

# Stata to R :: CHEAT SHEET



## Introduction

This cheat sheet summarizes common Stata commands for econometric analysis and provides their equivalent expression in R.

References for importing/cleaning data, manipulating variables, and other basic commands include Hanck et al. (2019), [Econometrics with R](#), and Wickham and Grolemund (2017), [R for Data Science](#).

Example data comes from Wooldridge *Introductory Econometrics: A Modern Approach*. Download Stata data sets [here](#). R data sets can be accessed by installing the [wooldridge](#) package from CRAN.

All R commands written in base R, unless otherwise noted.

## Setup

*Note:* While it is common to create a `log` file in Stata to store the commands and output of Stata sessions, the equivalent does not exist in R. A more savvy version in R is to create a [R-markdown](#) file to capture code and output.

```
ssc install outreg2 // install
`outreg2` package. Note: unlike R
packages, Stata packages do not have
to be loaded each time once installed.
```

```
install.packages("wooldridge") # install
`wooldridge` package
```

```
data(package = "wooldridge") # list
datasets in `wooldridge` package
```

```
load(wage1) # load `wage1` dataset into
session
```

```
?wage1 # consult documentation on
`wage1` dataset
```

## Basic plots

*example data: `wage1`*

```
hist(wage) // histogram of `wage`
hist(wage), by(nonwhite) //
scatter(wage educ) // scatter plot
of `wage` by `educ`
tway (scatter wage educ) (lfit
wage educ) // scatter plot with
fitted line
graph box wage, by(nonwhite) //
boxplot of wage by `nonwhite`
```



```
hist(wage1$wage) # histogram of `wage`
```



```
plot(y = wage1$wage, x = wage1$educ) #
scatter plot
abline(lm(wage1$wage~wage1$educ),
col="red") # add fitted line to
scatterplot
```



```
boxplot(wage1$wage~wage1$nonwhite) #
boxplot of `wage` by `nonwhite`
```

## Summarize Data

*example data: `wage1`*

Where Stata only allows one to work with one data set at a time, multiple data sets can be loaded into the R environment simultaneously, and hence must be specified with each function call. *Note:* R does not have an equivalent to Stata's `codebook` command.

```
browse // open browser for loaded data
```

```
describe // describe structure of
loaded data
summarize // display summary
statistics for all variables in
dataset
list in 1/6 // display first 6 rows
```

```
tabulate educ // tabulate `educ`
variable frequencies
tabulate educ female // cross-tabulate
`educ` and `female` frequencies
```



```
view(wage1) # open browser for loaded
`wage1` data
```

```
str(wage1) # describe structure of
`wage1` data
summary(wage1) # display summary
statistics for `wage1` variables
head(wage1) # display first 6 (default)
rows data
tail(wage1) # display last 6 rows
```

```
table(wage1$educ) #tabulate `educ`
frequencies
table("yrs_educ" = wage1$educ, "female" =
wage1$female) # tabulate `educ`
frequencies name table columns
```

*Tip:* The {AER} package will automatically load other useful dependent packages, including: {car}, {lmtest}, {sandwich} which are used for many of the commands listed in this cheat sheet.

## Estimate Models, 1/2

### OLS

*example data: `wage1`*

```
reg wage educ // simple regression
of `wage` by `educ` (Results
printed automatically).
```

```
reg wage educ if nonwhite==1 //
add condition with if statement
```

```
reg wage educ exper, robust //
multiple regression using HCl
robust standard errors
reg wage educ exper,
cluster(numdep) // use clustered
standard errors
```

*Tip:* An alternate way to compute robust standard errors in R for any models not covered by {estimatr} package is load the {AER} package and run:

```
coefest(mod1, vcov. = vcovHC,
type = "HCl")
```

### MLE (Logit/Probit/Tobit)

*example data: `mroz`*

```
logit inlf nwifeinc educ //
estimate logistic regression
```

```
probit inlf nwifeinc educ //
estimate logistic regression
```

```
tobit hours nwifeinc educ, ll(0)
// estimate tobit regression,
lower-limit of y censored at zero
```

## Postestimation, 1/2

*example data: `wage1`*

*Note:* Postestimation commands in Stata apply to the most recently run estimation commands.

```
reg wage educ // estimation used
for the following post-estimation
commands
predict yhat // get predicted
values from last estimation, store
as `yhat`
predict e, res // get residuals
from last estimation, store as `e`
```

```
mod1 <- lm(wage ~ educ, data =
wage1) # simple regression of
`wage` by `educ`, store results in
`mod1`
summary(mod1) # print summary of
`mod1` results
```

```
mod2 <- lm(wage ~ educ, data =
wage1[wage1$nonwhite==1, ]) # add
condition with if statement
mod3 <- estimatr::lm_robust(wage ~
educ + exper, data = wage1, se_type
= "stata") # multiple regression
with HCl (Stata default) robust
standard errors, use {estimatr}
package
mod4 <- estimatr::lm_robust(wage ~
educ + exper, dat = wage1, clusters
= numdep) # use clustered standard
errors.
```

```
mod_log <- glm(inlf~nwifeinc + educ
+ family=binomial(link="logit"),
data=mroz) # estimate logistic
regression
```

```
mod_pro <- glm(inlf~nwifeinc + educ
+ family=binomial(link="probit"),
data=mroz) # estimate logistic
regression
```

```
mod_tob <- AER::tobit(hours ~
nwifeinc + educ, left = 0, data =
mroz) # estimate tobit regression,
lower-limit of y censored at zero,
use {AER} package
```

```
mod1 <- lm(wage ~ educ, data =
wage1) # estimation used for the
following post-estimation commands
yhat <- predict(mod1) # get
predicted values
```

```
e <- residuals(mod1) # get residual
values
```



# Stata to R :: CHEAT SHEET



## Create/Edit Variables example data: `wage1`

*Note:* where Stata only allows one to work with one data set at a time, multiple data sets can be loaded into the R environment simultaneously, hence the data set must be specified for each command.

```
gen exper2 = exper^2 // create
`exper` squared variable
egen wage_avg = mean(wage) // create
average wage variable
```

```
drop tenursq // drop `tenursq`
variable
```

```
keep wage educ exper nonwhite // keep
selected variables
```

```
tab numdep, gen(numdep) // create
dummy variables for `numdep`
```

```
recode exper (1/20 = 1 "1 to 20
years") (21/40 = 2 "21 to 40 years")
(41/max = 3 "41+ years"),
gen(experlv1) // recode `exper` and
gen new variable
```

```
wage1$exper2 <- wage1$exper^2 #
create `exper` squared variable
wage1$wage_avg <- mean(wage1$wage) #
create average wage variable
```

```
wage1$tenursq <- NULL #drop `tenursq`
```

```
wage1 <- wage1[, c("wage", "educ",
"exper", "nonwhite")] # keep selected
variables
```

```
wage1 <-
fastDummies::dummy_cols(wage1,
select_columns = "numdep") # create
dummy variables for `numdep`, use
{fastDummies} package
```

```
wage1$experlv1 <- 3 # recode `exper`
wage1$experlv1[wage1$exper < 41] <- 2
wage1$experlv1[wage1$exper < 21] <- 1
```

## Statistical tests / diagnostics

```
reg lwage educ exper // estimation
used for examples below
estat hettest // Breusch-Pagan /
Cook-Weisberg test for
heteroskedasticity
estat ovtest // Ramsey RESET test
for omitted variables
ttest wage, by(nonwhite) //
independent group t-test, compare
means of same variable between
groups
```

example data: `wage1`

```
mod <- lm(lwage ~ educ exper, data =
wage1) # estimate used for examples
below
lmtest::bptest(mod) # Breusch-Pagan
/ Cook-Weisberg test for hetero-
skedasticity using the {lmtest}
package
lmtest::resettest(mod) # Ramsey
RESET test
t.test(wage ~ nonwhite, data =
wage1) # independent group t-test
```

## Interactions, categorical/continuous variables

example data: `wage1`

In Stata, it is common to use special operators to specify the treatment of variables as continuous (``c.``) or categorical (``i.``). Similarly, the ``#`` operator denotes different ways to return the interaction of those variables. Here we show some common uses of these operators as well as their R equivalents.

```
reg lwage i.numdep // treat
`numdep` as a factor variable
reg lwage c.educ#c.exper // return
full factorial specification
reg lwage c.educ##c.exper // return
full factorial specification
reg lwage c.exper##i.numdep //
return full factorial specification
```

```
lm(lwage ~ as.factor(numdep), data
= wage1) # estimate used for
examples below
lm(lwage ~ educ:exper, data =
wage1) # ...
lm(lwage ~ educ*exper, data =
wage1) # ...
lm(wage ~ exper*as.factor(numdep),
data = wage1) # Ramsey RESET test
```

## Estimate Models, 2/2

### Panel/Longitudinal

example data: `murder`

```
xtset id year // set `id` as
entities (panel) and `year` as
time variable
xtdescribe // describe pattern of
xt data
xtsum // summarize xt data
xtreg mrd rte unem, fe // fixed
effects regression
```

```
plm::plm(murder$id,
murder$year) # check panel balance
with {plm} package
modfe <- plm::plm(mrd rte ~ unem,
index = c("id", "year"), model =
"within", data = murder) # estimate
fixed effects ("within") model
summary(modfe) # display results
```

### Instrumental Variables (2SLS)

example data: `mroz`

```
ivreg lwage (educ = fatheduc),
first // show results of first
stage regression
etest first // test IV and endog.
variable
ivreg lwage(educ = fatheduc) //
show results of 2SLS directly
```

```
modiv <- AER::ivreg(lwage ~ educ |
fatheduc, data = mroz) # estimate
2SLS with {AER} package
summary(modiv, diagnostics = TRUE)
# get diagnostic tests of IV and
endogenous variable
```

## Post-estimation, 2/2

example data: `wage1`

*Note:* Postestimation commands in Stata apply to the most recently run estimation commands.

```
reg lwage educ exper##exper //
estimation used for following post-
estimation commands
estimates store mod1 // stores in
memory the last estimation results
to `mod1`
```

```
margins // get average predictive
margins
margins, dydx(*) // get average
marginal effects for all variables
marginsplot // plot marginal
effects
```

```
margins, dydx(exper) // average
marginal effects of experience
margins, at(exper=(1(10)51)) //
average predictive margins over
`exper` range at 10-year increments
```

```
estimates use mod1 // loads `mod1`
back into working memory
estimates table mod1 mod2 //
display table with stored
estimation results
```

```
mod1 <- lm(lwage ~ educ + exper +
I(exper^2), data = wage1) # Note: in
R, mathematical expressions inside a
formula call must be isolated with
`I()`
```

```
margins::prediction(mod1) # get
average predictive margins with
{margins} package
m1 <- margins::margins(mod1) # get
average marginal effects for all
variables
plot(m) # plot marginal effects
```

```
summary(m) # get detailed summary of
marginal effects
margins::prediction(mod1, at =
list(exper = seq(1,51,10))) #
predictive margins over `exper` range
at 10-year increments
```

```
stargazer::stargazer(mod1, mod2, type
= "text") # use {stargazer} package,
with `type=text` to display results
within R. Note: `type=` also can be
changed for LaTeX and HTML output.
```

