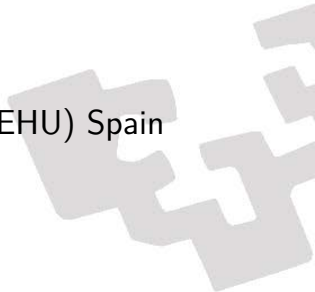


Empezando en R

David Hoyos

Universidad del País Vasco (UPV/EHU) Spain

febrero 17, 2020



¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

Vectores
○○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○

Conjuntos de datos
○○○○

¿Por qué R?

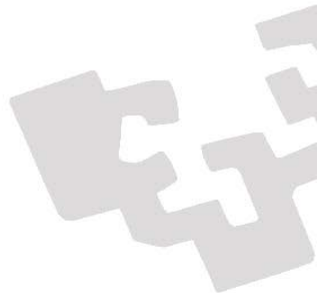
Conceptos básicos

Vectores

Matrices

Funciones y bucles

Conjuntos de datos



¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

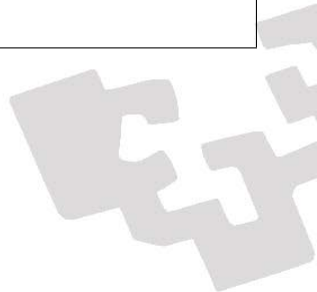
Vectores
○○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○

Conjuntos de datos
○○○○

¿Por qué R?



Software para análisis de datos

Software comercial

- **Nlogit** es una extensión del programa de estadística LIMDEP (variables dependientes limitadas)
- **STATA** es un paquete de estadística y análisis de datos
- **LATENT GOLD** es un programa especializado en clases latentes
- **Lenguajes de programación**: Ox (computación paralela), GAUSS, Mathlab

Software de código abierto:

- **R** (AER package, [[AER package](#)], Applied Econometrics with R)
- **Python-Biogeme**, **Panda-Biogeme**
- **Ox Consola** (gratis para educación e investigación)

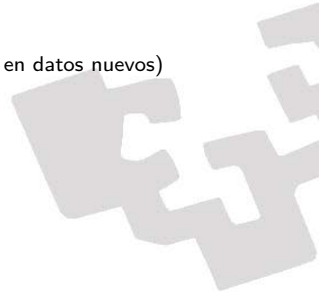
R



- R es un lenguaje y entorno para análisis estadístico y gráficos
- Está disponible como **Software Libre** en forma de código abierto
- Compila y funciona en una gran variedad de plataformas **UNIX** y sistemas similares como **Windows** y **MacOS**

Puntos fuertes (P. Dalgaard, R, Core Team) :

- Expresión compacta de idea, en una línea
- Fácil construcción de estudios de simulación
- Operaciones con objetos de un modelo (e.g. predicción en datos nuevos)
- Sistema gráfico flexible



Programas dirigidos con menús

El problema es que R puede ser difícil de aprender, pero ¿qué pasa con los programas comerciales que funcionan con menús?

Utilizar paquetes de este tipo puede tener también pros y contras.

En general:

Pros

- fácil de estimar modelos muy complejos
- bajo esfuerzo para escribir códigos
- coste de aprendizaje bajo

Contras

- a veces el analista no entiende lo que realmente está sucediendo en el proceso de estimación del modelo (black box)
- el analista sólo puede estimar los modelos que vienen incluidos en el programa

Sean J. Taylor: The Statistics Software Signal

seanjtaylor.com/post/39573264781/the-statistics-software-signal

Apps Google Keep Google logo Calendario Traductor de Google Gmail Contactos Gmail Google h

sean j. taylor

computational social scientist



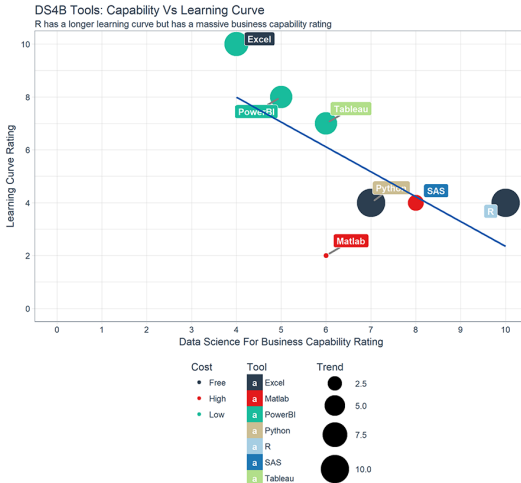
about cv faq facebook
twitter instagram

The Statistics Software Signal

Last night on Twitter, I went on a bit of a rant about statistics packages (namely Stata and SPSS). My point was not that these software packages are bad per se, but that I have found them to be correlated with bad quality science. Here is my theory why.

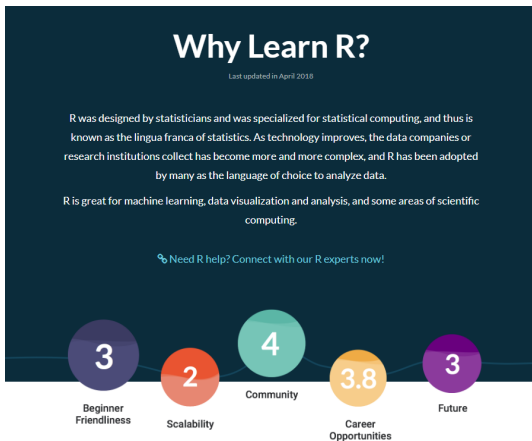
- When you don't have to code your own estimators, you probably won't understand what you're doing.** I'm not saying that you definitely won't, but push-button analyses make it easy to compute numbers that you are not equipped to interpret.
- When it's extremely low cost to perform inference, you are likely to perform a lot of inferences.** When your first regression gives a non-result, you run a second one, and a third one, etc. This leads untrained researchers to run into [multiple comparisons problems](#) and increases the risk of Type I errors.
- When operating software doesn't require a lot of training, users of that software are likely to be poorly trained.** This is an [adverse selection issue](#). Researchers who care about statistics enough should have gravitated toward R at some point. I also trust results produced using R, not because it is better software, but because it is difficult to learn. The software is not causing you to be a better scientist, but better scientists will be using it.
- When you use proprietary software, you are sending the message that you don't care about whether people can replicate your analyses or verify that the code was**

¿Por qué R?



Why R? Tools like Excel, Tableau, PowerBI are easier to learn, but have lower Business Capability. Tools like Python, SAS, and Matlab have high Data Science Capability, but lack the visualization and interactive application tools needed for business. R has the best data science, visualization, and interactive tools plus it's free!

¿Por qué R?



Source: <http://www.bestprogramminglanguagefor.me/why-learn-r>

¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

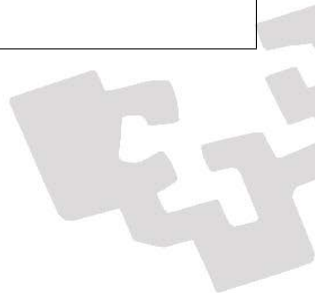
Vectores
○○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○

Conjuntos de datos
○○○○

Conceptos básicos



Conceptos básicos

Eliminar todos los objetos de la memoria

```
rm(list = ls(all = TRUE))
```

Diferentes tipos de variables que vamos a utilizar

```
A <- 10  
B <- "Computer"  
C <- TRUE
```

```
mode(A)
```

```
[1] "numeric"
```

```
mode(B)
```

```
[1] "character"
```

```
mode(C)
```

```
[1] "logical"
```

Conceptos básicos

Es habitual empezar los scripts de R con el espacio de trabajo vacío

```
# Preliminares
rm(list = ls(all = TRUE))
Sys.setenv(LANG = "en")
setwd(".")

# Limpia el espacio de trabajo
# Elegir idioma
# Elegir directorio de trabajo
```

Conceptos básicos

R es sensible a las mayúsculas, “Index” y “index” son dos variables diferentes

```
index <- 1  
Index <- 10
```

```
index
```

```
[1] 1
```

```
Index
```

```
[1] 10
```

Guía de estilo de R de Google

En <https://google.github.io/styleguide/Rguide.xml> puedes encontrar una guía de cómo hacer tus códigos comprensibles y fáciles de leer. La Guía de Estilo de R de Google recomienda, por ejemplo:

<i>Calidad del nombre</i>	<i>Nombre de la variable</i>
Good	avg.clicks
Acceptable	avgClicks
Bad	avg_Clicks

¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

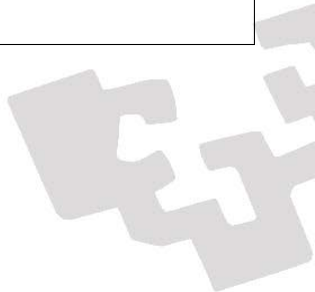
Vectores
○○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○

Conjuntos de datos
○○○○

Vectores



Vectores

Definición de vectores

```
x <- vector(mode = "numeric",  
             length = 4)
```

x

```
[1] 0 0 0 0
```

```
xlog <- vector(mode = "logical",  
               length = 3)
```

xlog

```
[1] FALSE FALSE FALSE
```

Acceso a elementos de un vector

```
x[1] <- 1  
x
```

```
[1] 1 0 0 0
```

```
xlog[3] <- TRUE  
xlog
```

```
[1] FALSE FALSE TRUE
```

Definición alternativa de un vector

Replicar

```
rep(5,3)
```

```
[1] 5 5 5
```

```
rep(10,6)
```

```
[1] 10 10 10 10 10 10
```

Combinar

```
c(1,2,3,4)
```

```
[1] 1 2 3 4
```

```
c(3,5,1,-8,7,6,0)/2
```

```
[1] 1.5 2.5 0.5 -4.0 3.5 3.0 0.0
```

Combinar comandos

```
rep(c(1,2,3,4),3)
```

```
[1] 1 2 3 4 1 2 3 4 1 2 3 4
```

```
rep(c(3,5,1,-8,7,6,0)/2,2)
```

```
[1] 1.5 2.5 0.5 -4.0 3.5 3.0 0.0 1.5 2.5 0.5 -4.0 3.5 3.0 0.0
```

Definición alternativa de un vector: secuencias

Definición de secuencias

```
z1 <- 1:4  
z1
```

```
[1] 1 2 3 4
```

```
z2 <- seq(1:4)  
z2
```

```
[1] 1 2 3 4
```

```
z3 <- seq(2, 8, by=2)  
z3
```

```
[1] 2 4 6 8
```

```
z4 <- seq(0,10,length=7)  
z4
```

```
[1] 0.000000 1.666667 3.333333 5.000000 6.666667 8.333333 10.000000
```

Comparaciones y operadores lógicos

Comparaciones básicas

```
1 == 2
```

```
[1] FALSE
```

```
1 != 2
```

```
[1] TRUE
```

```
1 <= 2
```

```
[1] TRUE
```

```
1 > 2
```

```
[1] FALSE
```

Operadores lógicos básicos

```
(1 == 2) & (1 < 2)
```

```
[1] FALSE
```

```
(1 == 2) | (1 < 2)
```

```
[1] TRUE
```

```
! c((1 == 2), 1 < 2)
```

```
[1] TRUE FALSE
```

```
x <- c(1,2,3,4,5,6)  
x
```

```
[1] 1 2 3 4 5 6
```

Funciones internas básicas I

```
sum(x)
```

```
[1] 21
```

```
prod(x)
```

```
[1] 720
```

```
max(x)
```

```
[1] 6
```

```
min(x)
```

```
[1] 1
```

```
which.max(x)
```

```
[1] 6
```

```
which.min(x)
```

```
[1] 1
```


Funciones internas básicas II

```
x
```

```
[1] 1 2 3 4 5 6
```

```
range(x)
```

```
[1] 1 6
```

```
length(x)
```

```
[1] 6
```

```
mean(x)
```

```
[1] 3.5
```

```
median(x)
```

```
[1] 3.5
```

```
x
```

```
[1] 1 2 3 4 5 6
```

```
var(x)
```

```
[1] 3.5
```

```
cumsum(x)
```

```
[1] 1 3 6 10 15 21
```

```
summary(x)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.00	2.25	3.50	3.50	4.75	6.00

¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

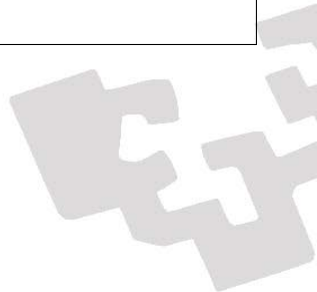
Vectores
○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○

Conjuntos de datos
○○○○

Matrices



Matrices

Cómo definir una matriz

```
m3 <- matrix(data = 1, nrow = 2, ncol = 2)
m3
```

	[,1]	[,2]
[1,]	1	1
[2,]	1	1

Acceso a los elementos de una matriz

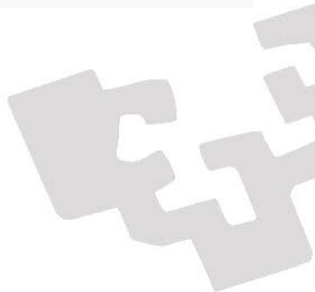
```
m3[2,1] <- 2
m3
```

	[,1]	[,2]
[1,]	1	1
[2,]	2	1

Trasposición de una matriz

`t(m3)`

	[,1]	[,2]
[1,]	1	2
[2,]	1	1



Matrices II

Cómo obtener la diagonal de una matriz

```
m3
```

```
      [,1] [,2]  
[1,]     1     1  
[2,]     2     1
```

```
diag(m3)
```

```
[1] 1 1
```

Cómo obtener la inversa de una matriz

```
m3
```

```
      [,1] [,2]  
[1,]     1     1  
[2,]     2     1
```

```
solve(m3)
```

```
      [,1] [,2]  
[1,]    -1     1  
[2,]     2    -1
```

Matrices III

```
m3
```

```
      [,1] [,2]  
[1,]     1     1  
[2,]     2     1
```

```
solve(m3)
```

```
      [,1] [,2]  
[1,]    -1     1  
[2,]     2    -1
```

Multiplicación de matrices

```
solve(m3) %*% m3
```

```
      [,1] [,2]  
[1,]     1     0  
[2,]     0     1
```

Multiplicación elemento a elemento

```
m3
```

```
      [,1] [,2]  
[1,]     1     1  
[2,]     2     1
```

```
solve(m3) * m3
```

```
      [,1] [,2]  
[1,]    -1     1  
[2,]     4    -1
```


Matrices IV

Cómo rellenar una matriz

```
xmat <- matrix(data = 1:6,  
               nrow = 2,  
               ncol = 3,  
               byrow = FALSE)
```

xmat

	[,1]	[,2]	[,3]
[1,]	1	3	5
[2,]	2	4	6

```
xmat <- matrix(data = 1:6,  
               nrow = 2,  
               ncol = 3,  
               byrow = TRUE)
```

xmat

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6

Matrices V

Cómo acceder a diferentes partes de una matriz

```
xmat
```

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6

Tercera columna

```
xmat[,3]
```

```
[1] 3 6
```

Primera fila

```
xmat[1,]
```

```
[1] 1 2 3
```

Matrices VI

Cómo eliminar diferentes partes una matriz

```
xmat
```

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6

Eliminar la primera fila

```
xmat[-1,]
```

```
[1] 4 5 6
```

Eliminar la primera y segunda columna

```
xmat[, -c(1,2)]
```

```
[1] 3 6
```

Matrices VIIa

Uniando matrices (o conjuntos de datos por filas y columnas)

```
m1 <- matrix(1, nr = 2, nc = 2)
m2 <- matrix(2, nr = 2, nc = 2)
m1
```

```
      [,1] [,2]
[1,]    1    1
[2,]    1    1
```

```
m2
```

```
      [,1] [,2]
[1,]    2    2
[2,]    2    2
```

```
rbind(m1, m2) # Unimos por filas
```

```
      [,1] [,2]
[1,]    1    1
[2,]    1    1
[3,]    2    2
[4,]    2    2
```

Matrices VIIb

Uniendo matrices (o conjuntos de datos por filas y columnas)

```
m1 <- matrix(1, nr = 2, nc = 2)
m2 <- matrix(2, nr = 2, nc = 2)
m1
```

```
      [,1] [,2]
[1,]    1    1
[2,]    1    1
```

```
m2
```

```
      [,1] [,2]
[1,]    2    2
[2,]    2    2
```

```
cbind(m1, m2) # Unimos por columnas
```

```
      [,1] [,2] [,3] [,4]
[1,]    1    1    2    2
[2,]    1    1    2    2
```

¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

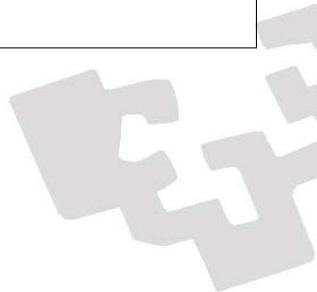
Vectores
○○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○○

Conjuntos de datos
○○○○

Funciones y bucles



Funciones I

Definición

Una función es un conjunto de órdenes organizadas conjuntamente para ejecutar una acción determinada.

```
f.sum <- function(parameters){  
    out.sum <- sum(parameters)  
    return(out.sum)  
}
```

```
f.sum(2)
```

```
[1] 2
```

```
f.sum(c(2,3,4,5))
```

```
[1] 14
```

Funciones II

Parámetros de una función

Los parámetros de una función se sustituyen por valores, por lo que cambiarlos dentro de la función no los cambia fuera de la función.

```
a <- 2  
cat("a antes de la función:",a,"\n")
```

a antes de la función: 2

```
f.sum <- function(parameters){  
  a <- 3  
  cat("a dentro de la función:",a,"\n")  
  out.sum <- parameters + a  
  return(out.sum)  
}  
  
f.sum(2)
```

a dentro de la función: 3

[1] 5

```
cat("a después de la función:",a,"\n")
```

a después de la función: 2

Funciones III

Variables dentro de funciones I

Las variables definidas dentro de una función **no estarán disponibles** fuera de ella.

```
f.sum <- function(parameters){  
  b <- 3  
  cat("b inside the function:",b,"\n")  
  out.sum <- parameters + b  
  return(out.sum)  
}  
  
f.sum(2)
```

b inside the function: 3

[1] 5

```
cat("b after the function:",b,"\n")
```

Error in cat("b after the function:", b, "\n"): object 'b' not found

Funciones IV

Variables dentro de funciones II

Si utilizamos “«-”, las variables definidas dentro de la función **estarán disponibles** fuera de ella.

```
f.sum <- function(parameters){  
    b <- 3  
    cat("b dentro de la función:",b,"\n")  
    out.sum <- parameters + b  
    return(out.sum)  
}  
  
f.sum(2)
```

b dentro de la función: 3

[1] 5

```
cat("b después de la función:",b,"\n")
```

b después de la función: 3

Bucles I

Un bucle (loop) básico

```
for(i in 1:3) {  
  cat("i:", i, "\n") }  
}
```

```
i: 1  
i: 2  
i: 3
```

Bucle anidado

```
mat = matrix(nrow=4,  
             ncol=4)  
for(i in 1:4){  
  for(j in 1:4){  
    mat[i,j] = i+j  }  }  
mat
```

	[,1]	[,2]	[,3]	[,4]
[1,]	2	3	4	5
[2,]	3	4	5	6
[3,]	4	5	6	7
[4,]	5	6	7	8

Bucles II

Los bucles son lentos

```
A <- matrix(1:(200*200),nrow=200, ncol=200,byrow=TRUE)
B <- matrix(
          nrow=200, ncol=200
        )
A[1:2,1:7]
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]     1     2     3     4     5     6     7
[2,]    201    202    203    204    205    206    207
```

Un bucle para una multiplicación de matrices $B = A \%*\% A$ es lento

```
system.time(
  for (i in 1:200) {
    for (j in 1:200) {
      temp <- 0
      for (k in 1:200) {
        temp <- temp + A[i,k]*A[k,j]
      }
      B[i,j] <- temp
    }
  }
)
```

```
user  system elapsed
0.71   0.00   0.71
```

Bucles III

Intenta evitar bucles

```
A <- matrix(1:(200*200),nrow=200, ncol=200,byrow=TRUE)
B <- matrix(
                    nrow=200, ncol=200
)
A[1:2,1:7]
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]     1     2     3     4     5     6     7
[2,]    201    202    203    204    205    206    207
```

Y, en su lugar, utiliza funciones internas para una multiplicación de matrices $B = A \%*\% A$, que son más rápidas

```
system.time(
  B <- A \%*\% A
)
```

```
user  system elapsed
0      0          0
```

Evitando bucles

- Trabajando con matrices, podemos evitar bucles utilizando las funciones **apply(M,1,fun)** o **apply(M, 2,fun)**
- Este comando aplica la función específica *fun*
 - a las filas de M, si toma el valor 1
 - a las columnas de M, si toma el valor 2

```
mat.A <- matrix(rep(1:3,3),3,3,byrow=TRUE)  
mat.A
```

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	1	2	3
[3,]	1	2	3

Suma de filas utilizando **apply**

```
apply(mat.A, 1, sum)
```

```
[1] 6 6 6
```

Multiplicación de columnas utilizando **apply**

```
apply(mat.A, 2, prod)
```

```
[1] 1 8 27
```

¿Por qué R?
○○○○○○○

Conceptos básicos
○○○○

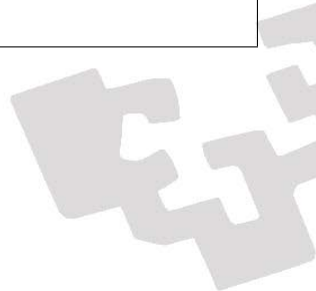
Vectores
○○○○○○○○○○

Matrices
○○○○○○○○○○

Funciones y bucles
○○○○○○○○○

Conjuntos de datos
○○○○

Conjuntos de datos



Data frames I

Los conjuntos de datos o *data frame* son estructuras bidimensionales de datos

```
employee.df <- data.frame("id"      = 101:104,  
                           "age"    = c(34,23,19,28),  
                           "Name"   = c("John",  
                                         "Sara",  
                                         "Rachel",  
                                         "Arnold"))  
  
employee.df
```

	id	age	Name
1	101	34	John
2	102	23	Sara
3	103	19	Rachel
4	104	28	Arnold

Data frames II

Los conjuntos de datos puede ser tratados como matrices

```
employee.df
```

```
   id age  Name
1 101  34  John
2 102  23  Sara
3 103  19 Rachel
4 104  28 Arnold
```

```
employee.df[,c(1,3)]
```

```
   id  Name
1 101  John
2 102  Sara
3 103 Rachel
4 104 Arnold
```

```
employee.df[employee.df$id > 102,]
```

```
   id age  Name
3 103  19 Rachel
4 104  28 Arnold
```

Leyendo un archivo de datos

- Comprueba tu directorio de trabajo: `getwd()`
- En caso necesario, cambia tu directorio de trabajo: `setwd("C:/...")`
- Para leer un archivo de datos

```
mydata <- read.csv("https://raw.githubusercontent.com/DabidH/datasets/master/mr
```

Fin de la sesión



David Hoyos

Universidad del País Vasco (UPV/EHU)

Facultad de Economía y Empresa

Departamento de Economía Aplicada III (Econometría y Estadística)

Tel.: +34 94 601 7019

Email: david.hoyos@ehu.eus

<http://www.ehu.eus/web/eephorad/david.hoyos>

<https://github.com/DabidH/Ekonometria>