über das Menü diese Option finden und ändern? (Hilfe: table→Options→Override display format→Create)
- g) Manipulieren Sie die Tabelle aus der vorherigen Aufgabe f indem Sie alle Mehr-Personen-Haushalte ausschließen.
- h) Erweitern Sie die Tabelle aus der vorherigen Aufgabe f indem sie diese für jedes Bundesland separat anzeigen. Versuchen Sie je eine Lösung mit `.table` und `.tabstat` zu finden.
- i) Stellen Sie die Tabelle aus Aufgabe c grafisch dar. Hinweis: `.graph bar` . Um die Bundesländer lesbar zu machen, kann man diese um 90° drehen. Versuchen Sie dies zunächst über das Menü zu finden.
- j) Für eine bessere Übersicht ist es sinnvoll, die Bundesländer mit der größten durchschnittlichen Wohnungsgröße zuerst anzuführen. Wie geht das?
- k) Es interessiert Sie nun, wie stark die Wohnungsgrößen von Frau und Mann schwanken. Erstellen Sie zwei Boxplots in einer Grafik, welche diese Schwankungen anzeigen.
- l) Erstellen Sie für jedes Bundesland einen Boxplot, der die Schwankungen der Wohnungsgröße angibt.
- m) Stellen Sie die mittlere Wohnungsgröße je Gebäudetyp in einem Balkendiagramm dar. Achten Sie auf eine angemessene Gestaltung der Grafik. Zeichnen Sie den Mittelwert aller Wohnungen in die Grafik mit ein. Lösung:

```
sum sqm if htype<.
local mean = r(mean)
graph bar sqm, over(htype, label(angle(45))) ///
title("Mittlere Wohnfläche") subtitle(Nach Gebäudeart -) ///
note("Daten: GSOEP") ytitle(Wohnfläche in qm) yline('mean') ///
text('mean' 50 "Mittelwert", placement(n) color(red))
```

- n) Stellen Sie die Einkommensverteilung in den verschiedenen Bundesländern mit Hilfe eines Punktdiagramms dar. Wählen Sie dazu aussagekräftige Kennziffern. Achten Sie wiederum auf eine angemessene Gestaltung. Was lässt sich erkennen? Lösung:

```
quietly centile income if state<. , centile (20 50 80)
local p20 = r(c_1)
local p50 = r(c_2)
local p80 = r(c_3)
graph dot (p20) income (median) income (p80) income, over(state, sort(2) total) ///
ytitle(Einkommen in Euro) ///
yline('p20', lcolor(blue)) yline('p50', lcolor(red)) yline('p80', lcolor(green)) ///
title("Verschiedene Quantile der Einkommensverteilung") ///
subtitle(nach Bundesländern -) ///
legend(label(1 "20 Perzentil") label(2 "Median") label(3 "80 Perzentil") col(3)) ///
note("Daten: GSOEP") ///
caption("Die vertikalen Linien zeigen die jeweiligen Quantile über alle Bundesländer")
```

- o) Führen Sie einen graphischen Test durch, ob das logarithmierte Einkommen einer Normalverteilung folgt. Lösung:

```
gen l_income = log(income)
histogram l_income, normal normopts(lcolor(blue)) kdensity kdenopts(lcolor(red)) ///
title("Verteilung des Log-Einkommens") ///
subtitle(Histogramm mit Normalverteilung und Kerndichteschätzer -) ///
ytitle("Dichte") ///
xtitle(Einkommen in Euro) ///
note("Daten: GSOEP") ///
caption("Blaue Linie: Normalverteilung Rote Linie: Epanechnikov-Kerndichteschätzer")
```

- p) Laden Sie den Datensatz „inflation.dta“ herunter (<https://t1p.de/stata-files> — Passwort: happystata) und stellen Sie die Entwicklung der Inflationsraten in Deutschland, Frankreich und Großbritannien in geeigneter Weise dar. Achten Sie wieder auf die Beschriftung der Grafik. Welche Interpretation bietet sich an?

```
use inflation, clear
line inf year if country=="Germany" | line inf year if country=="France" | line inf year if country=="United Kingdom", ///
legend(label(1 "Deutschland") label(2 "Frankreich") label(3 "Großbritannien") cols(3)) ///
title(„Inflationsraten 1970 - 2000“) ///
subtitle(In Deutschland, Frankreich und Großbritannien -) ///
note("Quelle: unbekannt (das sollte einem möglichst nicht passieren)")
```

- q) Laden Sie guns.dta und zeigen Sie für *stateid* 1 und 2, wie sich *rob* über die Zeit entwickelt hat. Speichern Sie die jeweiligen Grafiken mit der Option „saving()“.

- r) Kombinieren Sie nun die gespeicherten Grafiken und speichern Sie diese als .gph und als .wmf-Datei ab.

- s) Stellen Sie die beiden Linien in einer Grafik dar und machen sie die Legende lesbar.

- t) Für viele Länder ist das mühsam. Stata bietet zwei Lösungen:

```
line vio year, sort by(stateid, total)
xtset stateid year
xtline vio
xtline vio, overlay
```

## 8.17 Panel Bild Alternativen (panel\_pic.do)

Oft will man für mehrere Firmen oder für Länder einen Indikator im Zeitablauf betrachten. Stata bietet hier verschiedenste Wege **eine** Grafik zu erstellen. Lassen Sie folgendes in Stata (und Ihrem Kopf) durchlaufen.

```
*Panel_Bild_Alternative

*Alternative 1
clear all
webuse grunfeld,clear
levelsof com, local(levels)
keep mvalue year company
reshape wide mvalue, i(year) j(company)
twoway (line mval* year)

*Alternative 2
clear all
webuse grunfeld
levelsof com, local(levels)
foreach l of local levels {
 local gr 'gr' line mval time if com == 'l' ||
 }
 display "'gr'"
graph twoway 'gr' , legend('le') name(gr1)

*Alternative 3
clear all
webuse grunfeld
levelsof com, local(levels)
sort company time
line mval time, c(L) name(gr2) //see: .help connectstyle

*Alternative 4
clear all
webuse grunfeld
xtset company time
xtline mvalue, over

*Alternative 5
clear all
ssc install lgraph
webuse grunfeld
lgraph mval time, by(company)
```

## 8.18 Konvergenz (konvergenz.do)

Der Datensatz klausur2012.dta, den sie auf der elearning Plattform finden, enthält das pro Kopf GDP von 1960 (gdppc60) und die durchschnittliche Wachstumsrate des GDP pro Kopfes zwischen 1960 und 1995 (growth) für verschiedene Länder (country), sowie 3 Dummy-Variablen die mit einer '1' die Zugehörigkeit eines Landes zu der Region Asien (asia), Westeuropa (weurope) oder Afrika (africa) indizieren und ansonsten den Wert '0' besitzen.

- 1) Nennen Sie die Länder welche nicht in Westeuropa, Afrika oder Asien liegen? Sollten Sie Länder finden welche Mitglieder der EU sind weisen sie diesen eine '1' in der Variable weurope zu.
- 2) Erstellen Sie eine Tabelle die für alle vorhandenen Zeitpunkte das durchschnittliche GDP pro Kopf anzeigen. Unterscheiden sie dabei in Westeuropäische Länder, nicht Westeuropäische Länder und alle Länder.
- 3) Erstellen sie die Wachstumsrate des GDP pro Kopf von 1960 auf 1995 und nennen Sie diese gdpgrowth. (Anmerkung: Der logarithmierte Wert X abzüglich des logarithmierten Wertes X der Vorperiode ist approximativ gleich der Wachstumsrate.)
  - a) Berechnen Sie die unbedingte Konvergenz aller Länder indem sie eine Grafik erstellen in welcher ein Scatterplott abgetragen ist mit der durchschnittlichen Wachstumsrate des GDP pro Kopfes zwischen 1960 und 1995 (gdpgrowth) auf der y-Achse und dem pro Kopf GDP von 1960 (gdppc60) auf der x-Achse. Zusätzlich ist in dieselbe Grafik der geschätzte lineare Zusammenhang darzustellen. Sie müssen die Grafik nicht weiter beschriften, nur zwei Sachen: betiteln Sie die Grafik mit 'world' und labeln Sie die einzelnen Beobachtungen mit den Ländernamen. Speichern Sie die Grafik im Stata Format unter der Bezeichnung 'world' ab. (Anmerkung: Sollten sie gdpgrowth in Aufgabe 3 nicht berechnen können verwenden sie Alternativ die bereits im Datensatz enthaltene Variable growth.)
  - b) Erstellen Sie mit Hilfe einer foreach-Schleife 3 Grafiken welche denselben Zusammenhang wie in Aufgabe 2 beschreiben. Reduzieren Sie nun aber das Sample jeweils auf Westeuropäische, Afrikanische und auf Asiatische Länder. Betiteln sie die Grafik entsprechend mit 'weurope', 'africa' und 'asia'. Speichern sie diese 3 Grafik ebenfalls im Stata Format ab.
  - c) Kombinieren Sie die 4 abgespeicherten Grafiken in ein Bild. Und interpretieren Sie die Grafiken kurz. Erörtern Sie dabei was eine steigende und was eine fallende Schätzgerade bedeuten könnte?
- 4) Schätzen Sie mit der Methode der kleinsten Quadrate die in den 4 Grafiken veranschaulichten Zusammenhänge. Stellen sie die 4 Schätzergebnisse in einer Tabelle dar, wobei das Signifikanzniveau mit Sternen angezeigt werden soll. Zusätzlich soll in der Tabelle das Akaike Informationskriterium, sowie das Bestimmtheitsmaß und die Anzahl an Beobachtungen ausgegeben werden. Interpretieren sie die 4 Schätzergebnisse hinsichtlich ihrer Signifikanz.
- 5) Bringen sie den Datensatz ins long-Format und berechnen sie die Wachstumsraten des GDP pro Kopf für die vorhandenen Zeitpunkte in den Ländern.

## 8.19 Zipf's Law (zipf.do)

Laden Sie den Datensatz `cityGER.dta` herunter (<https://t1p.de/stata-files> — Passwort: `happystata`) und speichern Sie diesen in Ihrem persönlichen Laufwerk. Der Datensatz umfasst Daten der deutschen Volkszählung der Jahre 1970, 1980 und 2011.

- Finden Sie heraus welche Variable(n) die Beobachtungen des Datensatzes eindeutig identifizieren.
- Erstellen Sie eine Tabelle, welche die Anzahl der Beobachtungen, den Mittelwert, die Standardabweichung, sowie Minimum und Maximum der Variablen `pop1970`, `pop1987` und `pop2011` enthält.
- Löschen Sie alle numerischen Variablen die ein Missing enthalten.

Im Folgenden untersuchen Sie die Gültigkeit von Zipf's Law für Deutschland. Zipf's Law postuliert wie die Größe von Städten verteilt ist. Dieses „Gesetz“ besagt, dass es einen log-linearen Zusammenhang zwischen der Größe einer Stadt und dem Rang, den eine Stadt in einer nach der Stadtgröße sortierten Reihe einnimmt, gibt. In der Schätzgleichung

$$\log(M_j) = c - q \log(R_j),$$

postuliert das Gesetz einen Koeffizienten von  $q = 1$  ( $c$  ist eine Konstante;  $M_j$  ist die Größe der Stadt  $j$ ;  $R_j$  ist der Rang den die Stadt  $j$  in einer nach der Stadtgröße sortierten Reihe einnimmt).

- Erstellen Sie eine Variable Namens `rank` die ein Ranking der Städte bezüglich der Größe der Bevölkerung im Jahr 2011 beinhaltet. Berlin sollte also eine 1 aufweisen, Hamburg eine 2, München eine 3, usw.

*(Hinweis: Sollten Sie diese Aufgabe nicht lösen verwenden Sie im folgenden die Variable `rankX` als Ersatz für die nicht erzeugte Variable `rank`.)*

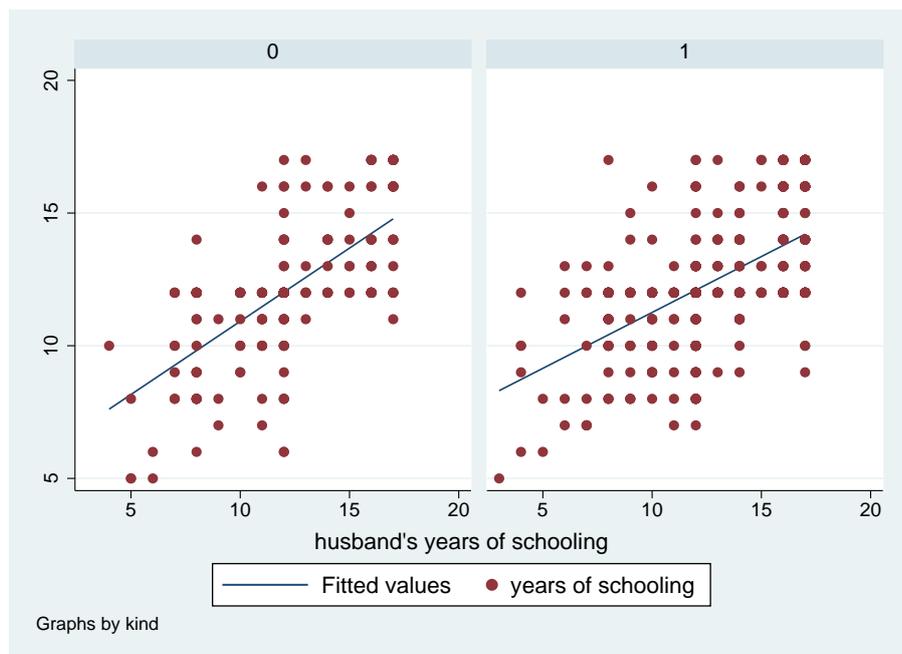
- Logarithmieren Sie die Variablen `rank` und `pop2011`. Benutzen Sie die neue Variablen mit `lnrank` und `lnpop2011`. Benützen Sie hierzu eine `foreach`-Schleife.
- Labeln Sie die Variablen `lnpop2011` mit „log Einwohner 2011“, sowie `lnrank` mit „Städte-Ranking“.
- Erstellen Sie einen Scatter-Plot. Auf der Abszisse (x-Achse) soll die Variable `lnrank` abgetragen sein und auf der Ordinate (y-Achse) `lnpop2011`. Fügen Sie dem selben Scatter-Plot eine Regressionsgerade des angezeigten Zusammenhangs hinzu.
- Testen Sie nun den im Zipf's Law postulierten Zusammenhang. Regressieren Sie hierzu die logarithmierte Stadtgröße des Jahres 2011 auf den logarithmierten Rang einer Stadt in einer nach der Stadtgröße sortierten Reihe. Interpretieren Sie das Ergebniss kurz, gehen Sie hierbei auf das Bestimmtheitsmaß ein.
- Testen Sie, ob der Schätzkoeffizient von `lnrank`, wie von Zipf's Law postuliert gleich Eins ist.

*(Hinweis: `help test`)*

- Geben Sie an wie viele Einwohner in Regensburg 2011 nach Angaben des Census wohnen und auf welchem Rang im Städtegrößenranking sich Regensburg befindet.
- Geben Sie an wie viele Einwohner nach der Schätzung der vorangegangenen Aufgabe in Regensburg wohnen.
- Regressieren Sie nun für alle Bundesländer (`state`) besagten Zusammenhang separat.
- Bonus Aufgabe:** Laden Sie den Datensatz `cityGER.dta` erneut. Löschen Sie alle Variablen bis auf `city`, `state`, `pop1970`, `pop1987` und `pop2011`. Der Datensatz befindet sich im sogenannten wide-Format. Bringen Sie diesen in das sogenannte long-Format. (5P)

## 8.20 Das Einkommen von verheirateten Frauen (marry.do)

- a) Laden Sie folgende Daten aus dem Internet und speichern Sie diese auf auf ihre Festplatte. Geben Sie hierzu direkt in Stata die entsprechenden Befehle ein, um die Daten zu laden und zu speichern. (5P)  
<http://www.stata.com/data/jwooldridge/eacsap/mroz.dta>
- b) Der Datensatz beinhaltet Informationen zu verheirateten weißen Frauen des Jahres 1975. Finden Sie heraus, ob diese Frauen durch eine Variable im Datensatz identifiziert werden. Ist dies nicht der Fall, erzeugen Sie eine identifizierende Variable mit den Variablennamen *id*. Prüfen Sie anschließend, ob die Variable *id* den Datensatz eindeutig identifiziert. (5P)
- c) Erstellen Sie eine Tabelle in der für alle Variablen die Anzahl der Beobachtungen, der Mittelwert, die Standardabweichung, das Minimum und das Maximum abgetragen ist. (3P)
- d) Die Variable *inlf* zeigt an, ob eine Frau im abgelaufenen Jahr beschäftigt war (=1) oder nicht (=0). Sie vermuten, dass ein geringes Einkommen des Mannes die Wahrscheinlichkeit erhöht, dass eine Frau beschäftigt ist. Testen Sie diese Vermutung mit Hilfe einer Probit-Schätzung (*Hinweis: help probit*). Erstellen Sie hierfür zunächst eine Variable namens *incman* die anzeigt wie viel der Mann verdient. (*Hinweis: faminc zeigt an wie viel beide Ehepaare zusammen verdienen, wage zeigt an wie viel eine Frau in der Stunde durchschnittlich verdient und hours zeigt an wie viele Stunden eine Frau gearbeitet hat*). Regressieren Sie *incman* auf *inlf*. Kann die Vermutung bestätigt werden? (7P)
- e) Fügen Sie nun zu der in Aufgabe d) durchgeführten Regression noch folgende Kontrollvariablen hinzu: *motheduc fatheduc huseduc exper expersq age kidslt6 kidsge6*. Kommentieren Sie nun folgendes Statement: „Die Partizipation der Frau am Arbeitsmarkt ist unabhängig vom Einkommen des Ehemannes.“ (5P)
- f) Erstellen Sie eine Dummy-Variable Namens *kind* welche Eins ist, wenn eine Frau mindestens ein Kind im Alter bis zu 18 Jahren hat und Null für alle sonstigen Beobachtungen. (4P)
- g) Replizieren Sie folgende Grafik und spekulieren Sie wie der scheinbar positive Zusammenhang von *educ* und *huseduc* interpretiert werden könnte. (6P)



- h) Erklären Sie den Stundenlohn von Frauen anhand der Mincer-Lohngleichung. Die Schätzgleichung lautet:

$$\ln wage_i = \alpha_0 + \alpha_1 educ_i + \alpha_2 exper_i + \alpha_3 expersq_i + \epsilon_i.$$

Diese Gleichung geht davon aus, dass der Lohn durch die Zahl der Ausbildungsjahre und der Berufserfahrung bestimmt wird. Letztere wird zudem im quadrierten Term in die Gleichung eingeführt, um den beobachteten sinkenden Löhnen im Alter Rechnung zu tragen. Verwenden Sie robuste Standardfehler. (5P)

- i) Erstellen Sie eine Variable die angibt wie viele minderjährige Kinder (<18 Jahre) eine Frau hat. Nennen Sie diese Variable *anzkind* und labeln Sie diese Variable mit „Anzahl der Kinder“. (5P)
- j) Testen Sie, ob Kinder einen Einfluss auf den Lohn der Mutter besitzen. Schätzen Sie hierzu folgende 3 Gleichungen und stellen Sie die alle drei Schätzungen in einer einzigen Tabelle dar. Achten Sie darauf, dass die Signifikanz der dargestellten Koeffizienten durch Sternchen indiziert wird und für jede Regression das Bestimmtheitsmaß angezeigt wird.

$$\ln wage_i = \alpha_0 + \alpha_1 educ_i + \alpha_2 exper_i + \alpha_3 expersq_i + \alpha_4 kidslt6_i + \alpha_5 kidsge6_i + \epsilon_i$$

$$\ln wage_i = \alpha_0 + \alpha_1 educ_i + \alpha_2 exper_i + \alpha_3 expersq_i + \alpha_4 kind_i + \epsilon_i$$

$$\ln wage_i = \alpha_0 + \alpha_1 educ_i + \alpha_2 exper_i + \alpha_3 expersq_i + \alpha_4 anzkind_i + \epsilon_i$$

Interpretieren Sie die Ergebnisse kurz. (10P)

- k) Erstellen Sie mit Hilfe des `.collapse` Befehls eine Tabelle bzw. einen Datensatz, welcher wie folgt aussieht: (5P)

| city | hours    | faminc   | sdhours  | sdfaminc |
|------|----------|----------|----------|----------|
| 0    | 759.5167 | 19075.22 | 880.9531 | 10506.88 |
| 1    | 730.0496 | 25306.72 | 866.6475 | 12498.29 |

(Hinweis: Bei der Variable *hours* und *faminc* handelt es sich um den Mittelwert, bei *sdhours* und *sdfaminc* handelt es sich um die Standardabweichung.)

- l) **Bonus Aufgabe:** Führen Sie folgenden Befehl aus:

```
.use http://www.uam.es/personal_pdi/economicas/rsmanga/docs/wage1.dta, clear
```

Der hiermit geladene Datensatz beinhaltet wieder Lohndaten von Individuen. Löschen Sie alle Beobachtungen, außer es handelt sich um verheiratete (*married*=1) weiße (*nonwhite*=0) Frauen (*female*=1). Kombinieren Sie nun diesen Datensatz mit den oben verwendeten Datensatz *mroz.dta* und schätzen Sie die Mincer-Lohnleichung aus Aufgabe 1h) erneut. (10P)

## 8.21 Migration in Deutschland (migration.do)

Im folgenden arbeiten Sie mit dem IAB Brain-Drain Datensatz. Dieser umfasst die Gesamtzahl der im Ausland geborenen Personen im Alter von 25 Jahren und mehr, die in einem der 20 OECD-Zielländer leben, nach Jahr, Geschlecht, Herkunftsland und Bildungsstand. Das Bildungsniveau wird in niedrig, mittel und hoch qualifiziert unterschieden. Die Beschreibung der Daten finden Sie—falls benötigt—hier: <https://www.iab.de/de/daten/iab-brain-drain-data.aspx>

- a) Bitte laden sie folgende Daten direkt aus dem Netz in Stata und finden Sie heraus, welche Variablen die Beobachtungen identifizieren:

```
.use https://doku.iab.de/daten/brain-drain/iabbd_8010_v1.dta, clear
```

- b) Die Variablen `gender` und `year` sind nicht gelabelt. Versetzen Sie diese Variablen mit einem aussagekräftigen Label.
- c) Sie interessieren sich ausschließlich für die Migration nach Deutschland. Bereiten Sie den Datensatz so auf, dass nur diese Information enthalten bleibt.
- d) Erläutern Sie, wie man in Stata einen Wiederherstellungspunkt setzen kann und wie man zu diesem Punkt zurückkehrt.
- e) Berechnen Sie die fünf Länder, aus denen Deutschland über alle Jahre hinweg am meisten Migranten empfängt. Zeigen Sie diese fünf Länder mit der Anzahl der Migranten über alle Jahre in Stata an.
- f) Erstellen Sie ein Balkendiagramm, welches im Zeitablauf anzeigt wie viele Migranten in der Summe nach Deutschland gekommen sind. Die y-Achse sollte hierbei die Summe der Migranten in Millionen anzeigen.
- g) Erstellen Sie drei Variablen, welche die Anzahl der niedrig-, mittel- und hochqualifizierten pro Tag anzeigt. Gehen Sie davon aus, dass ein Jahr aus 360 Tagen besteht.

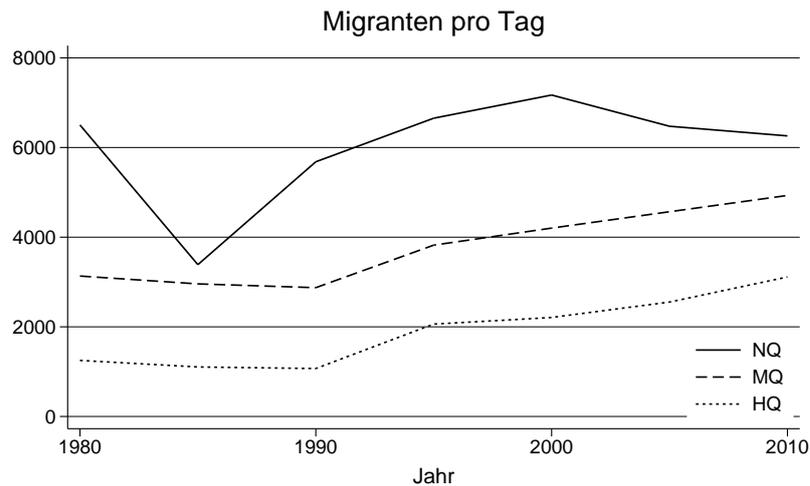
*(Hinweis: Wenn Sie es schaffen, die Variablen mit Hilfe einer Schleife zu erzeugen, erhalten Sie 2 Bonus-Punkte.)*

- h) Erstellen Sie drei Liniendiagramme, welche jeweils die Summe der niedrig-, mittel- und hochqualifizierten Migranten nach Deutschland pro Tag im Zeitablauf anzeigen und speichern Sie die drei Diagramme jeweils als \*.png-Datei ab.

*(Hinweis: Es hilft, den Datensatz zu ‘kollabieren’, siehe `.help collapse` .)*

*(Hinweis: Wenn Sie es schaffen, die Diagramme mit Hilfe einer Schleife zu erzeugen, erhalten Sie 3 Bonus-Punkte.)*

- i) Versuchen Sie nun, diese drei Bilder in einem Diagramm wie folgt darzustellen:



- j) Erstellen Sie jeweils ein Kuchendiagramm für das Jahr 1980 und 2010. Diese zwei Kuchendiagramme (auf englisch: pie plot) sollen die Anteile der niedrig-, mittel- und hochqualifizierten Migranten an der Gesamtzahl der Migranten anzeigen. Interpretieren Sie kurz, wie sich diese Aufteilung verändert hat.
- k) Laden Sie die IAB Brain-Drain Daten erneut und erstellen Sie eine Tabelle, welche die Summe der Migranten für alle Zielländer im Zeitablauf anzeigt. Die Tabelle soll wie folgt organisiert sein:

| Country name, destination | 1980    | 1985    | 1990 | 1995 | 2000 | 2005 | 2010 |
|---------------------------|---------|---------|------|------|------|------|------|
| Australia                 | 2323198 | 2593316 | ...  | ...  | ...  | ...  | ...  |
| Austria                   | 300177  | :       | :    | :    | :    | :    | :    |
| :                         | :       | :       | :    | :    | :    | :    | :    |

- l) Vervollständigen bzw. berichtigen Sie die folgende Befehle, so dass unten stehender Stata Output erzeugt wird:

```
use https://doku.iab.de/daten/brain-drain/iabbd_8010_v1.dta, clear
keep if destination=="Germany"
collapse () , by()
reshape wide tot, i() j() string
egen totboth=(totFemal totMale)
gsort -totboth
list in 1/5
```

```
+-----+
| origin totFem~e totMale totboth |
+-----+
1. | Turkey 3.1e+06 4.0e+06 7164094 |
2. | Italy 1.1e+06 1.8e+06 2940045 |
3. | Croatia 736197 1.0e+06 1779273 |
4. | Greece 701415 940276 1641691 |
5. | Poland 611876 708355 1320231 |
+-----+
```

## 8.22 Migration nach Deutschland

Im folgenden arbeiten Sie mit dem IAB Brain-Drain Datensatz. Dieser enthält für die Jahre 1980 bis 2010 (5-Jahres-Intervalle) Daten zur internationalen Migration für 20 OECD-Zielländer unterteilt nach Geschlecht, Herkunftsland und Bildungsstand. Das Bildungsniveau wird in niedrig, mittel und hoch qualifiziert unterschieden. Die Beschreibung der Daten finden Sie—falls benötigt—hier:

<https://www.iab.de/de/daten/iab-brain-drain-data.aspx>

- 1) Bitte laden Sie folgende Daten direkt aus dem Netz in Stata und finden Sie heraus, welche Variablen die Beobachtungen identifizieren:

```
.use https://doku.iab.de/daten/brain-drain/iabbd_8010_v1.dta, clear
```

- 2) Die Variablen `gender` und `year` sind nicht gelabelt. Versehen Sie diese Variablen mit einem aussagekräftigen Label.
- 3) Sie interessieren sich ausschließlich für die Migration nach Deutschland. Bereiten Sie den Datensatz so auf, dass nur diese Information enthalten bleibt.
- 4) Führen Sie folgenden Befehl in Stata aus und erläutern Sie, wie die Variablen `tot`, `low`, `med`, und `high` nun zu interpretieren sind.

```
collapse (sum) tot low med high, ///
by(ccode_destination ccode_origin year origin destination)
```

(Hinweis: Sollten Sie die Aufgaben bis hierher nicht lösen können, können Sie mit dem Datensatz `aftercollapse.dta` weiterarbeiten. Dieser ist hier zu finden: <https://t1p.de/stata-files> — Passwort: `happystata`.)

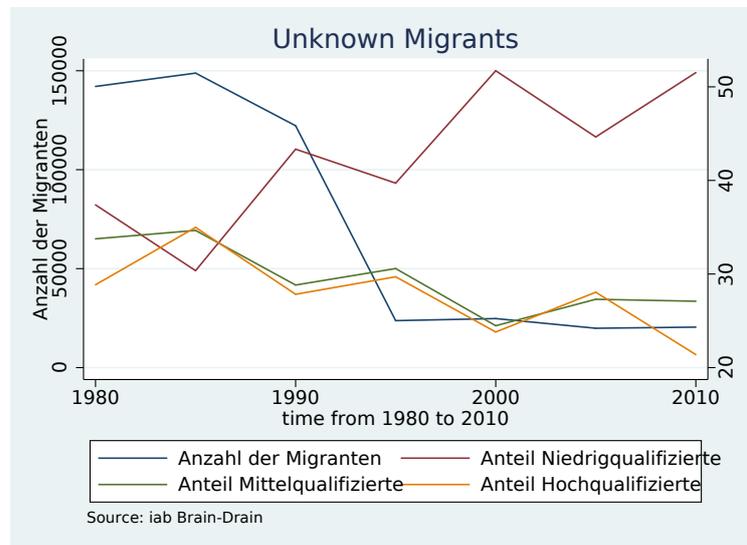
- 5) Sie bemerken, dass nicht alle Migranten einem Land zugeordnet werden können (“Unknown”). Stellen Sie durch den Befehl `.tabstat` dar, wie viele Migranten zu jedem verfügbaren Zeitpunkt nicht zugeordnet werden. Der Stata Output sollte wie folgt aussehen:

| year  | tot    | low    | med    | high   |
|-------|--------|--------|--------|--------|
| 1980  | 142072 | 53118  | 47962  | 40992  |
| 1985  | 148800 | 45176  | 51572  | 52052  |
| 1990  | 122200 | 52962  | 35224  | 34014  |
| 1995  | 23734  | 9425   | 7259   | 7050   |
| 2000  | 24784  | 12818  | 6066   | 5900   |
| 2005  | 19886  | 8878   | 5429   | 5579   |
| 2010  | 20464  | 10543  | 5544   | 4377   |
| Total | 501940 | 192920 | 159056 | 149964 |

- 6) Geben Sie in Prozent an, wie sich die keinem Ursprungsland zuzuordnenden Migranten auf hoch-, mittel- oder niedrigqualifizierte Personen verteilen. Erzeugen Sie hierzu folgenden Stata Output:

| year  | tot    | alow     | amed     | ahigh    |
|-------|--------|----------|----------|----------|
| 1980  | 142072 | 37.38808 | 33.75894 | 28.85298 |
| 1985  | 148800 | 30.36021 | 34.6586  | 34.98118 |
| 1990  | 122200 | 43.34043 | 28.82488 | 27.8347  |
| 1995  | 23734  | 39.71096 | 30.58482 | 29.70422 |
| 2000  | 24784  | 51.71885 | 24.47547 | 23.80568 |
| 2005  | 19886  | 44.64447 | 27.30061 | 28.05491 |
| 2010  | 20464  | 51.51974 | 27.09148 | 21.38878 |
| Total | 501940 | 298.6828 | 206.6948 | 194.6225 |

- 7) (10P) Sie erkennen, dass die Anzahl der nicht zuzuordnenden Migranten im Jahr 1995 stark abnimmt und, dass der Anteil der Niedrigqualifizierten ab 2000 sehr hoch ist. Veranschaulichen Sie diese Einsicht durch ein Liniendiagramm und speichern Sie dieses als `unknown.pdf` ab. Die Grafik sollte wie folgt aussehen:



- 8) (20P) Hier (<https://t1p.de/stata-files> — Passwort: happystata) finden Sie eine Datei mit dem Titel `data_gdp_dist.dta`. Diese Datei beinhaltet einen Paneldatensatz von Ländern und Angaben bezüglich des GDP pro Kopf und der Luftdistanz von Deutschland zu den Ursprungsländern in Kilometer.

- a) Mergen Sie diesen Datensatz mit dem IAB Brain-Drain Daten

(Hinweis: Sollten Sie diesen Schritt nicht durchführen können, besteht die Möglichkeit mit dem Datensatz `aftermerge.dta` fortzufahren. Dieser ist hier (<https://t1p.de/stata-files> — Passwort: happystata) zu finden.)

- b) Untersuchen Sie im Rahmen einer Regressionsanalyse, wie der Wohlstand des Ursprungslandes die Migration nach Deutschland befördert. Regressieren Sie hierzu folgende Schätzgleichungen:

$$tot_{it} = \alpha_0 + \alpha_1 gdp\_per\_cap_{it} + \alpha_2 gdp\_per\_cap_{it}^2 + \alpha_3 distw_{it} + \epsilon_{it}, \quad (1)$$

$$low_{it} = \alpha_0 + \alpha_1 gdp\_per\_cap_{it} + \alpha_2 gdp\_per\_cap_{it}^2 + \alpha_3 distw_{it} + \epsilon_{it}, \quad (2)$$

$$med_{it} = \alpha_0 + \alpha_1 gdp\_per\_cap_{it} + \alpha_2 gdp\_per\_cap_{it}^2 + \alpha_3 distw_{it} + \epsilon_{it}, \quad (3)$$

$$high_{it} = \alpha_0 + \alpha_1 gdp\_per\_cap_{it} + \alpha_2 gdp\_per\_cap_{it}^2 + \alpha_3 distw_{it} + \epsilon_{it}, \quad (4)$$

wobei  $\epsilon_{it}$  den Fehlerterm darstellt und die Variablen wie in dem Datensatz betitelt sind.

- c) Interpretieren Sie die Schätzergebnisse kurz.

- 9) **Bonusaufgabe (5P):** Sie glauben, dass zeitspezifische fixe Effekte ( $FE_t$ ) eine Rolle spielen. Integrieren Sie hierzu Dummies für jeden Zeitpunkt in die Regressionsanalyse aus Aufgabe 8b).
- 10) **Bonusaufgabe (5P):** Laden Sie den Brain-Grain Datensatz erneut in Stata ein und mergen Sie diesen mit den `data_gdp_dist.dta` Daten. Regressieren Sie die Gleichungen aus Aufgabe 8b) erneut unter Verwendung einer Schleife, jeweils für Frauen und Männer getrennt.

## Literatur

- Acock, Alan C. 2018. *A Gentle Introduction to Stata*. Stata Press, 6 ed.
- Adkins, Lee C. and R. Carter Hill. 2011. *Using Stata for Principles of Econometrics*. Wiley.
- Baum, Christopher F. 2016. *An Introduction to Stata Programming*. Stata Press, second edition ed.
- Bischof, Daniel and C Zurich. 2017. “New Graphic Schemes for Stata: Plotplain and Plottig.” *The Stata Journal* 17 (3):748–759.
- Cameron, A. Colin and Pravin K. Trivedi. 2010. *Microeconometrics Using Stata*, vol. 5. Stata Press, revised ed.
- Hamilton, Lawrence C. 2013. *Statistics with Stata: Version 12*. Cengage, 8 ed.
- Huber, Stephan. 2017a. “EXPY: Stata Module to Calculate the EXPY-Index as Proposed by Hausmann et al. (2007).” Statistical Software Components S458328, Boston College Department of Economics.
- . 2017b. “Indicators of Product Sophistication and Factor Intensities: Measurement Matters.” *Journal of Economic and Social Measurement* 42:27–65.
- . 2017c. “PRODY: Stata Module to Calculate Factor Intensity and Sophistication Indicators.” Statistical Software Components S458329, Boston College Department of Economics.
- . 2017d. “simcadi: Similarity Indices for Categorical Distributions.” Tech. rep., SSRN. DOI: 10.2139/ssrn.2870834.
- . 2020. “SXPOSE2: Stata Module to Transpose String and Numeric Variable Dataset Including Variable Names and Labels.” Statistical Software Components, Boston College Department of Economics. URL <https://ideas.repec.org/c/boc/bocode/s458854.html>.
- Huber, Stephan and Christoph Rust. 2016. “Calculate travel time and distance with OpenStreetMap data using the Open Source Routing Machine (OSRM).” *The Stata Journal* 16 (2):416–423.
- Kellman, Mitchell and Tim Schroder. 1983. “The export similarity index: Some structural tests.” *Economic Journal* 93 (369):193–98. URL <http://ideas.repec.org/a/ecj/econj1/v93y1983i369p193-98.html>.
- Kohler, Ulrich and Frauke Kreuter. 2016. *Datenanalyse mit Stata: Allgemeine Konzepte der Datenanalyse und ihre praktische Anwendung*. Oldenbourg: De Gruyter, 5. ed.
- Mitchell, Michael N. 2012. *A Visual Guide to Stata Graphics*. Stata Press, 3 ed.
- . 2020. *Data Management Using Stata: A Practical Handbook*. Stata Press, 2 ed.
- . 2021. *Interpreting and Visualizing Regression Models Using Stata*. Stata Press, 2 ed.
- StataCorp. 2015. *STATA USER’S GUIDE*. Stata Press.