

Introduction to R

Adrian Rohit Dass
Institute of Health Policy, Management, and
Evaluation

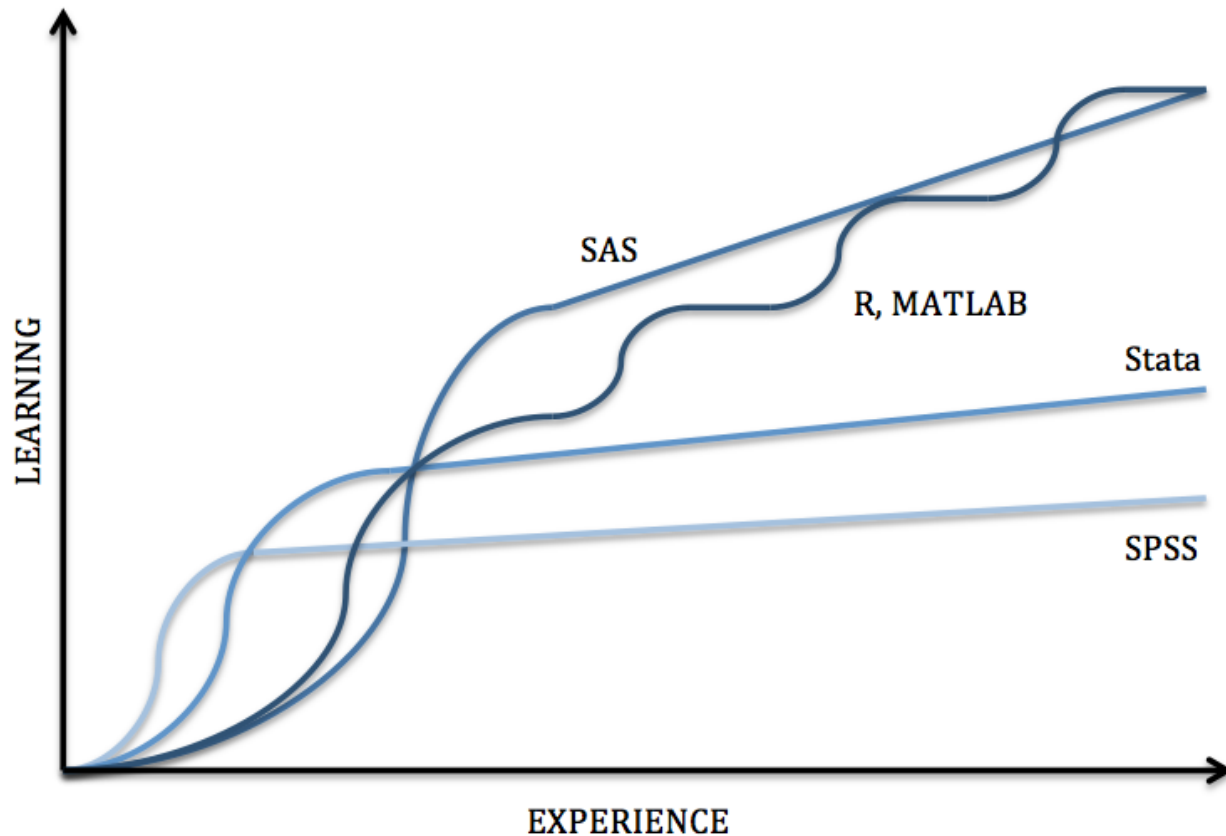
Canadian Centre for Health Economics

October 2, 2015

Outline

- Why use R?
- Reading/Cleaning data
- Ordinary Least Squares (OLS)
- Binary Outcomes
- Instrumental Variable
- Panel Data Econometrics
- Linear & Generalized Linear Mixed-Effects Models
- R Resources

Learning Curves of Various Software Packages



Source: <https://sites.google.com/a/nyu.edu/statistical-software-guide/summary>

Summary of Various Statistical Software Packages

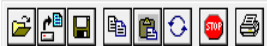
Software	Interface*	Learning Curve	Data Manipulation	Statistical Analysis	Graphics	Specialties
<i>SPSS</i>	Menus & Syntax	Gradual	Moderate	Moderate Scope Low Versatility	Good	Custom Tables, ANOVA & Multivariate Analysis
<i>Stata</i>	Menus & Syntax	Moderate	Strong	Broad Scope Medium Versatility	Good	Panel Data, Survey Data Analysis & Multiple Imputation
<i>SAS</i>	Syntax	Steep	Very Strong	Very Broad Scope High Versatility	Very Good	Large Datasets, Reporting, Password Encryption & Components for Specific Fields
<i>R</i>	Syntax	Steep	Very Strong	Very Broad Scope High Versatility	Excellent	Packages for Graphics, Web Scraping, Machine Learning & Predictive Modeling
<i>MATLAB</i>	Syntax	Steep	Very Strong	Limited Scope High Versatility	Excellent	Simulations, Multidimensional Data, Image & Signal Processing

* The primary interface is bolded in the case of multiple interface types available.

Source: <https://sites.google.com/a/nyu.edu/statistical-software-guide/summary>

Capability of R with Different Types of Data

- Feature rich software for analyzing various types of data:
 - Cross-sectional data: A collection of individuals (persons, countries, etc.) in one time period. Quite common form of micro-data, like surveys.
 - Time Series Data: Many points in time, but for one individual entity. Usually in aggregated form, like rates or percentages.
 - Panel Data: Combination of cross-sectional and time series data. Ex: survey of the same individuals over many years, or aggregate data on murder rates for each province in Canada over many years.
- R can handle large datasets, and has routines for analyzing virtually any type of data



R Console

```
R version 3.2.1 (2015-06-18) -- "World-Famous Astronaut"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Untitled - R Editor

Programming Language

- Programming language in R is *object oriented*
 - This means that data, variables, vectors, matrices, characters, arrays, etc. are treated as “objects” that are created throughout the analysis and stored by name
 - Case sensitive (like most statistical software packages), so be careful

Packages in R

- Functions in R are stored in *packages*
 - For example, the command for OLS (lm) is accessed via the “**stats**” package, which is available in R by default
 - Only when a package is *loaded* will its contents be available. The full list of packages is not loaded by default for computational efficiency
 - Most routines in R are not installed (and thus loaded) by default, meaning that we will have to install packages that we will need beforehand, and then load them later on

Packages in R (Continued)

- To install a package in R:
 - Type `install.packages("packagename")` in command window
 - For example, the package for panel data econometrics is `plm` in R. So, to install the plm package, I would type `install.packages("plm")`
 - Some packages will draw upon functions in other packages, so those packages will need to be installed as well. By using `install.packages(" ")`, it will automatically install dependent packages
- To load a package, type `require(packagename)`
 - Ex: To load the plm package, I would type `require(plm)` before running any routines that require this package

Reading Data into R

What format is the data in?

- Data from Comma Separated Values File (.csv)
 - Package: `utils`
 - Formula: `read.csv(file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE, comment.char = "", ...)`
- Data from Excel File (.xlsx)
 - Package: `xlsx`
 - Formula: `read.xlsx(file, sheetIndex, sheetName=NULL, rowIndex=NULL, startRow=NULL, endRow=NULL, colIndex=NULL, as.data.frame=TRUE, header=TRUE, colClasses=NA, keepFormulas=FALSE, encoding="unknown", ...)`
- Data from STATA (.dta)
 - Package: `foreign`
 - `read.dta(file, convert.dates = TRUE, convert.factors = TRUE, missing.type = FALSE, convert.underscore = FALSE, warn.missing.labels = TRUE)`

Other Formats: See package “foreign”

<https://cran.r-project.org/web/packages/foreign/foreign.pdf>

Reading Data into R

Examples:

- CSV file with variable names at top
 - `data = read.csv("C:/Users/adrianrohitdass/Documents/R Tutorial/data.csv")`
- CSV file with no variable names at top
 - `data = read.csv("C:/Users/adrianrohitdass/Documents/R Tutorial/data.csv", header=F)`
- STATA data file
 - `require(foreign)`
 - `data = read.dta("C:/Users/adrianrohitdass/Documents/R Tutorial/data.dta")`

Referring to Variables in a Dataset

- Suppose I had data stored in “mydata” (i.e an object created to store the data read-in from a .csv by R). To refer to a specific variable in the dataset, I would type

mydata\$varname

Name of Dataset

Name of Variable in dataset

Creating a new variable/object

- No specific command to generate new variables (in contrast to STATA's "gen" and "egen" commands) but may involve a routine depending on what's being generated
 - `x = 5` generates a 1x1 scalar called "x"
 - `data$age = year - dob` creates a new variable "age" in the dataset "data" that is equal to the year – the person's date of birth (let's say in years)

Looking at Data

- Display the first or last few entries of a dataset:
 - Package: `utils`
 - First few elements of dataset (default is 5):
 - `head(x, n, ...)`
 - Last few elements of dataset (default is 5):
 - `tail(x, n, ...)`
- List of column names in dataset
 - Package: `base`
 - Formula: `colnames(x)`

Missing Values

Missing Values are listed as “NA” in R

- Count number of NA's in column

```
sum(is.na(x))
```

- Recode Certain Value's as NA (i.e. non responses coded as -1)

```
x[x==-1] = NA
```

Classes in R

- In R, every object has a *class*
 - For example, character variables are given the class of **factor** or **character**, whereas numeric variables are **integer**
- Classes determine how objects are handled by generic functions
 - For example, the `mean(x)` function will work for integers but not for factors - which generally makes sense for categorical (with more than two categories) and ordinal variables)
 - In a regression, it makes no difference whether a variable is coded as a factor or character. R will automatically treat the latter as a factor (removing one category for reference) – but you will get a warning message saying that the character was converted to a factor
 - For database management (i.e. merging different datasets together) it's usually easier to work with **characters** than **factors**

Factors in R

- If the factor is coded numerically, this may cause unwanted behaviour in a regression
 - R will treat the factor as a continuous measure, which generally does not make sense for categorical responses.
- To create a factor variable from a numerically coded categorical/ordinal variable

```
data$race <- factor(data$race, levels = c(1,2,3,4), labels =  
c("Black", "Hispanic", "Mixed Race (Non-Hispanic)",  
"Non-Black / Non-Hispanic"))
```

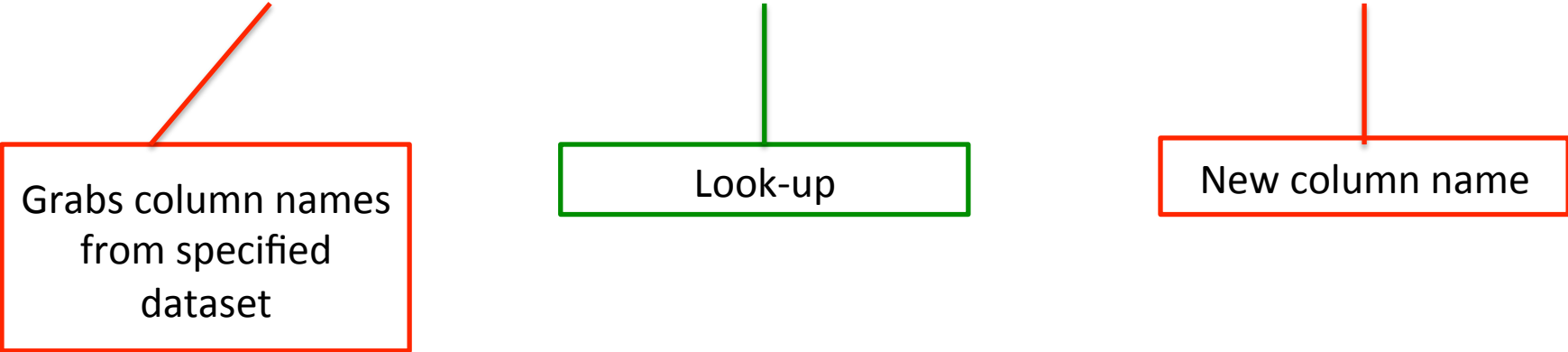
- To choose the reference category

```
contrasts(data$race) = contr.treatment(4,base=4)
```

Renaming Variables (Columns)

- A few different ways to do this, but I prefer this method:

```
colnames(data)[which(colnames(data) == 'R1482600')] = 'race'
```



Grabs column names
from specified
dataset

Look-up

New column name

Descriptive Statistics in R

- Mean
 - Package: `base`
 - Formula: `mean(x, trim = 0, na.rm = FALSE, ...)`
- Standard Deviation
 - Package: `stats`
 - Formula: `sd(x, na.rm = FALSE)`
- Correlation
 - Package: `stats`
 - Formula: `cor(x, y = NULL, use = "everything", method = c("pearson", "kendall", "spearman"))`

Tabulations R

- Tabulations of categorical/ordinal variables can be done with R's *table* command:
 - Package: **base**
 - Formula: **table(..., exclude = if (useNA == "no") c(NA, NaN), useNA = c("no", "ifany", "always"), dnn = list.names(...), deparse.level = 1)**

Ex: Table Sex Variable, with extra column for missing values (if any)

```
> mytable = table(pdata$sex, exclude=NULL)
> mytable
```

Female	Male	<NA>
17540	18396	0

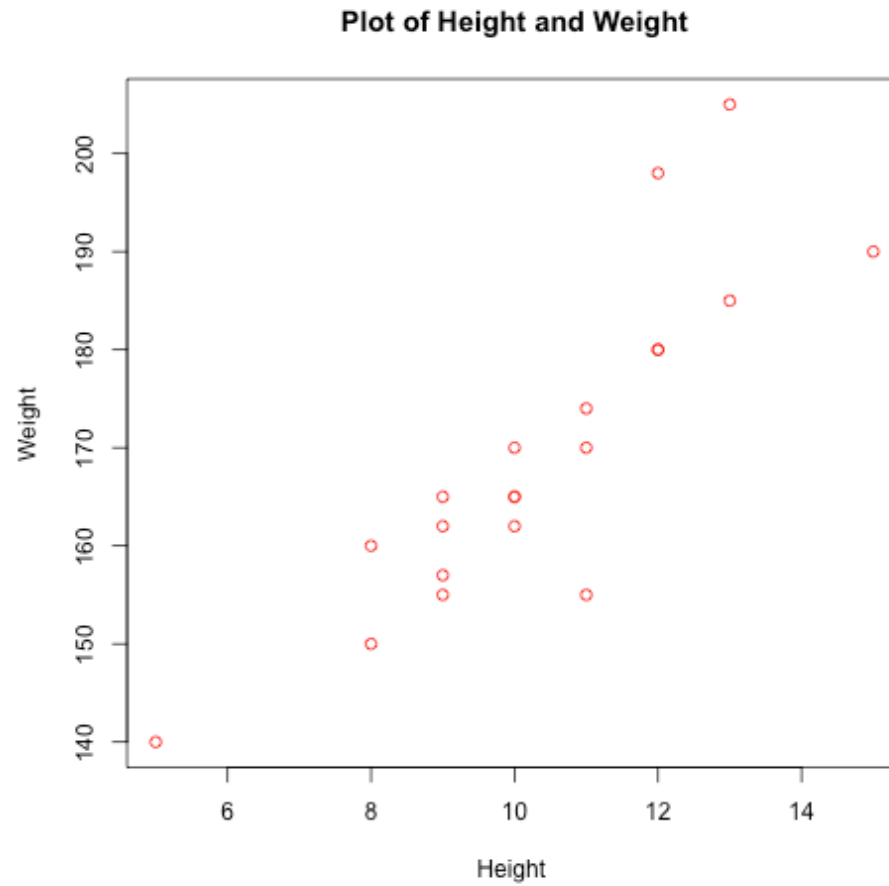
Graphing Data in R

- Basic plot in R
 - Package: `graphics`
 - Formula: `plot(x, y, ...)`

Example:

```
plot(lmdata$height, lmdata$weight, main = "Plot of Height and  
Weight", xlab="Height ", ylab="Weight", col=2)
```

Resulting Graph



Ordinary Least Squares

- The estimator of the regression intercept and slope(s) that minimizes the sum of squared residuals (Stock and Watson, 2007).
 - Package: `stats`
 - Formula: `lm(formula, data, subset, weights, na.action, method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE, singular.ok = TRUE, contrasts = NULL, offset, ...)`

Examples:

Regression of “height” on “weight” using dataset “lmdata”

```
ols = lm(weight~height + sex, data=lmdata)
```

Online Help File

<https://stat.ethz.ch/R-manual/R-devel/library/stats/html/lm.html>

Ordinary Least Squares

Example: OLS regression of Height on Weight (Univariate Analysis)

```
> #Linear Regression Model Example
> lmdata = read.csv("/Users/adrianrohitdass/Google Drive/Introduction to R/Examples Using
Data/ex1.csv")
> ols = lm(weight~height, data = lmdata)
> summary(ols)
```

Call:

```
lm(formula = weight ~ height, data = lmdata)
```

Residuals:

Min	1Q	Median	3Q	Max
-18.5451	-3.9467	-0.6108	3.1763	18.7007

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	103.3971	9.3421	11.068	1.83e-09	***
height	6.3771	0.8837	7.216	1.03e-06	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.502 on 18 degrees of freedom

Multiple R-squared: 0.7431, Adjusted R-squared: 0.7289

F-statistic: 52.07 on 1 and 18 DF, p-value: 1.031e-06

Post-Estimation

Package: **lmtest**

- Breusch-Pagan test against heteroskedasticity.

bptest(formula, varformula = NULL, studentize = TRUE, data = list())

- Ramsey's RESET test for functional form.

resettest(formula, power = 2:3, type = c("fitted", "regressor", "princomp"), data = list())

- Variance Inflation Factor (VIF)

Package: **car**

vif(model)

Exporting Regression Results

Outputting regression results to look at, modify, or insert into document.

Package: **estout**

Formula: **esttab(t.value = FALSE, p.value = FALSE, round.dec = 3, caption = NULL, label = NULL, texfontsize = NULL, sig.levels = c(0.1, 0.05, 0.01), sig.sym=c("*", "**", "***"), filename=NULL, csv=FALSE, dcolumn=NULL, table="table", table.pos="htbp", caption.top=FALSE, booktabs=FALSE, var.order=NULL, sub.sections=NULL, var.rename=NULL, resizebox=c(0,0), colnumber=FALSE, store="default")**

NB: This package was designed for LaTeX, but works with excel as well. To export to a .csv, be sure to have `csv = T`

Example:

esttab(filename = "/Users/adrianrohitdass/Google Drive/Introduction to R/Examples Using Data/out", csv=T)

Models for Binary Outcomes

- R does not come with different programs for binary outcomes. Instead, it utilizes a unifying framework of generalized linear models (GLMs) and a single fitting function, `glm()` (Kleiber & Zeileis (2008))

Package: `stats`

Formula: `glm(formula, family = gaussian, data, weights, subset, na.action, start = NULL, etastart, mustart, offset, control = list(...), model = TRUE, method = "glm.fit", x = FALSE, y = TRUE, contrasts = NULL, ...)`

- For binary outcomes, we specify `family="binomial"` and `link= "logit"` or `"probit"`
- Can be extended to count data as well (`family="poisson"`)

Online help: <http://www.inside-r.org/r-doc/stats/glm>

Models for Binary Outcomes

Example: The effect of age and sex on odds of arrest

```
> logit = glm(arrestbin~age + male, data = subdata, family="binomial"(link="logit"))
> summary(logit)
```

```
Call:
glm(formula = arrestbin ~ age + male, family = binomial(link = "logit"),
    data = subdata)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-0.5175	-0.4493	-0.3310	-0.2657	2.6483

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-5.57304	0.51595	-10.801	< 2e-16 ***
age	0.14977	0.03136	4.775	1.79e-06 ***
male	0.93428	0.09460	9.876	< 2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 4214.0 on 8360 degrees of freedom
Residual deviance: 4085.9 on 8358 degrees of freedom
(623 observations deleted due to missingness)
AIC: 4091.9

Number of Fisher Scoring iterations: 6

Instrumental Variables

A way to obtain a consistent estimator of the unknown coefficients of the population regression function when the regressor, X , is correlated with the error term, u . (Stock and Watson, 2007).

Package: **AER**

Formula: **ivreg(formula, instruments, data, subset, na.action, weights, offset, contrasts = NULL, model = TRUE, y = TRUE, x = FALSE, ...)**

Online documentation: <https://cran.r-project.org/web/packages/AER/AER.pdf>

IV Example

Example: Determinants of Income (As a function of Health)

```
> require(AER)
> iv = ivreg(Income ~ Health + Age | ParentHealth + Age)
> summary(iv, diagnostics = TRUE)
```

```
Call:
ivreg(formula = Income ~ Health + Age | ParentHealth + Age)
```

```
Residuals:
    Min       1Q   Median       3Q      Max
-3.1557 -0.6261  0.0130  0.6495  2.8700
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.03965     0.06817   29.92  <2e-16 ***
Health       0.99773     0.01186   84.16  <2e-16 ***
Age          2.00177     0.07256   27.59  <2e-16 ***
```

```
Diagnostic tests:
              df1 df2 statistic p-value
Weak instruments  1 997      1427  <2e-16 ***
Wu-Hausman       1 996      2271  <2e-16 ***
Sargan           0  NA         NA      NA
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.9742 on 997 degrees of freedom
Multiple R-Squared: 0.9573, Adjusted R-squared: 0.9572
Wald test: 1.067e+04 on 2 and 997 DF, p-value: < 2.2e-16
```

Prints out F-test for Weak Instruments, Hausman Test Statistic (vs ols) and Sargan's Test for Over-identifying Restrictions (if more than one instrument use)

Panel Data Econometrics

Panel Data: Data for multiple entities where each entity is observed in two or more time period (Stock and Watson, 2008).

Package: **plm**

Specifying Panel Data Data Object in R: **pdata = pdata.frame(mydata, , index = c("ID", "year"))**

General formula for panel routines in R

plm(formula, data, subset, na.action, effect = c("individual", "time", "twoways"), model = c("within", "random", "ht", "between", "pooling", "fd"), random.method = c("swar", "walhus", "amemiya", "nerlove", "kinla"), inst.method = c("bvk", "baltagi"), restrict.matrix = NULL, restrict.rhs = NULL, index = NULL, ...)

Online Documentation:

<https://cran.r-project.org/web/packages/plm/vignettes/plm.pdf>

Panel Data Econometrics

Fixed Effects (Within) Estimator

A panel data regression that includes entity fixed effects. Only utilizes the variation “within” individuals over time (Stock and Watson, 2008).

Fixed Effects Estimation in R

- Least Squares Dummy Variable (LSDV)

```
lsdv = lm(y~x + factor(id), data=mydata)
```

- Fixed Effects (Within) Estimator

```
fe <- plm(y~x, data = mydata, model = "within")
```


Fixed Effects in R

Example: FE regression of Height and Age on Weight

```
> require(plm)
> data = read.csv("/Users/adrianrohitdass/Google Drive/Introduction to R/Examples Using
Data/expdata.csv")
> pdata = pdata.frame(data, index=c("ID","year"))
> fe = plm(weight~height + age, data=pdata, model="within")
> summary(fe)
Oneway (individual) effect Within Model
```

```
Call:
plm(formula = weight ~ height + age, data = pdata, model = "within")
```

Unbalanced Panel: n=8732, T=1-4, N=31689

```
Residuals :
    Min. 1st Qu.  Median 3rd Qu.    Max.
-151.00   -4.45     0.00    4.44   128.00
```

```
Coefficients :
              Estimate Std. Error t-value Pr(>|t|)
height 23.831974    0.713102   33.42 < 2.2e-16 ***
age      4.110599    0.061879   66.43 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Total Sum of Squares:    4022900
Residual Sum of Squares: 3037100
R-Squared      : 0.24505
Adj. R-Squared : 0.17751
F-statistic: 3725.43 on 2 and 22955 DF, p-value: < 2.22e-16
```

Panel Data Econometrics

Random Effects Estimator

All individual differences are captured by the intercept, but these differences are treated as random *rather* than fixed. Estimates are a weighted average of a “within” and “between” estimate (Allison, 2009). Assumes $\text{corr}(x_i, u) = 0$

Random Effects in R

```
re = plm(y~x, data = mydata, method = “random”)
```

Random Effects in R

Example: RE regression of Height and Age on Weight

```
> re = plm(weight~height + age, data=pdata, model="random")
> summary(re)
Oneway (individual) effect Random Effect Model
(Swamy-Arora's transformation)

Call:
plm(formula = weight ~ height + age, data = pdata, model = "random")

Unbalanced Panel: n=8732, T=1-4, N=31689

Effects:
              var std.dev share
idiosyncratic 132.31  11.50 0.138
individual    825.15  28.73 0.862
theta :
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.6283  0.8037  0.8037  0.7960  0.8037  0.8037

Residuals :
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-151.000  -6.710   -1.700   -0.037   4.700  120.000

Coefficients :
              Estimate Std. Error t-value Pr(>|t|)
(Intercept) -112.923171   3.180832  -35.501 < 2.2e-16 ***
height       35.836939   0.581146  61.666 < 2.2e-16 ***
age          3.767148   0.059822  62.972 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares: 6171900
Residual Sum of Squares: 4315600
R-Squared : 0.3009
Adj. R-Squared : 0.30087
F-statistic: 6814.8 on 2 and 31686 DF, p-value: < 2.22e-16
```

Panel Data Econometrics

Hausman Test Routine to test the assumptions of the RE Estimator: $\text{corr}(\xi_i, u_i) = 0$

phtest(fe, re)

Example: From Previous Models

```
> phtest(fe, re)
```

```
Hausman Test
```

```
data: weight ~ height + age  
chisq = 851.0267, df = 2, p-value < 2.2e-16  
alternative hypothesis: one model is inconsistent
```

Linear Mixed Effects Models

- Mixed-effects models (aka: nested models, multilevel models, random effects models, etc.) allow for analysis of data at more than 1 level. Examples include:
 - Patients (level 1) in hospitals (level 2)
 - Students (level 1) in schools (level 2).
 - Professors (level 1) in departments (level 2) in universities
 - Time (level 1) nested within individuals (level 2) – For longitudinal studies

Difference between plm and lme4 for Longitudinal Data

“...longitudinal data models in nlme and lme4 are estimated by (restricted or unrestricted) maximum likelihood. While under normality, homoskedasticity and no serial correlation of the errors ols are also the maximum likelihood estimator, in all the other cases there are important differences.

The econometric gls approach has closed-form analytical solutions computable by standard linear algebra and, although the latter can sometimes get computationally heavy on the machine, the expressions for the estimators are usually rather simple. ml estimation of longitudinal models, on the contrary, is based on numerical optimization of nonlinear functions without closed-form solutions and is thus dependent on approximations and convergence criteria...”

Croissant & Millo, 2008

Linear Mixed Effects Model

- Package: **lme4**
- Linear Mixed Effects Model Formula:
 - `lmer(formula, data = NULL, REML = TRUE, control = lmerControl(), start = NULL, verbose = 0L, subset, weights, na.action, offset, contrasts = NULL, devFunOnly = FALSE, ...)`
- Generalized Linear Mixed Effects Model
 - `glmer(formula, data = NULL, family = gaussian, control = glmerControl(), start = NULL, verbose = 0L, nAGQ = 1L, subset, weights, na.action, offset, contrasts = NULL, mustart, etastart, devFunOnly = FALSE, ...)`

Online help file:

<https://cran.r-project.org/web/packages/lme4/lme4.pdf>

Linear Mixed Effects Model

Example: Mixed Effects regression of Height and Age on Weight (w/ random intercepts for each individual in sample)

```
> require(lme4)
> lmer(weight~height + age + (1 | ID), data=data)
Linear mixed model fit by REML ['lmerMod']
Formula: weight ~ height + age + (1 | ID)
Data: data
REML criterion at convergence: 272982.3
Random effects:
  Groups      Name      Std.Dev.
  ID          (Intercept) 29.84
  Residual                    11.57
Number of obs: 31689, groups:  ID, 8732
Fixed Effects:
(Intercept)      height      age
    -110.512     35.362     3.781
```


R Resources

R Online Resources

- A list of R packages is contained here:
https://cran.r-project.org/web/packages/available_packages_by_date.html
- By clicking on a particular package, you'll be taken to a page with more details, as well as a link to download the documentation
- Typing `help(topic)` in R pulls up a brief help file with syntax and examples, but the online manuals contain more detail

R Online Resources

UCLA Institute for Digital Research and Education

- List of topics and R resources (getting started, data examples, etc.) can be found here:

<http://www.ats.ucla.edu/stat/r/>

Other R Resources

1. Kleiber, C., & Zeileis, A. (2008). *Applied econometrics with R*. Springer Science & Business Media.
 - Great reference for the applied researcher wanting to use R for econometric analysis. Includes R basics, linear regression model, panel data models, binary outcomes, etc.
2. CRAN Task View: Econometrics
 - A listing of the statistical models used in econometrics, as well as the R package(s) needed to perform them. Available at: <https://cran.r-project.org/view=Econometrics>
3. R Studio
 - A more user-friendly R experience. Includes syntax colouring, window for variables in dataset, window for command syntax, and other nice features. Need to have R already installed. <https://www.rstudio.com>

Thanks for Listening

Good luck with R!