1 Data

1.1 Datasets

A **univariate dataset** is a sequence of observations $Y_1, \ldots, Y_n$. These n observations can be organized into the **data vector** $\boldsymbol{Y}$, represented as $\boldsymbol{Y} = (Y_1, \ldots, Y_n)'$. For example, if you conduct a survey and ask five individuals about their hourly earnings, your data vector might look like

$$\boldsymbol{Y} = \begin{pmatrix} 18.22 \\ 23.85 \\ 10.00 \\ 6.39 \\ 7.42 \end{pmatrix}.$$

Typically we have data on more than one variable, such as years of education and the gender. Categorical variables are often encoded as **dummy variables**, which are binary variables. The female dummy variable is defined as 1 if the gender of the person is female and 0 otherwise.

person	wage	education	female
1	18.22	16	1
2	23.85	18	0
3	10.00	16	1
4	6.39	13	0
5	7.42	14	0

A k**-variate dataset** (or multivariate dataset) is a collection of n vectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ containing data on k variables. The i-th vector $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{ik})'$ contains the data on all k variables for individual i. Thus, X_{ij} represents the value for the j-th variable of individual i.

The full k-variate dataset is structured in the $n \times k$ **data matrix** $\boldsymbol{X}$:

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_1' \\ \vdots \\ \boldsymbol{X}_n' \end{pmatrix} = \begin{pmatrix} X_{11} & \ldots & X_{1k} \\ \vdots & \ddots & \vdots \\ X_{n1} & \ldots & X_{nk} \end{pmatrix}$$

The i-th row in $\boldsymbol{X}$ corresponds to the values from $\boldsymbol{X}_i$. Since $\boldsymbol{X}_i$ is a column vector, we use the transpose notation $\boldsymbol{X}_i'$, which is a row vector. The data matrix and vectors for our example are:

$$\boldsymbol{X} = \begin{pmatrix} 18.22 & 16 & 1 \\ 23.85 & 18 & 0 \\ 10.00 & 16 & 1 \\ 6.39 & 13 & 0 \\ 7.42 & 14 & 0 \end{pmatrix}, \quad \boldsymbol{X}_1 = \begin{pmatrix} 18.22 \\ 16 \\ 1 \end{pmatrix}, \boldsymbol{X}_2 = \begin{pmatrix} 23.85 \\ 18 \\ 0 \end{pmatrix}, \ldots$$

Vector and matrix algebra provide a compact mathematical representation of multivariate data and an efficient framework for analyzing and implementing statistical methods. We will use matrix algebra frequently throughout this course.

To refresh or enhance your knowledge of matrix algebra, please consult the following resources:

> **Crash Course on Matrix Algebra:**
>
> matrix.svenotto.com
> Section 19.1 of the Stock and Watson book also provides a brief overview of matrix algebra concepts.

1.2 R programming language

The best way to learn statistical methods is to program and apply them yourself. Throughout this course, we will use the `R` programming language for implementing empirical methods and analyzing real-world datasets.

If you are just starting with `R`, it is crucial to familiarize yourself with its basics. Here's an introductory tutorial, which contains a lot of valuable resources:

> **Getting Started with R:**
>
> rintro.svenotto.com

For those new to R, I also recommend the interactive R package SWIRL, which offers an excellent way to learn directly within the R environment. Additionally, a highly recommended online book to learn R programming is Hands-On Programming with R.

One of the best features of R is its extensive ecosystem of packages contributed by the statistical community. You find R packages for almost any statistical method out there and many statisticians provide R packages to accompany their research.

One of the most frequently used packages in applied econometrics is the `AER` package ("Applied Econometrics with R"), which provides a comprehensive collection of inferential methods for linear models. You can install the package with the command `install.packages("AER")` and you can load it with

```r
library(AER)
```

at the beginning of your code. We will explore several additional packages in the course of the lecture.

1.3 Datasets in R

R includes many built-in datasets and packages of datasets that can be loaded directly into your R environment. For illustration, we consider the `CASchools` dataset available in the `AER` package. This dataset is used in the Stock and Watson textbook in sections 4-8. It contains information on various characteristics of schools in California, such as test scores, teacher salaries, and student demographics.

To load this dataset into your R session, simply use:

```r
data(CASchools, package = "AER")
```

To get a description of the dataset, use the `?CASchools` command.

```r
class(CASchools)
```

```
[1] "data.frame"
```

The `CASchools` dataset is stored as a `data.frame`, R's most common data storage class for tabular data as in $\boldsymbol{X}$. It organizes data in the form of a table, with variables as columns and observations as rows.

To inspect the structure of your dataset, you can use `str()`:

```r
str(CASchools)
```

```
'data.frame':	420 obs. of  14 variables:
 $ district   : chr  "75119" "61499" "61549" "61457" ...
 $ school     : chr  "Sunol Glen Unified" "Manzanita Elementary" "Thermalito Union Elementary
 $ county     : Factor w/ 45 levels "Alameda","Butte",..: 1 2 2 2 2 6 29 11 6 25 ...
 $ grades     : Factor w/ 2 levels "KK-06","KK-08": 2 2 2 2 2 2 2 2 2 1 ...
 $ students   : num  195 240 1550 243 1335 ...
```

```
 $ teachers   : num  10.9 11.1 82.9 14 71.5 ...
 $ calworks   : num  0.51 15.42 55.03 36.48 33.11 ...
 $ lunch      : num  2.04 47.92 76.32 77.05 78.43 ...
 $ computer   : num  67 101 169 85 171 25 28 66 35 0 ...
 $ expenditure: num  6385 5099 5502 7102 5236 ...
 $ income     : num  22.69 9.82 8.98 8.98 9.08 ...
 $ english    : num  0 4.58 30 0 13.86 ...
 $ read       : num  692 660 636 652 642 ...
 $ math       : num  690 662 651 644 640 ...
```

The dataset contains variables of different types: `chr` for character/text data, `Factor` for categorical data, and `num` for numeric data. The `head()` function displays its first few rows:

```r
head(CASchools)
```

```
  district                          school  county grades students teachers
1    75119              Sunol Glen Unified Alameda  KK-08      195    10.90
2    61499            Manzanita Elementary   Butte  KK-08      240    11.15
3    61549     Thermalito Union Elementary   Butte  KK-08     1550    82.90
4    61457 Golden Feather Union Elementary   Butte  KK-08      243    14.00
5    61523        Palermo Union Elementary   Butte  KK-08     1335    71.50
6    62042         Burrel Union Elementary  Fresno  KK-08      137     6.40
  calworks   lunch computer expenditure    income   english  read  math
1   0.5102  2.0408       67    6384.911 22.690001  0.000000 691.6 690.0
2  15.4167 47.9167      101    5099.381  9.824000  4.583333 660.5 661.9
3  55.0323 76.3226      169    5501.955  8.978000 30.000002 636.3 650.9
4  36.4754 77.0492       85    7101.831  8.978000  0.000000 651.9 643.5
5  33.1086 78.4270      171    5235.988  9.080333 13.857677 641.8 639.9
6  12.3188 86.9565       25    5580.147 10.415000 12.408759 605.7 605.4
```

The pipe operator `|>` efficiently chains commands. It passes the output of one function as the input to another. For example:

```r
CASchools[,c("school", "county", "income")] |> summary()
```

```
    school                     county        income      
 Length:420         Sonoma     : 29   Min.   : 5.335  
 Class :character   Kern       : 27   1st Qu.:10.639  
 Mode  :character   Los Angeles: 27   Median :13.728  
                    Tulare     : 24   Mean   :15.317  
                    San Diego  : 21   3rd Qu.:17.629  
```

```
                 Santa Clara: 20   Max.   :55.328
                 (Other)    :272
```

The `summary()` function presents a concise overview, showing absolute frequencies for categorical variables and descriptive statistics for numerical variables.

The variable `students` contains the total number of students enrolled in a school. It is the fifth variable in the data set. To access the variable as a vector, you can type `CASchools[,5]` (the fifth column in your data matrix), or `CASchools[,"students"]`, or simply `CASchool$students`.

We can easily add new variables to a dataframe, for instance, the student-teacher ratio (the total number of students per teacher) and the average test score (average of the math and reading scores):

```
# compute student-teacher ratio and append it to CASchools
CASchools$STR = CASchools$students/CASchools$teachers
# compute test score and append it to CASchools
CASchools$score = (CASchools$read+CASchools$math)/2
```

The variable `english` indicates the proportion of students whose first language is not English and who may need additional support. We might be interested in the dummy variable `HiEL`, which indicates whether the proportion of English learners is above 10 percent or not:

```
# append HiEL to CASchools
CASchools$HiEL = (CASchools$english >= 10) |> as.numeric()
```

Note that `CASchools$english >= 10` is a logical expression with either `TRUE` or `FALSE` values. The command `as.numeric()` creates a dummy variable by translating `TRUE` to `1` and `FALSE` to `0`.

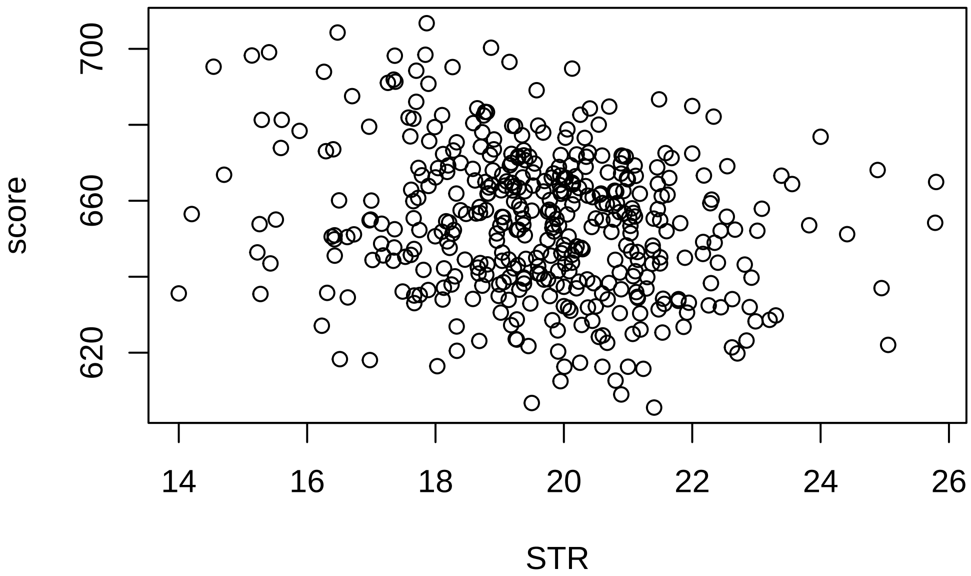
The first few values of some selected variables look like this:

```
CASchools[,c("STR", "score", "english", "HiEL", "income")] |> head()
```

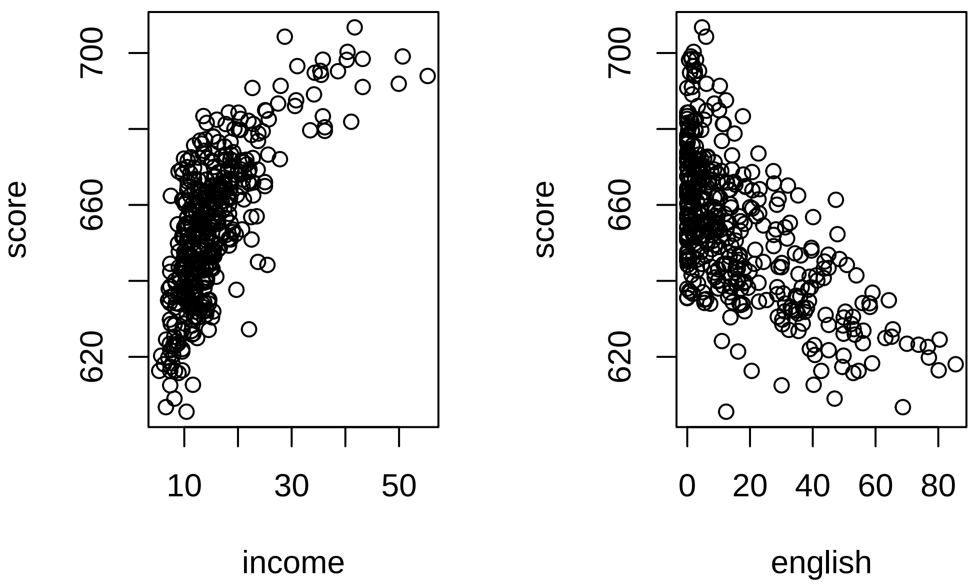
```
       STR  score   english HiEL    income
1 17.88991 690.80  0.000000    0 22.690001
2 21.52466 661.20  4.583333    0  9.824000
3 18.69723 643.60 30.000002    1  8.978000
4 17.35714 647.70  0.000000    0  8.978000
5 18.67133 640.85 13.857677    1  9.080333
6 21.40625 605.55 12.408759    1 10.415000
```

Scatterplots provide further insights:

```
plot(score~STR, data = CASchools)
```




```
par(mfrow = c(1,2))
plot(score~income, data = CASchools)
plot(score~english, data = CASchools)
```



The option `par(mfrow = c(1,2))` allows to display multiple plots side by side. Try what happens if you replace `c(1,2)` with `c(2,1)`.

1.4 Importing data

The internet serves as a vast repository for data in various formats, with `csv` (comma-separated values), `xlsx` (Microsoft Excel spreadsheets), and `txt` (text files) being the most commonly used.

R supports various functions for different data formats:

- `read.csv()` for reading comma-separated values
- `read.csv2()` for semicolon-separated values (adopting the German data convention of using the comma as the decimal mark)
- `read.table()` for whitespace-separated files
- `read_excel()` for Microsoft Excel files (requires the `readxl` package)
- `read_stata()` for STATA files (requires the `haven` package)

Let's import the CPS dataset from Bruce Hansen's textbook. The Current Population Survey (CPS) is a monthly survey conducted by the U.S. Census Bureau for the Bureau of Labor Statistics, primarily used to measure the labor force status of the U.S. population.

- Dataset: cps09mar.txt
- Description: cps09mar_description.pdf

```
url = "https://users.ssc.wisc.edu/~bhansen/econometrics/cps09mar.txt"
varnames = c("age","female","hisp","education","earnings","hours",
             "week","union","uncov","region","race","marital")
cps = read.table(url, col.names=varnames)
```

Let's create further variables:

```
# wage per hour
cps$wage = cps$earnings/(cps$week*cps$hours)
# years since graduation
cps$experience = (cps$age - cps$education - 6)
# married dummy
cps$married = cps$marital %in% c(1,2) |> as.numeric()
# Black dummy
cps$Black = (cps$race %in% c(2,6,10,11,12,15,16,19)) |> as.numeric()
# Asian dummy
cps$Asian = (cps$race %in% c(4,8,11,13,14,16,17,18,19)) |> as.numeric()
```

We will be using the cps data in the next sections, so it is a good idea to save the dataset to your computer:

```r
write.csv(cps, "cps.csv", row.names = FALSE)
```

To read the data back into R later, just type `cps = read.csv("cps.csv")`.

1.5 R-codes

statistics-sec01.R