Data Analytics for Social Science
Linear regression

Johan A. Elkink

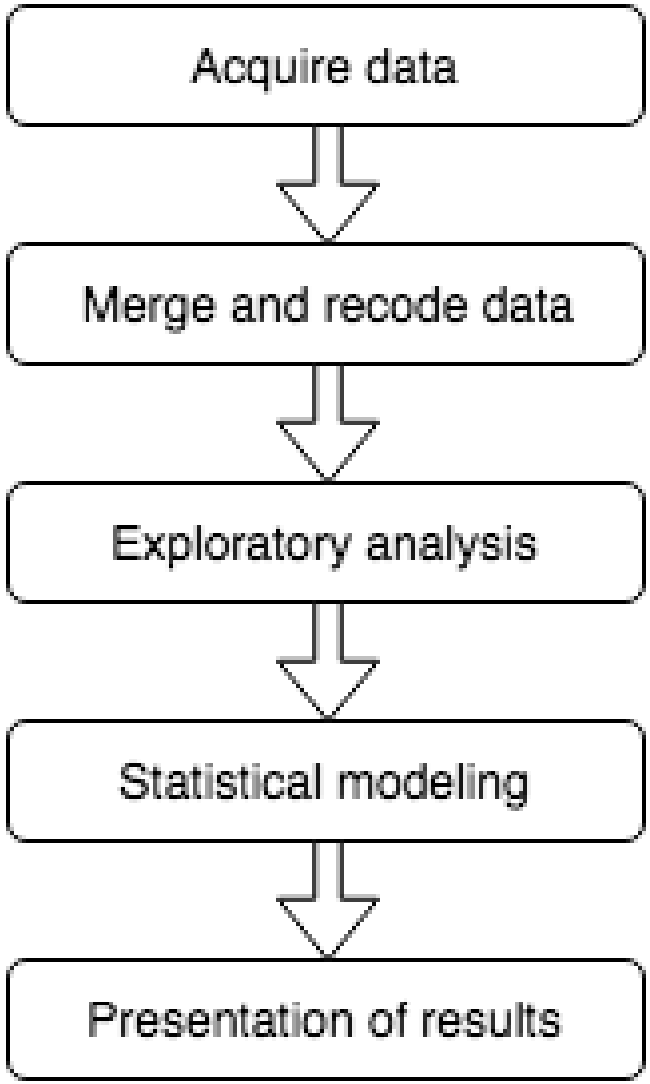
School of Politics & International Relations
University College Dublin

13 February 2020

1. Computing, recoding, and merging
2. Linear model and Ordinary Least Squares
3. Multiple regression
4. Regression in R
5. Model selection
6. More on computing, recoding, and merging
7. Some more details on OLS

Outline

Recoding



Basic computations

R can be used as a basic calculator:

```
5 * 3 / (3 + 1)
```

Commonly variables are created by some basic computations:

```
data <- data %>% mutate(
  age = 2017 - birthyear,
  age2 = age ^ 2
)
```

Transformations

Similar computations can also involve other functions, such as a logarithmic transformation:

```
data <- data %>% mutate(
  logGDP = log(GDP, base = 10)
)
```

Comparison operators

We often need to select cases, or recode variables, based on ranges of data. We can use comparison operators for this.

==	Equal
!=	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal

```
data <- data %>% mutate(
  young = factor(ifelse(age < 41,
                        "Young",
                        "Old"))
)
```

Logical operators

In addition to comparison operators, we can use logical operators.

or	`TRUE | TRUE = TRUE`
	`TRUE | FALSE = TRUE`
	`FALSE | TRUE = TRUE`
	`FALSE | FALSE = FALSE`

and	`TRUE & TRUE = TRUE`
	`TRUE & FALSE = FALSE`
	`FALSE & TRUE = FALSE`
	`FALSE & FALSE = FALSE`

```
middleAged <- factor(ifelse(age >= 41 & age < 55,
    "Middle-aged", "Not middle-aged"))
```

Recoding variables

We often encounter situations where we want to re-order, re-group, or re-label categories of a variable.

```
data <- data %>% mutate(
  abolishSeanad = recode(q41, `1` = 1, `2` = 1,
    `3` = 2, `4` = 3, `5` = 3, .default = NA)
)
```

old categories (q41)	new categories (abolishSeanad)
1 = Strongly agree	1 = Agree
2 = Agree	
3 = Neutral	2 = Neutral
4 = Disagree	3 = Disagree
5 = Strong disagree	

Recoding variables

We often encounter situations where we want to re-order, re-group, or re-label categories of a variable.

```
data <- data %>% mutate(
  abolishSeanad <- recode(q41,
  `1` = "Agree", `2` = "Agree", `3` = "Neutral",
  `4` = "Disagree", `5` = "Disagree", .default = NA)
)
```

old categories (q41)	new categories (abolishSeanad)
1 = Strongly agree	Agree
2 = Agree	
3 = Neutral	Neutral
4 = Disagree	Disagree
5 = Strong disagree	

Check recoding

When you recode, *always* check how it worked out.

Usually by producing a cross-table, and maybe a plot of the new variable, to see if it makes sense.

```
pander(table(data$abolishSeanad, data$q41,
             exclude = NULL))
```

Note that `exclude = NULL` means that missing values are included, which is often crucial to check the recoding. Normally you do not want this in cross-tables.

Merging data

We often have data from different sources, that we want to put together in one data set for further analysis.

This requires that observations have the same identifier to merge on. For example, we might have data on candidates in electoral districts, and other data on the electoral districts themselves.

party	name	district	votes
FF	O'Brien	Longford S	10,432
FG	Fitzgerald	Longford S	5,429
FF	MacGuinness	Dublin NC	15,436
Labour	Shenigan	Dublin NC	2,013

district	seats	total
Dublin SC	5	100,410
Dublin NC	4	98,991
Longford S	3	70,001
Wexford	4	81,013

party	name	district	votes
FF	O'Brien	Longford S	10,432
FG	Fitzgerald	Longford S	5,429
FF	MacGuinness	Dublin NC	15,436
Labour	Shenigan	Dublin NC	2,013

district	seats	total
Dublin SC	5	100,410
Dublin NC	4	98,991
Longford S	3	70,001
Wexford	4	81,013

party	name	district	votes	seats	total
FF	O'Brien	Longford S	10,432	3	70,001
FG	Fitzgerald	Longford S	5,429	3	70,001
FF	MacGuinness	Dublin NC	15,436	4	98,991
Labour	Shenigan	Dublin NC	2,013	4	98,991

Merging data

We often have data from different sources, that we want to put together in one data set for further analysis.

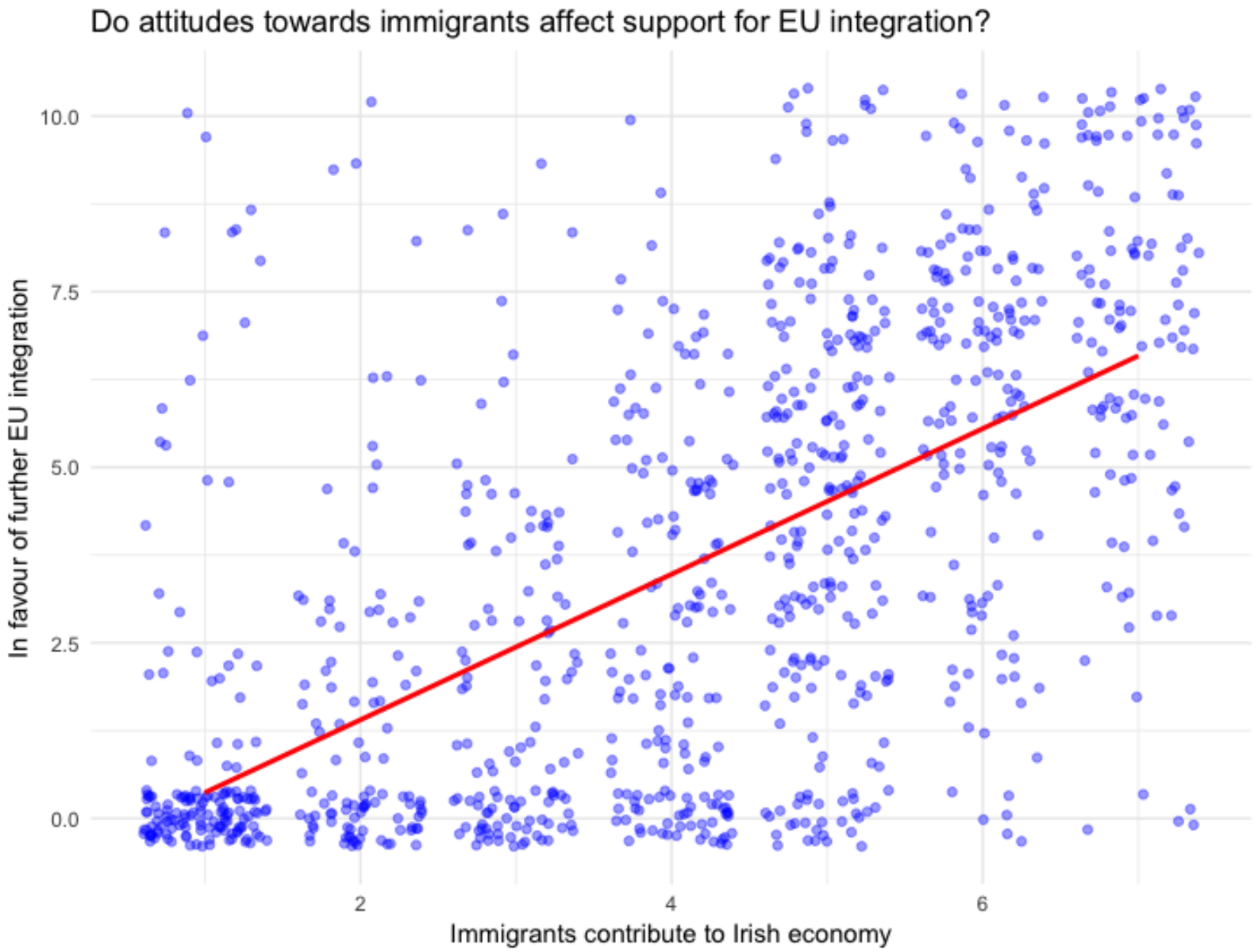
This requires that observations have the same identifier to merge on. For example, we might have data on candidates in electoral districts, and other data on the electoral districts themselves.

```
mergedData <- merge(candidates, districts,
  by = "districtID", all.x = TRUE)
```

This merges two data files, "candidates" and "districts", by a variable that occurs in each called "districtID", and we ensure that all candidates stay in the data set, even if there is no data on that particular district.

Outline

Linear model



Linear model

The regression equation here is

$$y_i = \beta_1 + \beta_2 x_i + \varepsilon_i,$$

whereby **y** is the dependent variable, **x** the independent variable, i an indicator of the case (e.g. country), β_1 and β_2 the model parameters, and ε the error term.

The model thus uses a linear combination of the independent variables to predict or explain the dependent variable, whereby both are assumed to be interval or ratio variables.

The high level of **parsimony** of the model reduces the risk of **overfitting** and thus helps us to generalize (cf. the lines we saw with `geom_smooth()`).

Residuals

$$y_i = \beta_1 + \beta_2 x_i + \varepsilon_i$$

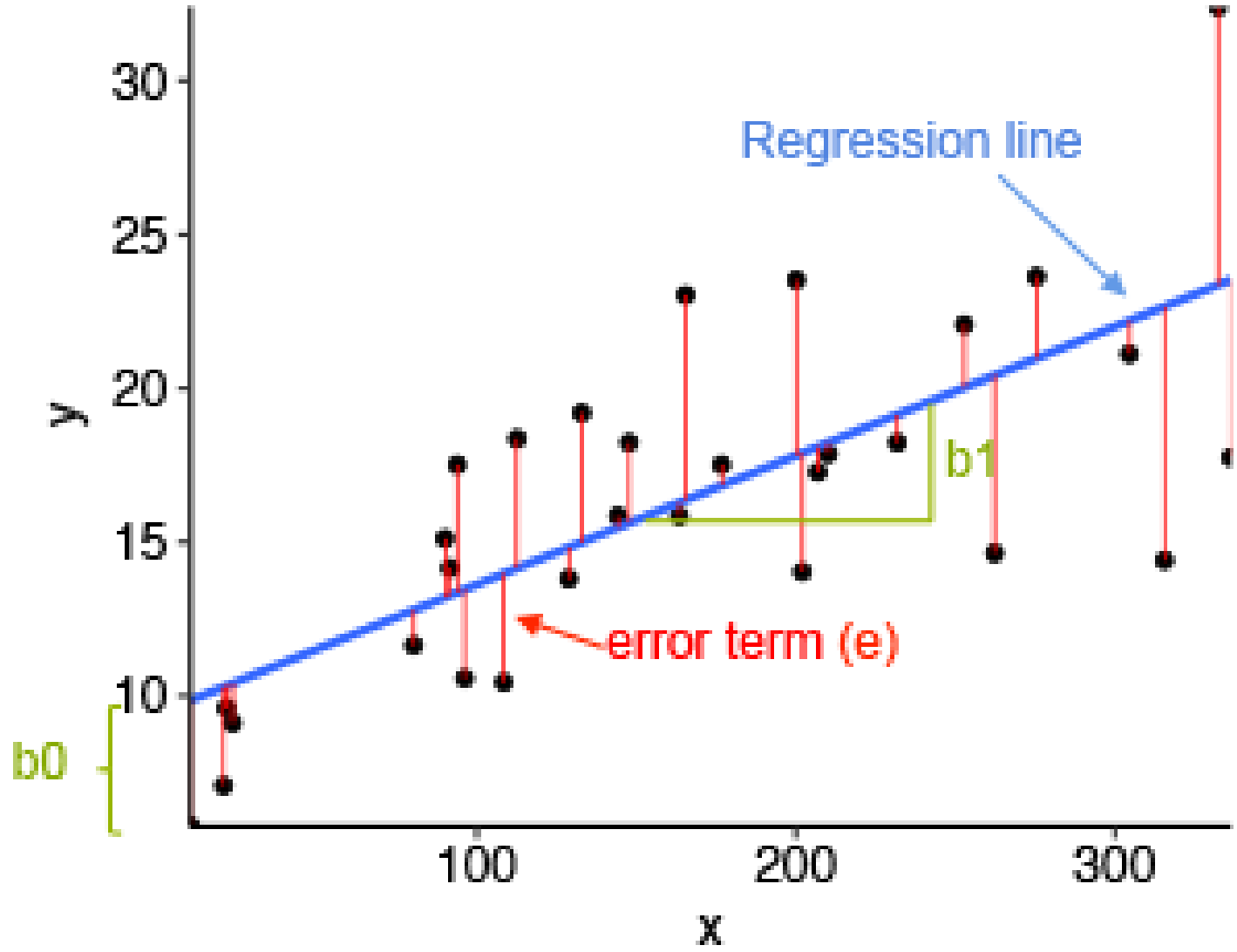
The linear prediction given the parameters would be $\hat{y}_i = \hat{\beta}_1 + \hat{\beta}_2 x_i$.

The extend to which the real value differs from the predicted value is:

$$y_i - \hat{y}_i = y_i - \hat{\beta}_1 - \hat{\beta}_2 x_i = e_i.$$

By this formulation, the **residuals** ($\boldsymbol{e}$) are the vertical distances between a point and the regression line (i.e. not the shortest distance between the point and the line).

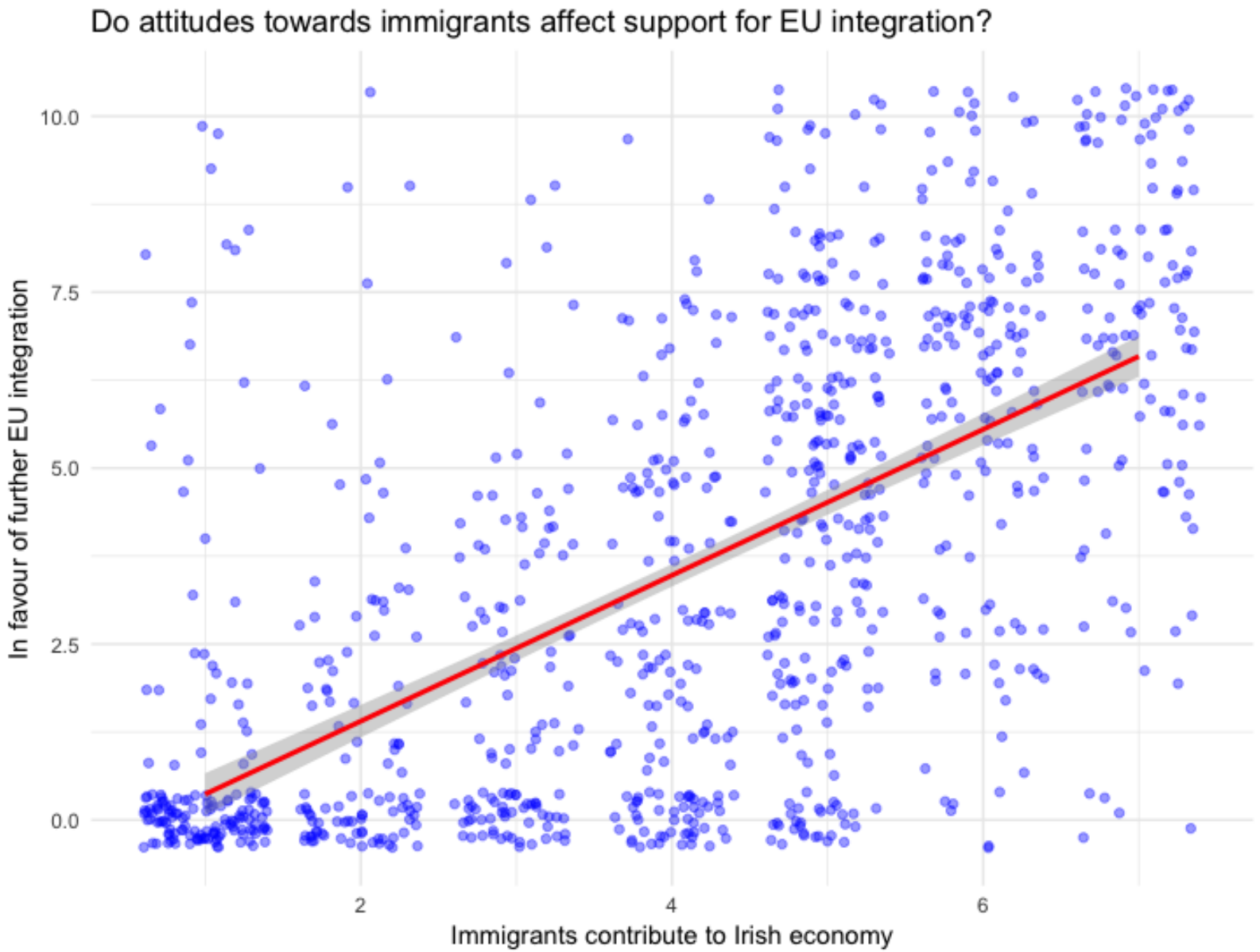
Linear model



Terminology

- **y** is the **dependent** variable
  - also known as **regressand**
- **X** are the **independent** variables
  - also known as **explanatory** variables
  - also known as **regressors** or **predictors** (or **factors**, **carriers**)
  - **X** is sometimes called the **design matrix** or **factor space**
- **y** is **regressed on X**
- β is the population **parameter**, $\hat{\beta}$ the estimated **coefficient**.
- The **error** term or **disturbance** $\varepsilon = \mathbf{y} - \mathbf{X}\beta$.
- The difference between the observed and predicted dependent variable is the **residual** $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}} = \mathbf{y} - \mathbf{X}\hat{\beta}$.

Linear model



Ordinary Least Squares

To estimate the regression line, we need to estimate the parameters β_1 and β_2.

For the **linear** model, the most popular method of estimation is **ordinary least squares** (OLS).

With OLS, we estimate the parameters such that the **sum of squared residuals** are minimized. This is the same as minimizing the variance of the residuals.

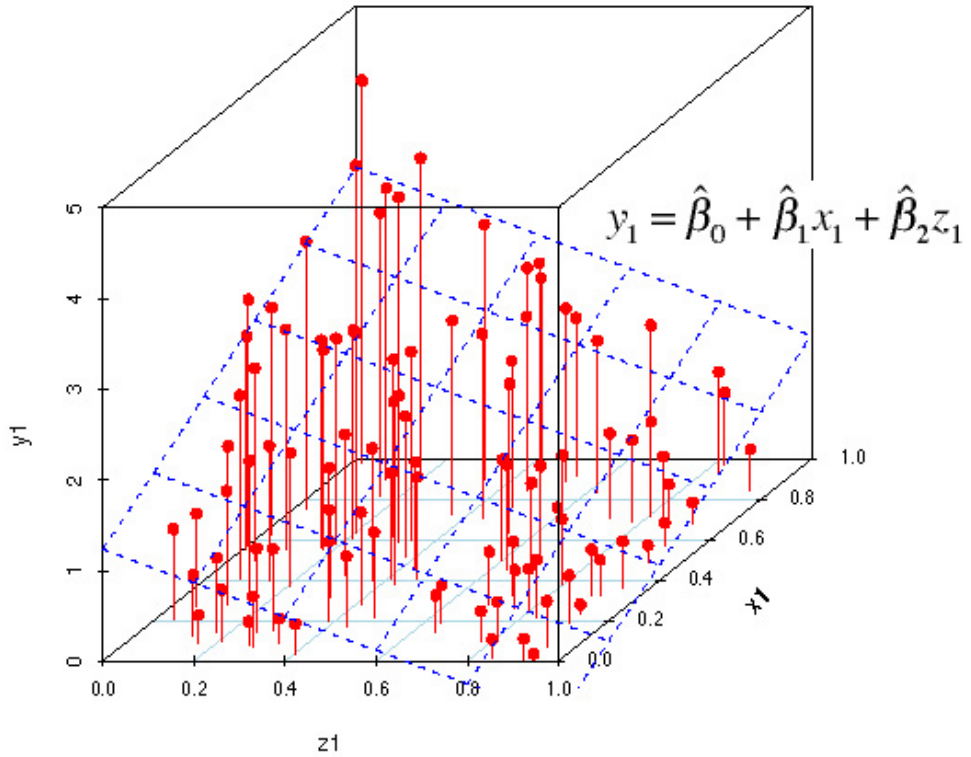
OLS is the **best linear unbiased estimator** (BLUE).

Outline

Multiple regression

For causal inference, we can add **control variables** to capture confounding factors, such as here a dummy variable where the country is a democracy, the log of GDP, and the log of population size.

For predictive inference, we can add variables to get a more accurate prediction.



Multiple regression

	proIntegration
immigrationEcon	1.006***
	(0.043)
age	−0.020***
	(0.006)
genderMale	−0.229
	(0.164)
income	−0.006
	(0.024)
nationalRegionNorthern Ireland	0.626***
	(0.216)
nationalRegionScotland	0.476
	(0.298)
nationalRegionWales	0.425
	(0.301)
Constant	0.486
	(0.401)
Observations	1,000
R^2	0.395

Dummy variables

Linear regression is only suitable for continuous variables.

However, when a categorical variable has only two categories, one coded as "1" and the other as "0", it is reasonable to insert into a regression as explanatory variable, take x_i a continuous and d_i a **dummy** variable:

$$y_i = \beta_1 + \beta_2 x_i + \beta_3 d_i$$

We can distinguish two scenarios:

$$\begin{array}{l|l} d_i{=}0 & y_i = \beta_1 + \beta_2 x_i + \beta_3 d_i = \beta_1 + \beta_2 x_i \\ d_i{=}1 & y_i = \beta_1 + \beta_2 x_i + \beta_3 d_i = (\beta_1 + \beta_3) + \beta_2 x_i \end{array}$$

So the coefficient β_3 is the *difference in intercept* for those where $d_i = 0$ and those where $d_i = 1$.

Nominal variables

Dealing with a nominal variable with multiple categories as explanatory variable is then straightforward: create multiple dummy variables, one for each category.

party	ff	fg	lab	other
ff	1	0	0	0
ff	1	0	0	0
fg	0	1	0	0
labour	0	0	1	0
fg	0	1	0	0
other	0	0	0	1

Remember to always leave one of the dummy variables out of the regression, which then becomes the **reference category**.

$$leftRight_i = \beta_1 + \beta_2 ff_i + \beta_3 lab_i + \beta_4 other_i$$

This means that β_2 represents the difference in intercept between FF and FG; β_3 between Labour and FG; etc.

Outline

Plotting the regression line

```
ggplot(brexit, aes(x = immigrationEcon,
                   y = proIntegration)) +
  geom_jitter(col = "blue", alpha = .4) +
  theme_minimal() +
  labs(x = "Immigrants contribute to Irish economy",
       y = "In favour of further EU integration",
       title = "Do attitudes towards ...") +
  geom_smooth(method = "lm", se = TRUE, col = "red")
```

Regression lines are added with the `geom_smooth` command where you need to add that `method` is equal to "lm", i.e., the linear model, instead of the standard smoothened curve that is plotted.

Getting the regression coefficients

```
library(stargazer)

stargazer(lm(proIntegration ~ immigrationEcon,
  brexit), type = "html")
```

If you combine this with the

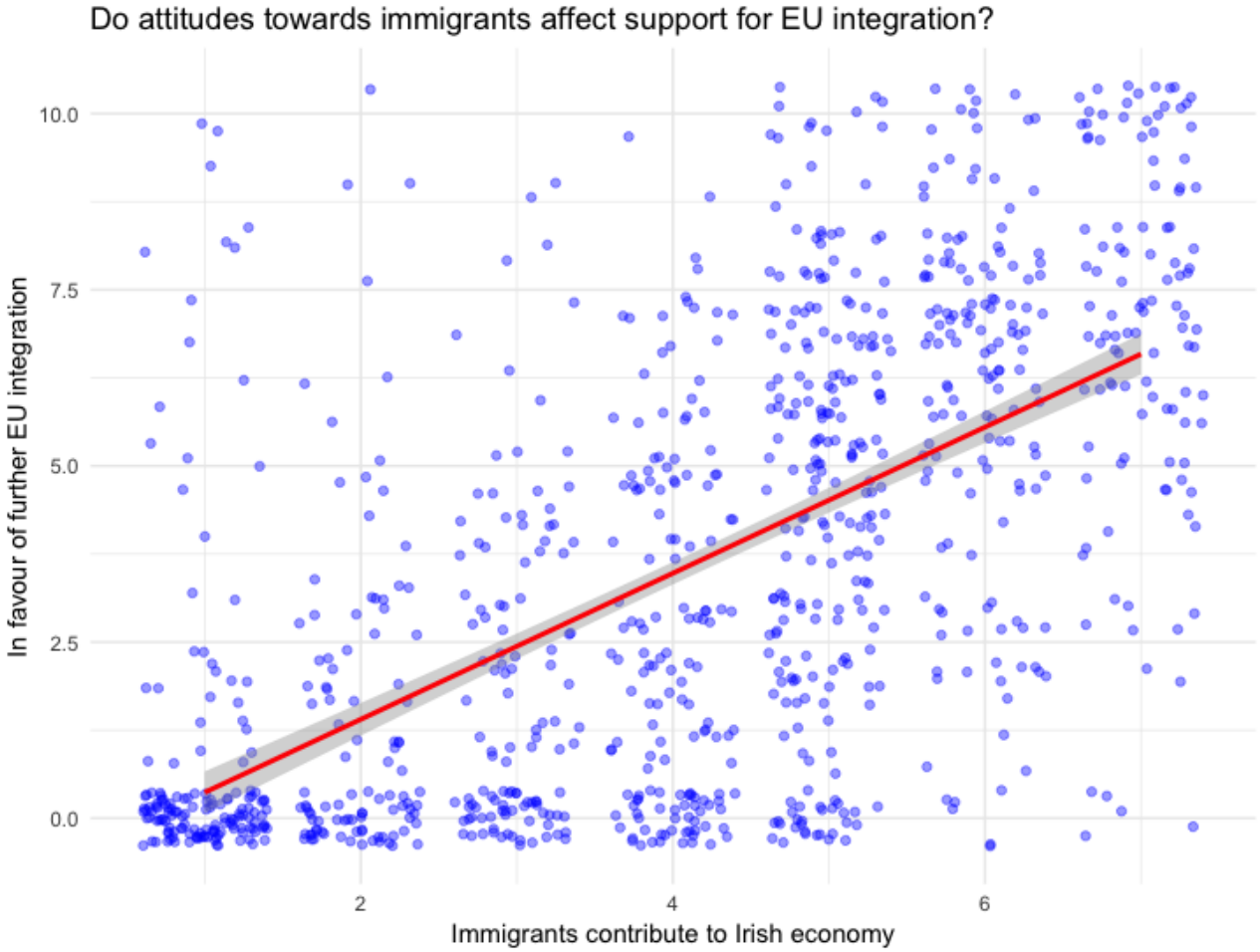
```
```{r results = "asis"}
```

option in the RMarkdown chunk, you get a nice looking regression table in your output file.
```

	Dependent variable:
	proIntegration
immigrationEcon	1.037^{***}
	(0.042)
Constant	-0.670^{***}
	(0.188)
Observations	1,000
R^2	0.380
Adjusted R^2	0.379
Residual Std. Error	2.559 (df = 998)
F Statistic	611.868^{***} (df = 1; 998)
Note:	*p<0.1; **p<0.05; ***p<0.01

$$proIntegration_i = \hat{\beta}_1 + \hat{\beta}_2 immigrationEcon_i$$

Linear model



$$proIntegration_i = -0.67 + 1.04 \cdot immigrationEcon_i$$

Multiple regression

Adding additional variables in the regression command is straightforward:

```
stargazer(m <- lm(proIntegration ~ immigrationEcon +
  gender + income + nationalRegion, brexit))
```

Here we estimate the impact of positive attitudes towards immigrants on pro-integration attitudes, controlling for age, gender, income, and national region.

Outline

R^2

Once we have estimated a line, we might ask how well this line summarizes the relationship between those two variables.

A common measure is R^2:

$$R^2 = 1 - \frac{\text{residual sum of squares}}{\text{total sum of squares}} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}.$$

This can be interpreted as the proportion of the variation in **y** explained by this model.

R^2 rises with the addition of more explanatory variables. For this reason we often report the **adjusted** R^2:

$$1 - (1 - R^2)\frac{n-1}{n-k}.$$

Akaike Information Criterion (AIC)

Another approach than the adjusted R^2 to making a similar balance between parsimony and explained variance is **Akaike Information Criterion**:

$$AIC = \log\left(\frac{\mathbf{e}'\mathbf{e}}{n}\right) + \frac{2k}{n}$$

Thus the smaller AIC, the better.

$$BIC = \log\left(\frac{\mathbf{e}'\mathbf{e}}{n}\right) + \frac{(\log n)k}{n}$$

$$AICc = AIC + \frac{2k(k+1)}{n-k-1}$$

and there are many more similar variations.

Stepwise regression

Stepwise regression is an algorithm where the computer adds or removes variables and evaluates the improvements in fit (e.g. AIC).

Step-Forward: adding variables until candidates add too little in terms of prediction.

Step-Backward: removing variables until candidates remove too much in terms of prediction.

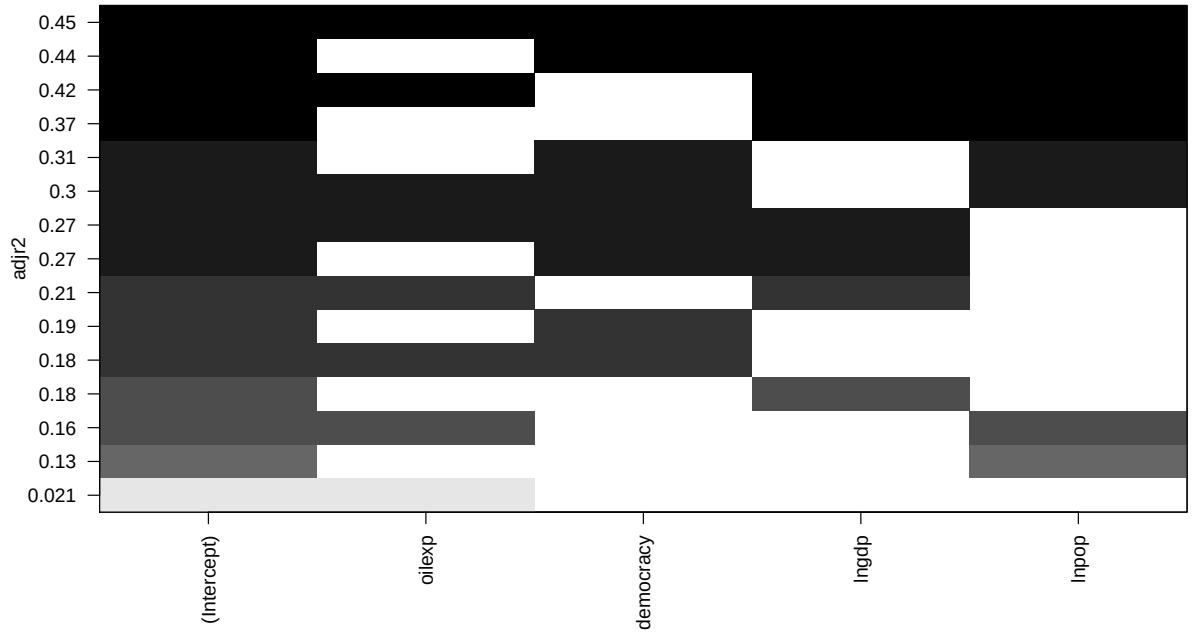
Mixed: allow both direction to find the best fitting model.

Note that this is only relevant for prediction, not causal inference! For the latter, we select variables on the basis of potential confounders.

All subsets regression

If the model is relatively quick to estimate—not too many variables and not a very large data set—then we can also just estimate models for all reasonable subsets, and select based on some fit criterion (R^2, AIC, etc.).

The *leaps* package in R provides a useful plot to select a model:



Ridge regression

Instead of adding or dropping variables, we can also simply put less weight on those variables that contribute less. In **ridge regression** we shrink β-estimates for variables which contribute less.

Ridge regression is similar to regular linear regression, but instead of only minimizing the sum of squared residuals, it also adds a penalty for high values of β.

$$\hat{\beta}^{OLS} = \arg\min_{\hat{\beta}} (\mathbf{y} - \mathbf{X}\hat{\beta})'(\mathbf{y} - \mathbf{X}\hat{\beta}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

$$\hat{\beta}^{ridge} = \arg\min_{\hat{\beta}} \left[(\mathbf{y} - \mathbf{X}\hat{\beta})'(\mathbf{y} - \mathbf{X}\hat{\beta}) + \lambda\beta'\beta\right]$$

$$= \left(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}'\mathbf{y}$$

The penalty is set by λ. If $\lambda = 0$, $\hat{\beta}^{OLS} = \hat{\beta}^{ridge}$.

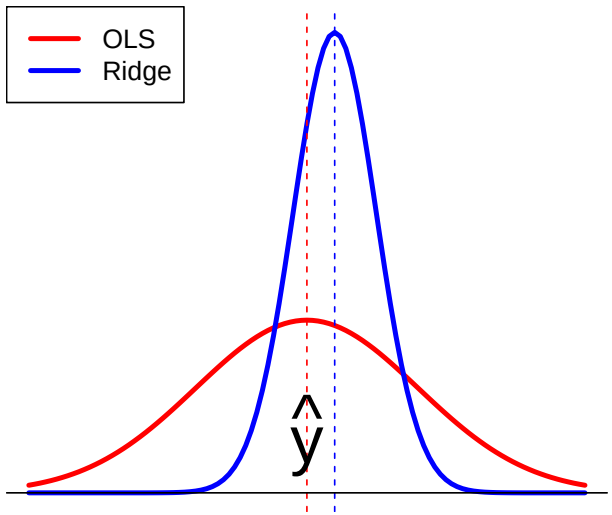
Ridge regression: benefits

Ridge regression is particularly useful with **high multicollinearity**, as highly multicollinear variables will be downweighted.

$\hat{\beta}^{ridge}$ a biased estimate of β, but typically has a lower prediction variance for $\hat{\mathbf{y}}$, such that the **mean squared error** might in fact be smaller.

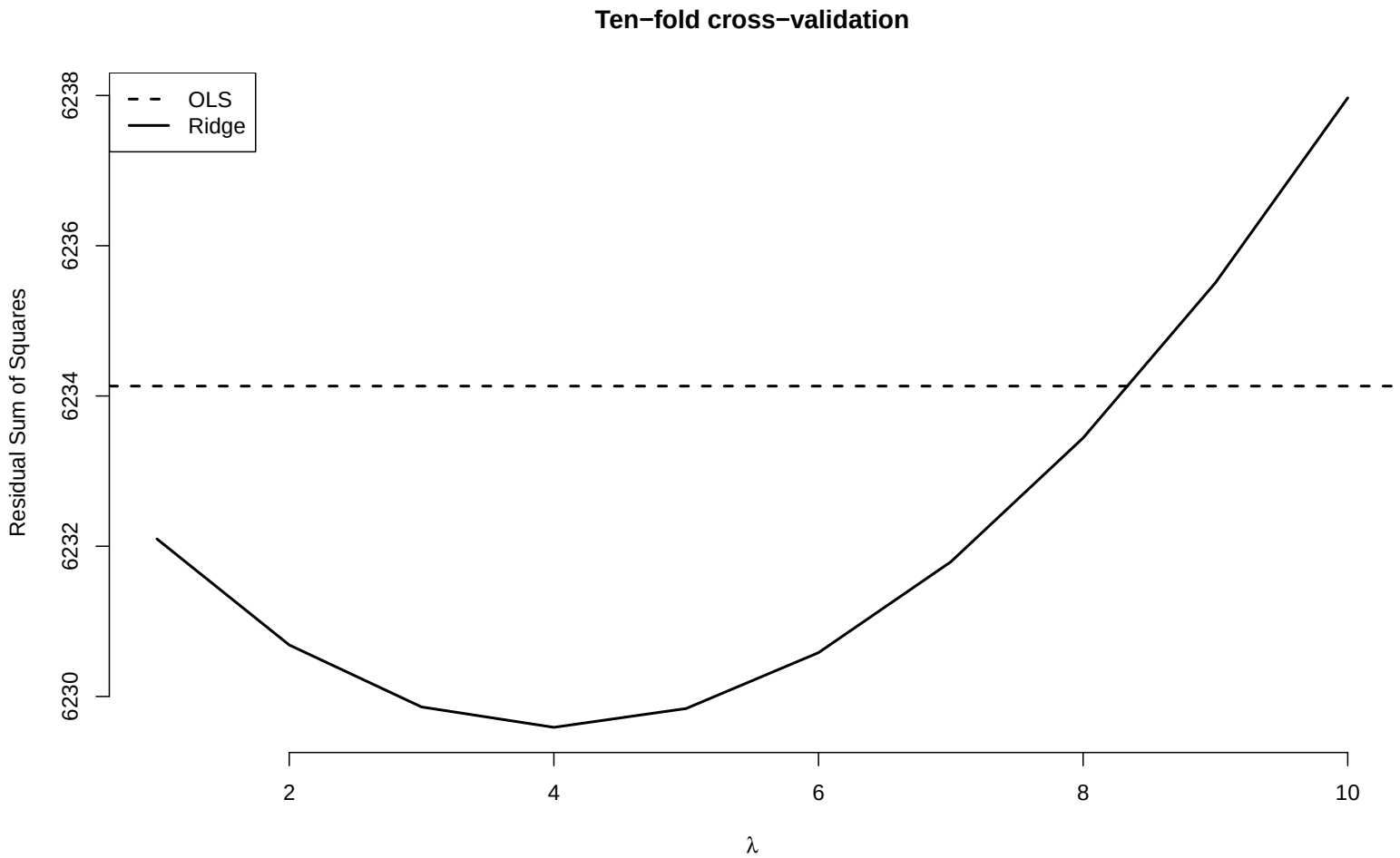
Ridge regression also protects against **overfitting**.

Other variations, such as the **lasso**, are also common in machine learning.



(Hastie, Tibshirani and Friedman, 2009, 61–68)

Ridge regression: selecting λ



Outline

Transformations

We can also create our own functions:

```
stdize <- function(x) {
  (x - mean(x, na.rm = TRUE)) /
    sd(x, na.rm = TRUE)
}

data <- data %>% mutate(
  zGDP = stdize(GDP),
  zAge = stdize(age)
)
```

(A **standardized variable** is one where, for each observation, we subtract the mean and divide by the standard deviation, so that the resulting variable has a mean of zero and a standard deviation (and variance) of one. This is also called the z-**score** of the value.)

Missing data

In many surveys, "don't know" or other missing data have been attributed a numerical value.

```
ess <- ess %>% mutate(
  proIntegration = 10 - recode(EUIntegrationSelfW8,
    `9999` = NA),
  party = recode(generalElectionVoteW9,
    `0` = "Would not vote",
    `1` = "Conservative",
    `2` = "Labour",
    `3` = "Liberal Democrat",
    `6` = "UKIP",
    `9999` = NA,
    .default = "Other")
)
```

Reshaping data

Often data that we download will have the same variable across different columns, for example for different years.

In most analyses this is difficult to work with and we need the same variable to be in one column. This requires reshaping the data.

country	pop1991	pop1992	pop1993
Brazil	150337	152680	154964
Bolivia	6733	6897	7065
Paraguay	4345	4470	4592
Chile	13319	13544	13771

country	pop1991	pop1992	pop1993
Brazil	150337	152680	154964
Bolivia	6733	6897	7065
Paraguay	4345	4470	4592
Chile	13319	13544	13771

country	year	pop
Brazil	1991	150337
Brazil	1992	152680
Brazil	1993	154964
Bolivia	1991	6733
Bolivia	1992	6897
Bolivia	1993	7065
Paraguay	1991	4345
Paraguay	1992	4470
Paraguay	1993	4592
Chile	1991	13319
Chile	1992	13544
Chile	1993	13771

Reshaping data

Often data that we download will have the same variable across different columns, for example for different years.

In most analyses this is difficult to work with and we need the same variable to be in one column. This requires reshaping the data.

In data science (another Wickham package!) we use the terms **melt** and **cast** for the two reshape directions.

```
library(reshape)
stackedData <- melt(downloadData, id = "country")
colnames(stackedData) <- c("country", "year",
  "population")
yearMeans <- cast(stackedData, ~ year, mean,
  value = "population")
```

Outline

Sums of squares

SST Total sum of squares $\sum(y_i - \bar{y})^2$

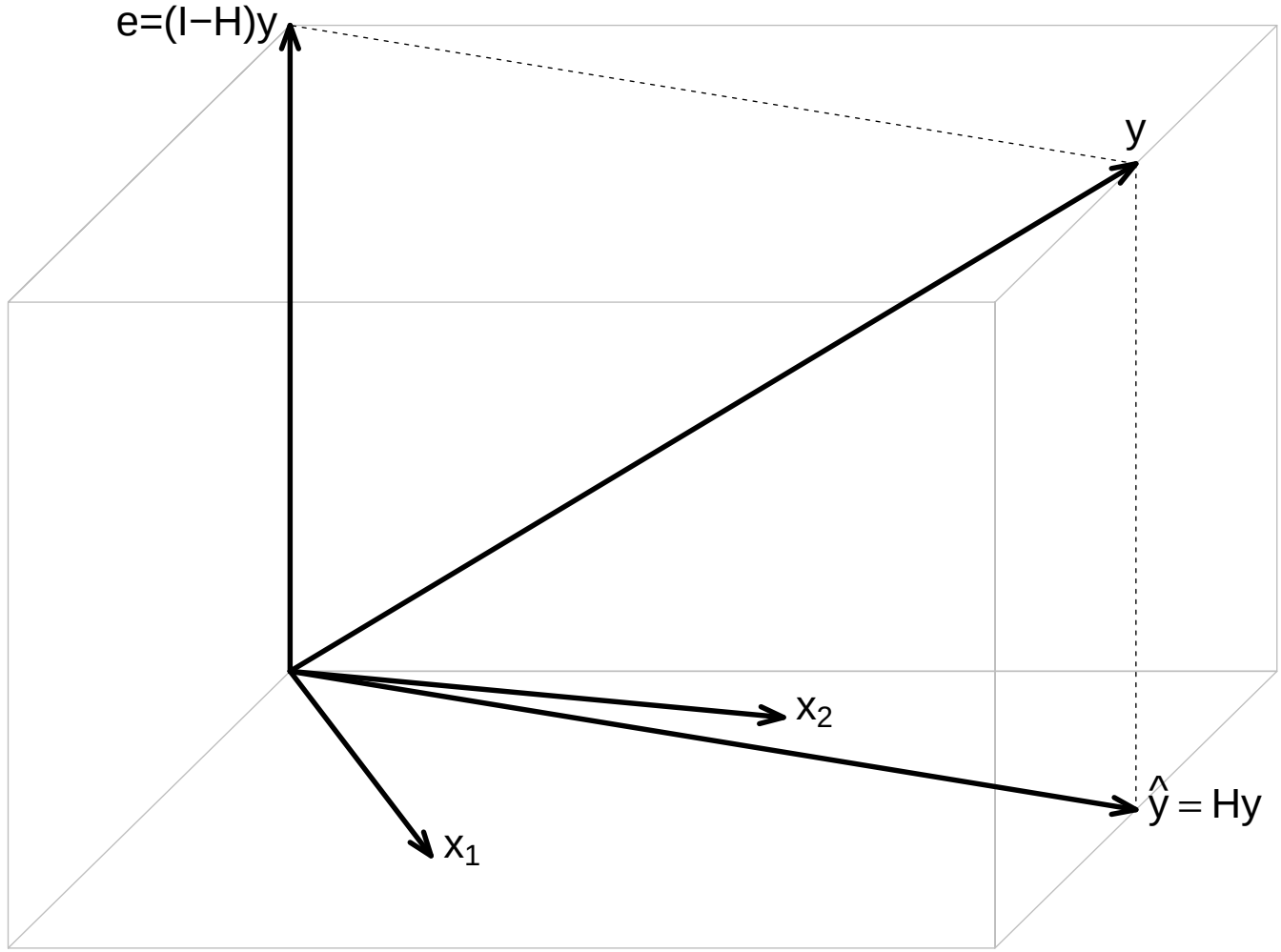
SSE Explained sum of squares $\sum(\hat{y}_i - \bar{y})^2$

SSR Residual sum of squares
$\sum e_i^2 = \sum(\hat{y}_i - y_i)^2 = \mathbf{e}'\mathbf{e}$

The key to remember is that **SST = SSE + SSR**

Sometimes instead of "explained" and "residual", "regression" and "error" are used, respectively, so that the abbreviations are swapped (!).

OLS as projection



$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} = \mathbf{H}\mathbf{y}$$

OLS assumptions

specification

Linear in parameters (i.e. $f(\mathbf{X}\beta) = \mathbf{X}\beta$ and $E(\mathbf{y}) = \mathbf{X}\beta$)
No extraneous variables in $\mathbf{X}$
No omitted independent variables
Parameters to be estimated are constant
Number of parameters is less than the number of cases, $k < n$

disturbances

Errors have an expected value of zero, $E(\varepsilon|\mathbf{X}) = 0$
Errors are normally distributed, $\varepsilon \sim N(0, \sigma^2)$
Errors have a constant variance, $var(\varepsilon|\mathbf{X}) = \sigma^2 < \infty$
Errors are not autocorrelated, $cov(\varepsilon_i, \varepsilon_j|\mathbf{X}) = 0 \quad \forall \quad i \neq j$
Errors and $\mathbf{X}$ are uncorrelated, $cov(\mathbf{X}, \varepsilon) = 0$

regressors

$\mathbf{X}$ varies and is of full column rank (note: requires $k < n$)
No measurement error in $\mathbf{X}$
No endogenous variables in $\mathbf{X}$

Components

Two components of the model:

$$\begin{array}{l|l} \mathbf{y} \sim N(\boldsymbol{\mu}, \sigma^2) & \text{Stochastic} \\ \boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta} & \text{Systematic} \end{array}$$

Generalised version (not necessarily linear):

$$\begin{array}{l|l} \mathbf{y} \sim f(\boldsymbol{\mu}, \boldsymbol{\alpha}) & \text{Stochastic} \\ \boldsymbol{\mu} = g(\mathbf{X}, \boldsymbol{\beta}) & \text{Systematic} \end{array}$$

Two types of uncertainty:

Estimation uncertainty: lack of knowledge about $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$; can be reduced by increasing n.

Fundamental uncertainty: represented by stochastic component and exists independent of researcher.

(King, 1998, 8)

t- and *F*-tests

The two most important tests in regression are, for each coefficient, the ***t*-test**:

$H_0 : \beta = 0$ and $H_1 : \beta \neq 0$

If a β is zero, it means there is no relationship between the respective independent variable and the dependent variable.

For all coefficients together we have the ***F*-test**:

$H_0 : \beta_1 = \beta_2 = \beta_3 = ... = 0$ and the alternative that not all of them are zero.

If the F-test is not significant, the model explains very little of the dependent variable.

(Insignificant t-tests in combination with a significant F-test is an indication of multicollinearity.)

Multicollinearity

When two or more variables in **X** are highly correlated:

- It will be harder to estimate **b**.
- Each individual **x** adds less to the prediction of **y**.

We can identify high multicollinearity by looking at **variance inflation factors**, or VIF scores—a VIF score of 4 or higher, approximately, raises concerns.

```
library(faraway)
m <- lm(y ~ x1 + x2 + x3, data)
vif(m)
```

Hastie, Trevor, Robert Tibshirani and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd edition ed. New York: Springer.

King, Gary. 1998. *Unifying political methodology. The likelihood theory of statistical inference*. University of Michigan Press.