Stata a R :: CHEAT SHEET

stata ⇔ R

Introducción

Esta cheat sheet resume comandos comunes de Stata para análisis econométrico y su expresión equivalente en R.

Como referencias para importar/limpiar datos, manipular variables y otros comandos básicos están Hanck et al. (2019), *Econometrics with R* y Wickham y Grolemund (2017), *R for Data Science*.

Datos de ejemplo de Wooldridge, *Introductory Econometrics: A Modern Approach*. Datos para Stata aquí. En R, instala el paquete `wooldridge` de CRAN.

Todo el código de R usa R base, salvo indicación contraria.

Preparación

Nota: Aunque en Stata es común crear un archivo `log` para guardar los comandos y resultados de las sesiones de Stata, en R no existe un equivalente. Una alternativa más práctica en R es crear un archivo R-markdown que capture el código y los resultados.

```
ssc install outreg2 // instala `outreg2`. Nota: a diferencia de R, en Stata los paquetes instalados no se cargan cada vez.
```

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

data(package = "wooldridge") # lista los datos de `wooldridge`

data("wage1", package = "wooldridge") # carga `wage1`

?wooldridge::wage1 # ayuda sobre los datos `wage1`
```

Resumir datos

datos de ejemplo: `wage1`

Mientras Stata solo permite trabajar con un conjunto de datos a la vez, en el entorno de R se pueden cargar varios simultáneamente, por lo que hay que indicarlo en cada llamada a una función. *Nota:* R no tiene un equivalente al comando `codebook` de Stata.

```
browse // abre el visor de datos

describe // estructura de los datos
summarize // estadísticos de resumen de todas las variables
list in 1/6 // primeras 6 filas

tabulate educ // frecuencias de `educ`
tabulate educ female // tabla cruzada de `educ` y `female`
```

```
View(wage1) # abre el visor de `wage1`

str(wage1) # estructura de `wage1`
summary(wage1) # estadísticos de resumen de `wage1`

head(wage1) # primeras 6 filas (por defecto)
tail(wage1) # últimas 6 filas

table(wage1$educ) #frecuencias de `educ`
table("yrs_edu" = wage1$educ, "female" = wage1$female) # tabla cruzada de `educ` y `female`, con nombres
```

Consejo: El paquete {AER} carga automáticamente otros paquetes útiles de los que depende, como {car}, {lmtest} y {sandwich}, que se usan en muchos de los comandos de esta cheat sheet.

Gráficos básicos

datos de ejemplo: `wage1`

```
hist wage // histograma de `wage`
hist wage, by(nonwhite)
scatter wage educ // dispersión de `wage` vs. `educ`
twoway (scatter wage educ) (lfit wage educ) // dispersión con recta ajustada
graph box wage, by(nonwhite) // boxplot de wage por `nonwhite`
```



```
hist(wage1$wage) # histograma de `wage`

plot(y = wage1$wage, x = wage1$educ) # dispersión
abline(lm(wage1$wage~wage1$educ), col="red") # añade la recta ajustada

boxplot(wage1$wage~wage1$nonwhite) # boxplot de `wage` por `nonwhite`
```

Estimar modelos, 1/2

MCO *datos de ejemplo:* `wage1`

```
reg wage educ // regresión simple de `wage` sobre `educ` (imprime resultados).

reg wage educ if nonwhite==1 // añade una condición con if

reg wage educ exper, robust // regresión múltiple con errores estándar robustos HC1
reg wage educ exper, cluster(numdep) // errores estándar agrupados
```

Consejo: Otra forma de calcular errores estándar robustos en R para modelos no cubiertos por {estimatr} es cargar el paquete {AER} y ejecutar:

```
coeftest(mod1, vcov. = vcovHC, type = "HC1")
```



```
mod1 <- lm(wage ~ educ, data = wage1) # regresión simple de `wage` sobre `educ`; guarda en `mod1`
summary(mod1) # resumen de resultados de `mod1`

mod2 <- lm(wage ~ educ, data = wage1[wage1$nonwhite==1, ]) # añade una condición (como `if`)
mod3 <- estimatr::lm_robust(wage ~ educ + exper, data = wage1, se_type = "stata") # regresión múltiple con errores estándar robustos HC1 (como Stata), usa {estimatr}
mod4 <- estimatr::lm_robust(wage ~ educ + exper, data = wage1, clusters = numdep) # errores estándar agrupados.
```

MV (Logit/Probit/Tobit) *datos de ejemplo:* `mroz`

```
logit inlf nwifeinc educ // regresión logística

probit inlf nwifeinc educ // regresión probit

tobit hours nwifeinc educ, ll(0) // regresión tobit, y censurada por abajo en cero
```



```
mod_log <- glm(inlf~nwifeinc + educ, family=binomial("logit"), data=mroz) # regresión logística

mod_pro <- glm(inlf~nwifeinc + educ, family=binomial("probit"), data=mroz) # regresión probit

mod_tob <- AER::tobit(hours ~ nwifeinc + educ, left = 0, data = mroz) # regresión tobit, y censurada por abajo en cero; usa {AER}
```

Postestimación, 1/2

datos de ejemplo: `wage1`

Nota: Los comandos de postestimación de Stata se aplican a la última estimación ejecutada.

```
reg wage educ // estimación usada en la postestimación siguiente
predict yhat // valores predichos de la última estimación, en `yhat`
predict e, res // residuos de la última estimación, en `e`
```



```
mod1 <- lm(wage ~ educ, data = wage1) # estimación usada en la postestimación siguiente
yhat <- predict(mod1) # valores predichos

e <- residuals(mod1) # residuos
```



CC BY SA Anthony Nguyen • @anguyen1210 • mentalbreaks.rbind.io • versión 1.0.0 • Actualizado: 2019-10 • Traducido por Anthony Romero-Cerdán • anthonyromero.netlify.app

stata ⇔ R

Crear/editar variables *datos de ejemplo*: `wage1`

Nota: mientras Stata solo permite trabajar con un conjunto de datos a la vez, en el entorno de R se pueden cargar varios simultáneamente, por lo que hay que indicar el conjunto de datos en cada comando.

```
gen exper2 = exper^2 // crea `exper` al cuadrado
egen wage_avg = mean(wage) // crea el salario promedio

drop tenursq // elimina `tenursq`

keep wage educ exper nonwhite // conserva esas variables

tab numdep, gen(numdep) // crea variables dummy de `numdep`

recode exper (1/20 = 1 "1 to 20 years") (21/40 = 2 "21 to 40 years") (41/max = 3 "41+ years"), gen(experlvl) // recodifica `exper` en variable nueva
```

```
wage1$exper2 <- wage1$exper^2 # crea `exper` al cuadrado
wage1$wage_avg <- mean(wage1$wage) # crea el salario promedio

wage1$tenursq <- NULL #elimina `tenursq`

wage1 <- wage1[ , c("wage", "educ", "exper", "nonwhite")] # conserva esas variables

wage1 <- fastDummies::dummy_cols(wage1, select_columns = "numdep") # crea variables dummy de `numdep`; usa {fastDummies}

wage1$experlvl <- 3 # recodifica `exper`
wage1$experlvl[wage1$exper < 41] <- 2
wage1$experlvl[wage1$exper < 21] <- 1
```

Pruebas estadísticas / diagnóstico *datos de ejemplo*: `wage1`

```
reg lwage educ exper // modelo de los ejemplos siguientes
estat hettest // Breusch-Pagan / Cook-Weisberg: heterocedasticidad
estat ovtest // RESET de Ramsey: variables omitidas
ttest wage, by(nonwhite) // prueba t de grupos independientes: compara medias entre grupos
```

```
mod <- lm(lwage ~ educ + exper, data = wage1) # modelo de los ejemplos siguientes
lmtest::bptest(mod) # Breusch-Pagan / Cook-Weisberg: heterocedasticidad, con {lmtest}
lmtest::resettest(mod) # RESET de Ramsey
t.test(wage ~ nonwhite, data = wage1) # prueba t, grupos independientes
```

Interacciones, variables categóricas/continuas *datos de ejemplo*: `wage1`

En Stata es común usar operadores especiales para indicar si las variables son continuas (`c.`) o categóricas (`i.`). Del mismo modo, el operador `#` indica distintas formas de obtener la interacción de esas variables. Aquí mostramos algunos usos comunes de estos operadores y sus equivalentes en R.

```
reg lwage i.numdep // `numdep` como variable de factor
reg lwage c.educ#c.exper // solo el término de interacción
reg lwage c.educ##c.exper // especificación factorial completa
reg lwage c.exper##i.numdep // completa, interacción continua × categórica
```

```
lm(lwage ~ as.factor(numdep), data = wage1) # `numdep` como factor
lm(lwage ~ educ:exper, data = wage1) # solo el término de interacción
lm(lwage ~ educ*exper, data = wage1) # especificación factorial completa
lm(wage ~ exper*as.factor(numdep), data = wage1) # completa, interacción continua × categórica
```

Estimar modelos, 2/2

Panel/longitudinal *datos de ejemplo*: `murder`

```
xtset id year // `id` como unidades (panel) y `year` como tiempo
xtdescribe // patrón de los datos xt
xtsum // resume los datos xt
xtreg mrdrte unem, fe // regresión de efectos fijos
```

```
plm::is.pbalanced(murder$id, murder$year) # revisa el balance del panel con {plm}
modfe <- plm::plm(mrdrte ~ unem, index = c("id", "year"),model = "within", data = murder) # modelo de efectos fijos (“within”)
summary(modfe) # muestra resultados
```

Variables instrumentales (MC2E) *datos de ejemplo*: `mroz`

```
ivregress 2sls lwage (educ = fatheduc), first // con 1.ª etapa
estat firststage // relevancia del instrumento
ivregress 2sls lwage (educ = fatheduc) // MC2E directo
```

```
modiv <-AER::ivreg(lwage ~ educ | fatheduc, data = mroz) # estima MC2E con {AER}
summary(modiv, diagnostics = TRUE) # pruebas diagnósticas de VI y variable endógena
```

Postestimación, 2/2 *datos de ejemplo*: `wage1`

Nota: Los comandos de postestimación de Stata se aplican a la última estimación ejecutada.

```
reg lwage educ c.exper##c.exper // estimación usada en la postestimación siguiente
estimates store mod1 // guarda la última estimación en memoria como `mod1`

margins // márgenes predictivos promedio
margins, dydx(*) // efectos marginales promedio de cada variable
marginsplot // grafica efectos marginales

margins, dydx(exper) // efectos marginales promedio de exper
margins, at(exper=(1(10)51)) // márgenes predictivos promedio en el rango de `exper`, cada 10 años

estimates restore mod1 // vuelve a cargar `mod1` en memoria
estimates table mod1 mod2 // tabla con los resultados guardados
```

```
mod1 <- lm(lwage ~ educ + exper + I(exper^2), data = wage1) # Nota: en R, las expresiones matemáticas dentro de la llamada a una fórmula deben aislarse con `I()`

prediction::prediction(mod1) # obtiene los márgenes predictivos promedio con el paquete {prediction}
m1 <- margins::margins(mod1) # efectos marginales promedio de cada variable
plot(m1) # grafica efectos marginales

summary(m1) # resumen detallado de efectos marginales
prediction::prediction(mod1, at = list(exper = seq(1,51,10))) # márgenes predictivos en el rango de `exper`, cada 10 años

stargazer::stargazer(mod1, mod2, type = "text") # usa el paquete {stargazer} y `type=text` para verlos en R. *Nota*: `type= ` admite también LaTeX y HTML.
```

CC BY SA Anthony Nguyen • @anguyen1210 • mentalbreaks.rbind.io • versión 1.0.0 • Actualizado: 2019-10 • Traducido por Anthony Romero-Cerdán • anthonyromero.netlify.app