Bivariate analysis: simple linear regression

2024-09-05

- **M1 MIDS/MFA**
- **Université Paris Cité**
- Année 2024-2025
- Course Homepage

- Moodle

Objectives

Quantitative bivariate samples and Simple linear regression

Numerical summaries

The numerical summary of a numerical bivariate sample consists of an *empirical mean*

$$\begin{pmatrix} \overline{X}_n \\ \overline{Y}_n \end{pmatrix} = \frac{1}{n} \sum_{i=1}^{n} \begin{pmatrix} x_i \\ y_i \end{pmatrix}$$

and an empirical covariance matrix

$$\begin{pmatrix} \mathrm{var}_n(X) & \mathrm{cov}_n(X,Y) \\ \mathrm{cov}_n(X,Y) & \mathrm{var}_n(Y) \end{pmatrix}$$

with

$$\mathrm{var}_n(X,Y) = \frac{1}{n} \sum_{k=1}^{n} \left(x_i - \overline{X}_n \right)^2$$

and

$$\mathrm{cov}_n(X,Y) = \frac{1}{n} \sum_{k=1}^{n} \left(x_i - \overline{X}_n \right) \times \left(y_i - \overline{Y}_n \right)$$

Covariance matrices have properties

The empirical covariance matrix is the *covariance matrix of the joint empirical distribution.*

As a covariance matrix, the empirical covariance matrix is *symmetric*, *semi-definite positive (SDP)*

Semi-definiteness

- A square $n \times n$ matrix A is semi-definite positive (SDP) iff

$$\forall u \in \mathbb{R}^n, \qquad u^T \times Au = \langle u, Au \rangle \geq 0$$

- A square $n \times n$ matrix A is definite positive (DP) iff

$$\forall u \in \mathbb{R}^n \setminus \{0\}, \qquad u^T \times Au = \langle u, Au \rangle > 0$$

Linear correlation coefficient

The *linear correlation coefficient* is defined from the covariance matrix as

$$\rho = \frac{\mathrm{cov}_n(X, Y)}{\sqrt{\mathrm{var}_n(X)\,\mathrm{var}_n(Y)}}$$

👉 By the Cauchy-Schwarz inequality, we always have

$$-1 \leq \rho \leq 1$$

👉 Translating and/or rescaling the columns does not modify the linear correlation coefficient!

Functions `cov` and `cor` from base R perform the computations

Visualizing quantitative bivariate samples

Suppose now, we want to visualize a quantitative bivariate sample of length n.

This bivariate sample (a dataframe) may be handled as a *real matrix* with n rows and 2 columns

Geometric concepts come into play

Exploring column space

We may attempt to visualize the two columns, that is the two n-dimensional vectors or the rows, that is n points on the real plane.

💡 If we try to visualize the two columns, we simplify the problem by *projecting on the plane generated by the two columns*

Then what matters is the *angle* between the two vectors.

Its *cosine* is precisely the *linear correlation coefficient* defined above

Exploring row space

If we try visualize the rows, the most basic visualization of a quantitative bivariate sample is the *scatterplot*.

In the grammar of graphics parlance, it consists in mapping the two variables on the two axes, and mapping rows to points using `geom_point` and `stat_identity`

A Gaussian cloud

We build an artificial bivariate sample, by first building a covariance matrix `COV` (it is randomly generated). Then we build a bivariate normal sample `s` of length `n` and turn it into a dataframe `u`. The dataframe is then fed to `ggplot`.

```
set.seed(1515) # for the sake of reproducibility

n <- 100
V <- matrix(rnorm(4, 1, 1), nrow = 2)
COV <- V %*% t(V)          # a random covariance matrix, COV is symmetric and SDP

s <- t(V %*% matrix(rnorm(2 * 10 * n), ncol=10*n))
u <- as_tibble(list(X=s[,1], Y=s[, 2]))                      # a bivariate normal sample

emp_mean <- as_tibble(t(colMeans(u)))
```

Numerical summary

- Mean vector (Empirical mean)

```
t(colMeans(u)) %>%
  knitr::kable(digits = 3, col.names = c("$\\overline{X_n}$", "$\\overline{Y_n}$"))
```

$\overline{X_n}$	$\overline{Y_n}$
0.004	-0.004

- Covariance matrix (Empirical covariance matrix)

```
cov(u) %>% as.data.frame() %>% knitr::kable(digits = 3)
```

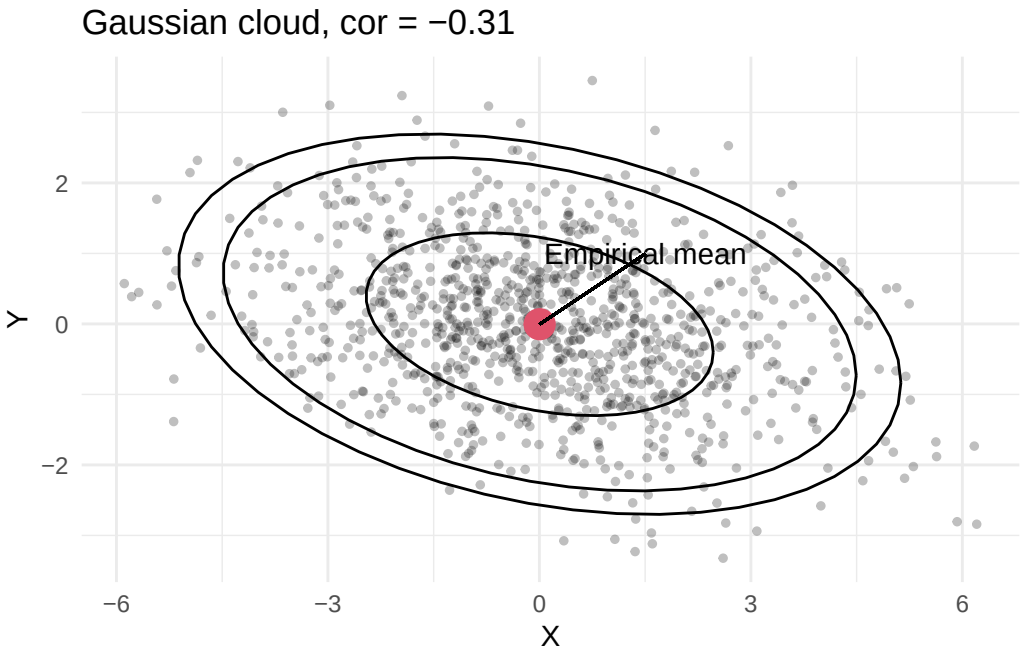
	X	Y
X	4.370	-0.706
Y	-0.706	1.212

Code

```
p_scatter_gaussian <- u %>%
  ggplot() +
  aes(x = X, y = Y) +
  geom_point(alpha = .25, size = 1) +
  geom_point(data = emp_mean, color = 2, size = 5) +
  stat_ellipse(type = "norm", level = .9) +
  stat_ellipse(type = "norm", level = .5) +
  stat_ellipse(type = "norm", level = .95) +
  annotate(geom="text", x=emp_mean$X+1.5, y= emp_mean$Y+1, label="Empirical mean")+
  geom_segment(aes(x=emp_mean$X, y=emp_mean$Y, xend=emp_mean$X+1.5, yend=emp_mean$Y+1)) +
  coord_fixed() +
  ggtitle(stringr::str_c("Gaussian cloud, cor = ",
    round(cor(u$X, u$Y), 2),
    sep = ""
  ))
```

```
p_scatter_gaussian
```

```
Warning in geom_segment(aes(x = emp_mean$X, y = emp_mean$Y, xend = emp_mean$X + : All aesth
i Please consider using `annotate()` or provide this layer with data containing
  a single row.
```



Qualitative and quantitative variables

Conditional summaries

For each modality $i \in \mathcal{X}$, we define:

- Conditional Mean of X given $\{X = i\}$

$$\overline{Y}_{n|i} = \frac{1}{n_i} \sum_{k \leq n} \mathbb{I}_{x_k = i} \times y_k$$

- Conditional Variance Y given $\{X = i\}$

$$\sigma^2_{Y|i} = \frac{1}{n_i} \sum_{k \leq n} \mathbb{I}_{x_k = i} \times \left(y_k - \overline{Y}_{n|i} \right)^2$$

Huygens-Pythagoras formula

$$\sigma^2_Y = \underbrace{\sum_{i \in \mathcal{X}} \frac{n_i}{n} \sigma^2_{Y|i}}_{\text{mean of conditional variances}} + \underbrace{\sum_{i \in \mathcal{X}} \frac{n_i}{n} (\overline{Y}_{n|i} - \overline{Y}_n)^2}_{\text{variance of conditional means}}$$

🧠 Check this

Robust bivariate summaries

It is also possible and fruitful to compute

- conditional quantiles (median, quartiles) and
- conditional interquartile ranges (IQR)

Conditional mean, variance, median, IQR ()

```
tit <-  readr::read_csv("../DATA/titanic/train.csv")
```

```
Rows: 891 Columns: 12
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (5): Name, Sex, Ticket, Cabin, Embarked
dbl (7): PassengerId, Survived, Pclass, Age, SibSp, Parch, Fare

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
tit <-  tit |>
    mutate(across(c(Survived, Pclass, Name, Sex, Embarked), as.factor))

tit |>
  glimpse()
```

```
Rows: 891
Columns: 12
$ PassengerId <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,~
$ Survived    <fct> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1~
$ Pclass      <fct> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3~
$ Name        <fct> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley (Fl~
$ Sex         <fct> male, female, female, female, male, male, male, male, fema~
$ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 14, ~
$ SibSp       <dbl> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1, 0~
$ Parch       <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0, 0~
$ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", "37~
$ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625,~
$ Cabin       <chr> NA, "C85", NA, "C123", NA, NA, "E46", NA, NA, NA, "G6", "C~
$ Embarked    <fct> S, C, S, S, S, Q, S, S, S, C, S, S, S, S, S, S, Q, S, S, C~
```

```
tit %>%
  dplyr::select(Survived, Fare) %>%
  dplyr::group_by(Survived) %>%
  dplyr::summarise(cmean=mean(Fare, na.rm=TRUE), #<<
                   csd=sd(Fare,na.rm = TRUE),
                   cmedian=median(Fare, na.rm = TRUE),
                   cIQR=IQR(Fare,na.rm = TRUE))
```

```
# A tibble: 2 x 5
  Survived cmean   csd cmedian  cIQR
  <fct>    <dbl> <dbl>   <dbl> <dbl>
1 0         22.1  31.4    10.5  18.1
2 1         48.4  66.6    26    44.5
```

Visualization of mixed bivariate samples

Visualization of qualitative/quantitative bivariate samples

consists in displaying visual summaries of conditional distribution of Y given $X = i, i \in \mathcal{X}$

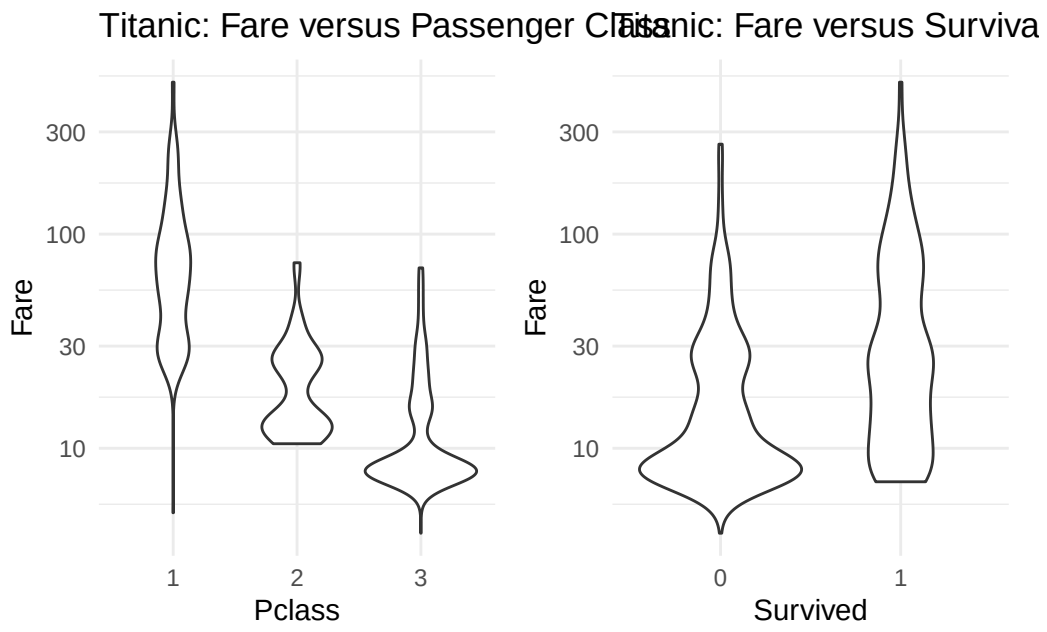
`Boxplots` and `violinplots` are relevant here

Mixed bivariate samples from Titanic (violine plots)

```
filtered_tit <- tit %>%
  dplyr::select(Pclass, Survived, Fare) %>%
  dplyr::filter(Fare > 0 )

v <- filtered_tit %>%
  ggplot() +
  aes(y=Fare) +
  scale_y_log10()

# vv <- v + geom_violin()

filtered_tit |>
  glimpse()
```

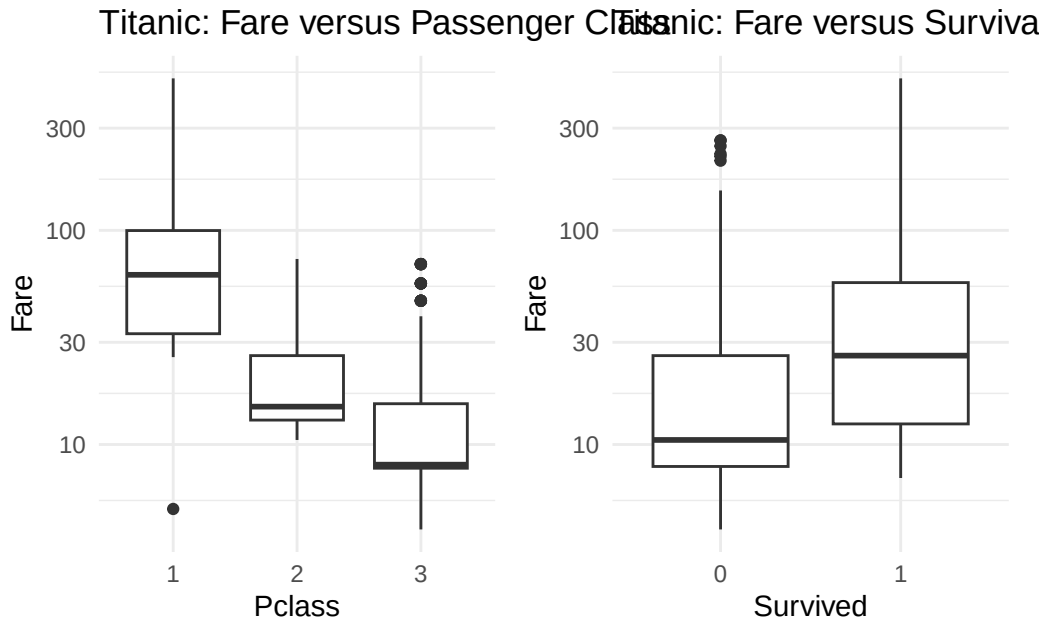
```
Rows: 876
Columns: 3
$ Pclass   <fct> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3, 3, 2~
$ Survived <fct> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0~
$ Fare     <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.8625, 21~
```

```
p <- v +
  aes(x=Pclass) +
  geom_violin() +
  ggtitle("Titanic: Fare versus Passenger Class")

q <- v +
  aes(x=Survived) +
  geom_violin() +
  ggtitle("Titanic: Fare versus Survival")

p + q
```



Mixed bivariate samples from Titanic (boxplots)

```
(
v + aes(x=Pclass) +
  geom_boxplot() +
  ggtitle("Titanic: Fare versus Passenger Class")
) + (
v +
  aes(x=Survived) +
  geom_boxplot() +
  ggtitle("Titanic: Fare versus Survival")
)
```



Dataset `whiteside` (from package `MASS` of R)

> Mr Derek Whiteside of the UK Building Research Station recorded the weekly gas consumption and average external temperature at his own house in south-east England for two heating seasons, one of 26 weeks before, and one of 30 weeks after cavity-wall insulation was installed. The object of the exercise was to assess the effect of the insulation on gas consumption.

Dataset `whiteside`

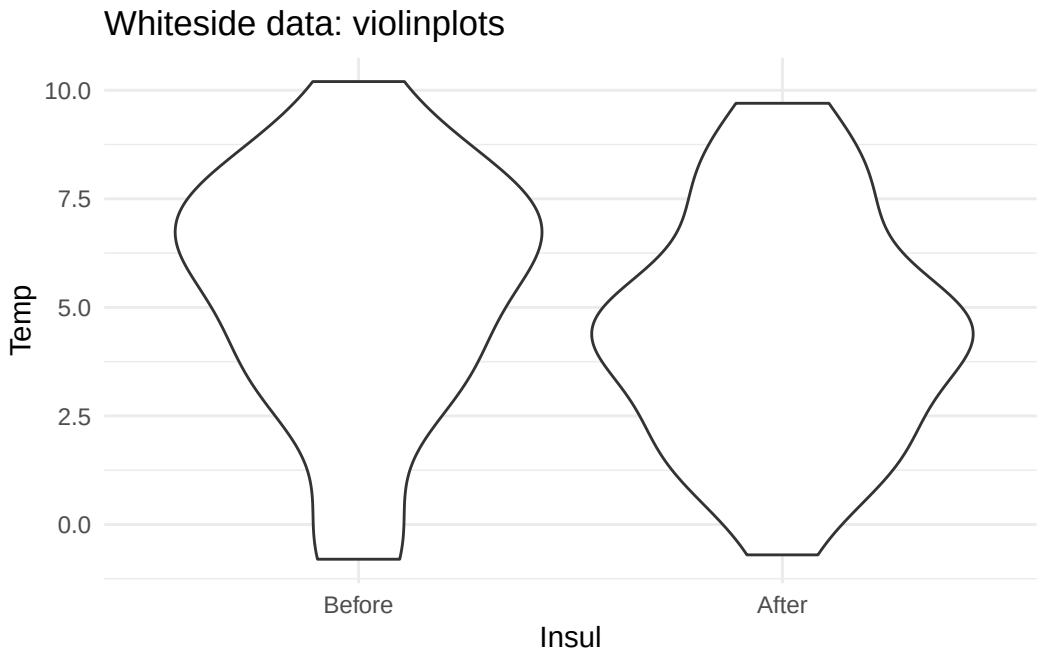
`Gas` and `Temp` are both quantitative variables while `Insul` is qualitative with two modalities (`Before`, `After`).

`Insul` A factor, before or after insulation.

`Temp` Purportedly the average outside temperature in degrees Celsius. (These values is far too low for any 56-week period in the 1960s in South-East England. It might be the weekly average of daily minima.)

`Gas` The weekly gas consumption in 1000s of cubic feet.

```
MASS::whiteside %>%
  ggplot(aes(x=Insul, y=Temp)) +
  geom_violin() +
  ggtitle("Whiteside data: violinplots")
```



Simple linear regression

- We now explore association between *two* quantitative variables
- We investigate the association between two quantitative variables as a *prediction* problem
- We aim at predicting the value of Y as a function of X.
- We restrict our attention to linear/affine prediction.

We look for $a, b \in \mathbb{R}$ such that $y_i \approx ax_i + b$

Making $\approx$ meaningful compels us to choose a *goodness of fit* criterion.

Several criteria are possible, for example:

$$\begin{aligned} \text{Mean absolute deviation} &= \frac{1}{n}\sum_{i=1}^{n} \left|y_i - ax_i - b\right| \\ \text{Mean quadratic deviation} &= \frac{1}{n}\sum_{i=1}^{n} \left|y_i - ax_i - b\right|^2 \end{aligned}$$

In their days, Laplace championed the mean absolute deviation, while Gauss advocated the mean quadratic deviation. For computational reasons, we focus on minimizing the mean quadratic deviation.

> The fourth chapter of Laplace treatise includes an exposition of the *method of least squares*, a remarkable testimony to Laplace's command over the processes of analysis.
> In 1805 Legendre had published the *method of least squares*, making no attempt to tie it to the theory of probability.
>
> In 1809 Gauss had derived the *normal distribution* from the principle that the arithmetic mean of observations gives the most probable value for the quantity measured; then, turning this argument back upon itself, he showed that, if the errors of observation are normally distributed, the *least squares estimates* give the most probable values for the coefficients in *regression* situations

Least Square Regression

Minimizing a cost function

The *Least Square Regression* problem consists of minimizing with respect to (a, b) :

$$\begin{aligned}\ell_n(a,b) &= \sum_{i=1}^n \left(y_i - ax_i - b\right)^2 \\ &= \sum_{i=1}^n \left((y_i - \overline{Y}_n) - a(x_i - \overline{X}_n) + \overline{Y}_n - a\overline{X}_n - b\right)^2 \\ &= \sum_{i=1}^n \left((y_i - \overline{Y}_n) - a(x_i - \overline{X}_n)\right)^2 + n(\overline{Y}_n - a\overline{X}_n - b)^2\end{aligned}$$

Deriving the solution

The function to be minimized is smooth and strictly convex over $\mathbb{R}^2$: a unique minimum is attained where the gradient vanishes

It is enough to compute the partial derivatives.

$$\begin{aligned}\frac{\partial \ell_n}{\partial a} &= -2\,\mathrm{cov}(X,Y) + 2a\,\mathrm{var}(X) - 2n(\overline{Y}_n - a\overline{X}_n - b)\overline{X}_n \\ \frac{\partial \ell_n}{\partial b} &= -2n(\overline{Y}_n - a\overline{X}_n - b)\end{aligned}$$

A closed-form solution

Zeroing partial derivatives leads to

$$\begin{aligned}\hat{a} &= \frac{\mathrm{cov}(X,Y)}{\mathrm{var}(X)} \\ \hat{b} &= \overline{Y}_n - \frac{\mathrm{cov}(X,Y)}{\mathrm{var}(X)}\overline{X}_n\end{aligned}$$

or

$$\begin{aligned}\hat{a} &= \rho\frac{\sigma_y}{\sigma_x} \\ \hat{b} &= \overline{Y}_n - \rho\frac{\sigma_y}{\sigma_x}\overline{X}_n\end{aligned}$$

☛ If the sample were standardized, that is, if X (resp. Y) were divided by σ_X (resp. σ_Y), the slope of the regression line would be the correlation coefficient

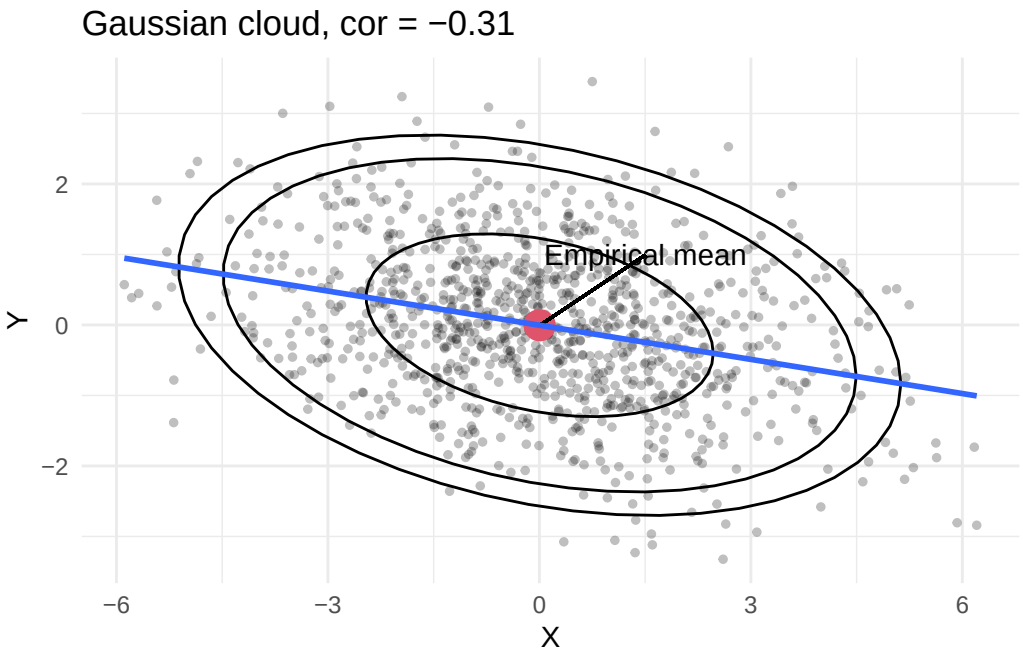
Overplotting the Gaussian cloud

- The *slope* and *intercept* can be computed from the sample summary (empirical mean and covariance matrix)
- In higher dimension, coefficients are from `lm(...)`

```
p_scatter_gaussian +
  geom_smooth(method="lm", se=FALSE)
```

```
Warning in geom_segment(aes(x = emp_mean$X, y = emp_mean$Y, xend = emp_mean$X + : All aesth
i Please consider using `annotate()` or provide this layer with data containing
  a single row.
```

```
`geom_smooth()` using formula = 'y ~ x'
```



lm(formula, data)

```
mod <- lm(formula=Y ~ X, data=u)

mod %>% summary()
```

```
Call:
lm(formula = Y ~ X, data = u)

Residuals:
    Min      1Q  Median      3Q     Max
-3.0168 -0.7106 -0.0079  0.7294  3.5773

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.003685   0.033145  -0.111    0.911
X           -0.161562   0.015864 -10.184   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.048 on 998 degrees of freedom
Multiple R-squared:  0.09415,   Adjusted R-squared:  0.09324
F-statistic: 103.7 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
sqrt(sum((mod$residuals)^2)/(mod$df.residual))
```

```
[1] 1.048131
```

```
cor(u)^2
```

```
           X          Y
X 1.00000000 0.09414501
Y 0.09414501 1.00000000
```

Residuals

The *residuals* are the prediction errors $\left(y_i - \hat{a}x_i - \hat{b}\right)_{i\leq n}$

Residuals play a central role in *regression diagnostic*

The `Residual Standard Error`, is the square root of the normalized sum of squared residuals:

$$\frac{1}{n-2}\sum_{i=1}^{n}\left(y_i - \hat{a}x_i - \hat{b}\right)^2$$

The normalization coefficient is the number of rows n diminished by the number of adjusted parameters (the so-called *degrees of freedom*)

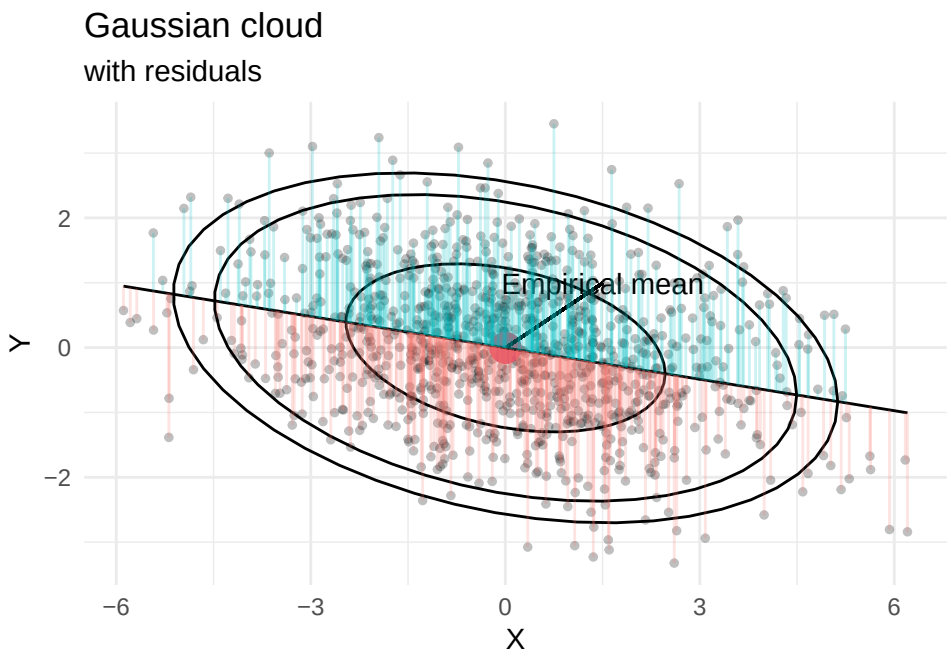
```
sqrt(sum((mod$residuals)^2)/(mod$df.residual))
```

```
[1] 1.048131
```

👉 This makes sense if we adopt a modeling perspective, if we accept the *Gaussian Linear Models* assumptions from the Statistical Inference course

```
p_scatter_gaussian %+%
  broom::augment(lm(Y ~ X, u)) +  #<<
  geom_line(aes(x=X, y=.fitted)) +
  geom_segment(aes(x=X, xend=X, y=.fitted, yend=Y,
                   color=forcats::as_factor(sign(.resid))),
               alpha=.2) +
  theme(legend.position = "None") +
  ggtitle("Gaussian cloud",subtitle = "with residuals")
```

```
Warning in geom_segment(aes(x = emp_mean$X, y = emp_mean$Y, xend = emp_mean$X + : All aesth
i Please consider using `annotate()` or provide this layer with data containing
  a single row.
```



The residuals are the lengths of the segments connecting sample points to their projections on the regression line

Technically, the `Multiple R-squared` or *coefficient of determination* is the squared empirical correlation coefficient ρ^2 between the explanatory and the response variables (in simple linear regression)

$$1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{a}x_i - \hat{b}\right)^2}{\sum_{i=1}^{n} \left(y_i - \overline{Y}_n\right)^2} = 1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n} \left(y_i - \overline{Y}_n\right)^2}$$

```
cor(u$X, u$Y)^2
```

```
[1] 0.09414501
```

```
(sum(u$X*u$Y)/nrow(u) - mean(u$X)* mean(u$Y))*(nrow(u)/(nrow(u)-1))
```

```
[1] -0.7059866
```

```
((988/999)*cov(u$X, u$Y)/sqrt(var(u$X)*var(u$Y)))^2
```

```
[1] 0.09208316
```

It is also understood as the share of the variance of the response variable that is *explained* by the explanatory variable

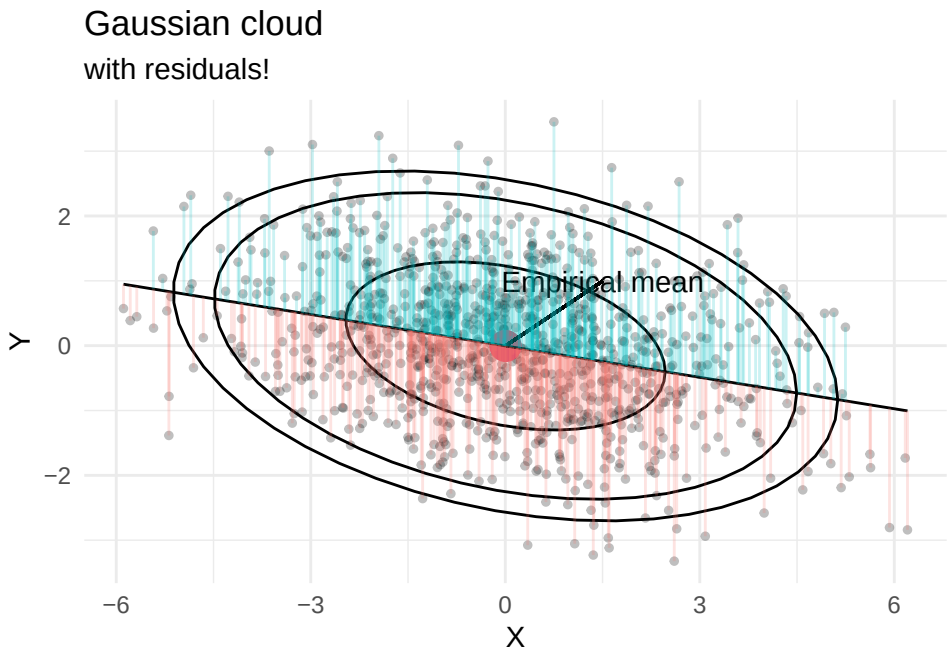
The `Adjusted R-squared` is a deflated version of `Multiple R-squared`

$$1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{a}x_i - \hat{b}\right)^2 /(n-p-1)}{\sum_{i=1}^{n} \left(y_i - \overline{Y}_n\right)^2 /(n-1)}$$

It is useful when comparing the merits of several competing models (this takes us beyond the scope of this lesson)

```
p_scatter_gaussian %+%
  broom::augment(lm(Y ~ X, u)) +  #<<
  geom_line(aes(x=X, y=.fitted)) +
  geom_segment(aes(x=X,
                   xend=X,
                   y=.fitted,
                   yend=Y,
                   color=as_factor(sign(.resid))),
               alpha=.2) +
  theme(legend.position = "None") +
  ggtitle("Gaussian cloud",
          subtitle = "with residuals!")
```

```
Warning in geom_segment(aes(x = emp_mean$X, y = emp_mean$Y, xend = emp_mean$X + : All aesth
i Please consider using `annotate()` or provide this layer with data containing
  a single row.
```



$y = x^T\beta + \sigma\epsilon$: The biggest lie?

- Any numeric bivariate sample can be fed to `lm`
- Whatever the bivariate dataset, you will obtain a linear prediction model
- It is not wise to rely exclusively on the `Multiple R-squared` to assess a linear model
- 👉 Different datasets can lead to the same regression line and the same `Multiple R-squared` and the same `Adjusted R-squared`

Anscombe quartet

4 simple linear regression problems packaged in dataframe `datasets::anscombe`

- `y1 ~ x1`
- `y2 ~ x2`
- `y3 ~ x3`
- `y4 ~ x4`

```
anscombe <- datasets::anscombe

anscombe %>%
    gt::gt()
```

x1	x2	x3	x4	y1	y2	y3	y4
10	10	10	8	8.04	9.14	7.46	6.58
8	8	8	8	6.95	8.14	6.77	5.76
13	13	13	8	7.58	8.74	12.74	7.71
9	9	9	8	8.81	8.77	7.11	8.84
11	11	11	8	8.33	9.26	7.81	8.47
14	14	14	8	9.96	8.10	8.84	7.04
6	6	6	8	7.24	6.13	6.08	5.25
4	4	4	19	4.26	3.10	5.39	12.50
12	12	12	8	10.84	9.13	8.15	5.56
7	7	7	8	4.82	7.26	6.42	7.91

5	5	5	8	5.68	4.74	5.73	6.89

Anscombe quartet: 4 datasets, 1 linear fit with almost identical goodness of fits

```
lm(y1 ~ x1, anscombe) %>% summary
```

```
Call:
lm(formula = y1 ~ x1, data = anscombe)

Residuals:
     Min       1Q   Median       3Q      Max
-1.92127 -0.45577 -0.04136  0.70941  1.83882

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0001     1.1247   2.667  0.02573 *
x1            0.5001     0.1179   4.241  0.00217 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6665,    Adjusted R-squared:  0.6295
F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.00217
```

```
lm(y2 ~ x2, anscombe) %>% summary
```

```
Call:
lm(formula = y2 ~ x2, data = anscombe)

Residuals:
    Min      1Q  Median      3Q     Max
-1.9009 -0.7609  0.1291  0.9491  1.2691

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.001      1.125   2.667  0.02576 *
x2             0.500      0.118   4.239  0.00218 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6662,    Adjusted R-squared:  0.6292
F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002179
```

```
lm(y3 ~ x3, anscombe) %>% summary
```

```
Call:
lm(formula = y3 ~ x3, data = anscombe)

Residuals:
    Min      1Q  Median      3Q     Max
-1.1586 -0.6146 -0.2303  0.1540  3.2411

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0025     1.1245   2.670  0.02562 *
x3            0.4997     0.1179   4.239  0.00218 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.236 on 9 degrees of freedom
Multiple R-squared:  0.6663,    Adjusted R-squared:  0.6292
F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002176
```

```
lm(y4 ~ x4, anscombe) %>% summary
```

```
Call:
lm(formula = y4 ~ x4, data = anscombe)

Residuals:
   Min     1Q Median     3Q    Max
-1.751 -0.831  0.000  0.809  1.839

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0017     1.1239   2.671  0.02559 *
x4            0.4999     0.1178   4.243  0.00216 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.236 on 9 degrees of freedom
Multiple R-squared:  0.6667,    Adjusted R-squared:  0.6297
F-statistic:    18 on 1 and 9 DF,  p-value: 0.002165
```

All four numerical summaries look similar:

- `Intercept` ≈ 3.0017
- `slope` ≈ 0.5
- Residual standard error ≈ 1.236
- Multiple R-squared $\approx .67$
- F-statistic ≈ 18

n is equal to 11

The number of adjusted parameters p is 2 The number of degrees of freedom is $n - p = 9$

How is RSE computed ?

$$\frac{1}{n-p}\sum_{i=1}^{n}\left(y_j[i] - \hat{y}_j[i]\right)^2$$

> ! Visual inspection of the data reveals that some linear models are more relevant than others

This is the message of the Anscombe quartet.

It is made of four bivariate samples with $n = 11$ individuals.

```
datasets::anscombe %>%
  pivot_longer(everything(),  #<<
```

```
    names_to = c(".value", "group"), #<<
    names_pattern = "(.)(.)" #<<
  )  %>%
  rename(X=x, Y=y) %>%
  arrange(group)-> anscombe_long
```

From https://tidyr.tidyverse.org/articles/pivot.html

Performing regression per group

For each value of `group` we perform a linear regression of `Y` versus `X`

```
list_lm <- purrr::map(anscombe_long$group ,
                      .f = \(x) lm(Y ~ X,
                                   anscombe_long,
                                   subset = anscombe_long$group==x))
```

🪄 Don't Repeat Yourself (DRY)

We use *functional programming*: `purrr::map(.l, .f)` where

- `.l` is a list
- `.f` is a function to be applied to each item of list `.l` or a `formula` to be evaluated on each list item

`purrr` package

Inspecting summaries

All four regressions lead to the same intercept and the same slope

All four regressions have the same Sum of Squared Residuals

All four regressions have the same Adjusted R-square

We are tempted to conclude that

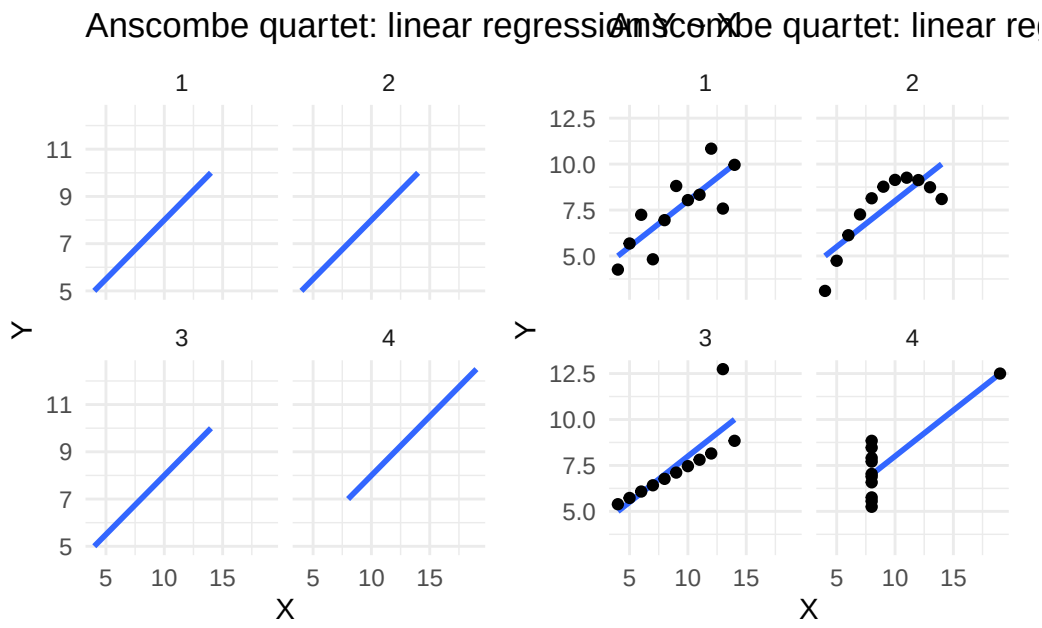
> all four linear regressions are equally relevant

Plotting points and lines helps dispel this illusion

Unveiling points

```
p <- anscombe_long %>%
  ggplot(aes(x=X, y=Y)) +
  geom_smooth(method="lm", se=FALSE) +  #<<
  facet_wrap(~ group) +                 #<<
  ggtitle("Anscombe quartet: linear regression Y ~ X")

(p + (p + geom_point()))
```

```
`geom_smooth()` using formula = 'y ~ x'
`geom_smooth()` using formula = 'y ~ x'
```



Among the four datasets, only the two left ones are righteously handled using simple linear regression

The bottom left dataset outlines the impact of *outliers* on Least Squares Minimization

Regression on the Whiteside data

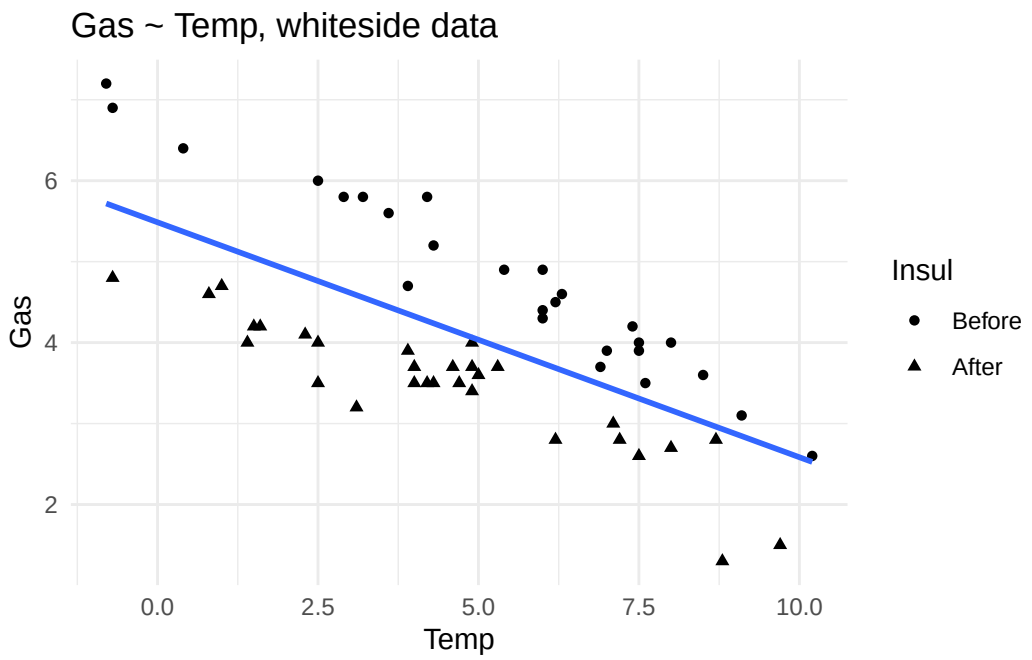
```
whiteside <- MASS::whiteside
lm0 <- whiteside %>%
  lm(Gas ~ Temp, .)
```

```
lm0 |>
  broom::tidy() |>
  gt::gt() |>
  gt::fmt_number(
    columns = -term,
    decimals=2
  )
```

term	estimate	std.error	statistic	p.value
(Intercept)	5.49	0.24	23.28	0.00
Temp	−0.29	0.04	−6.88	0.00

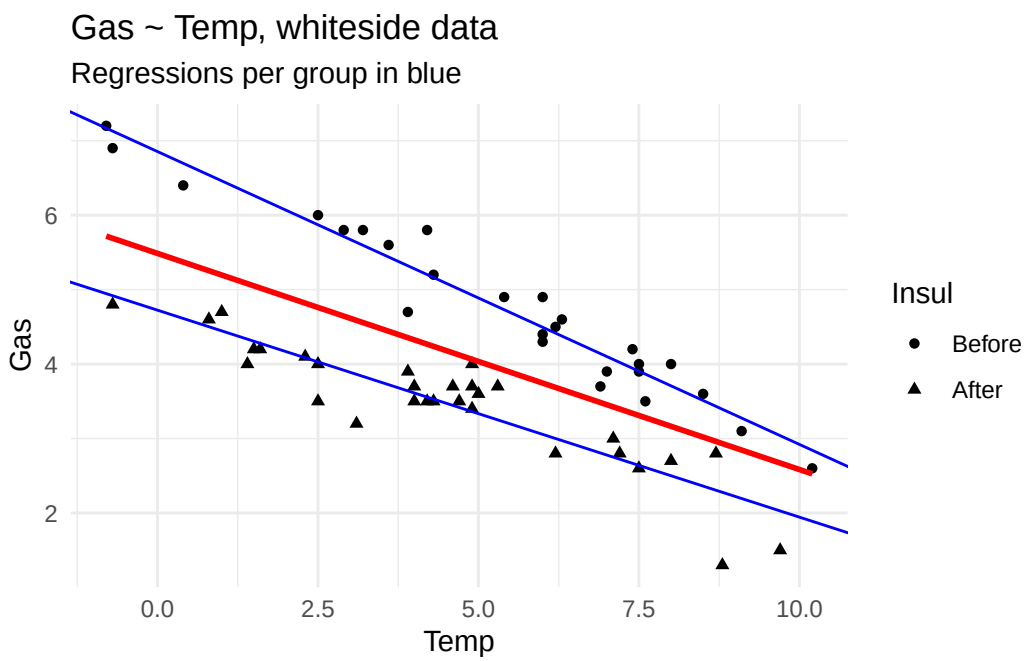
```
p <- lm0 |>
  broom::augment(data=whiteside) |>
  ggplot() +
  aes(x=Temp, y=Gas) +
  geom_point(aes(shape=Insul))

p +
  geom_smooth(
    formula = y ~ x,
    method="lm",
    se=FALSE
```

```
    ) +
  ggtitle("Gas ~ Temp, whiteside data")
```




```
lm_before <- whiteside %>%
  filter(Insul=="Before") %>%
  lm(Gas ~ Temp, .)

lm_after <- whiteside %>%
  filter(Insul=="After") %>%
  lm(Gas ~ Temp, .)
```

```
p +
  geom_smooth(
    formula = y ~ x,
    method="lm",
    se=FALSE,
    color="red",
    ) +
  geom_abline(
    intercept=coefficients(lm_before)[1],
    slope=coefficients(lm_before)[2],
    color='blue'
  ) +
  geom_abline(
    intercept=coefficients(lm_after)[1],
    slope=coefficients(lm_after)[2],
    color='blue'
  ) +
  labs(
    title="Gas ~ Temp, whiteside data",
    subtitle="Regressions per group in blue"
    )
```



Questions

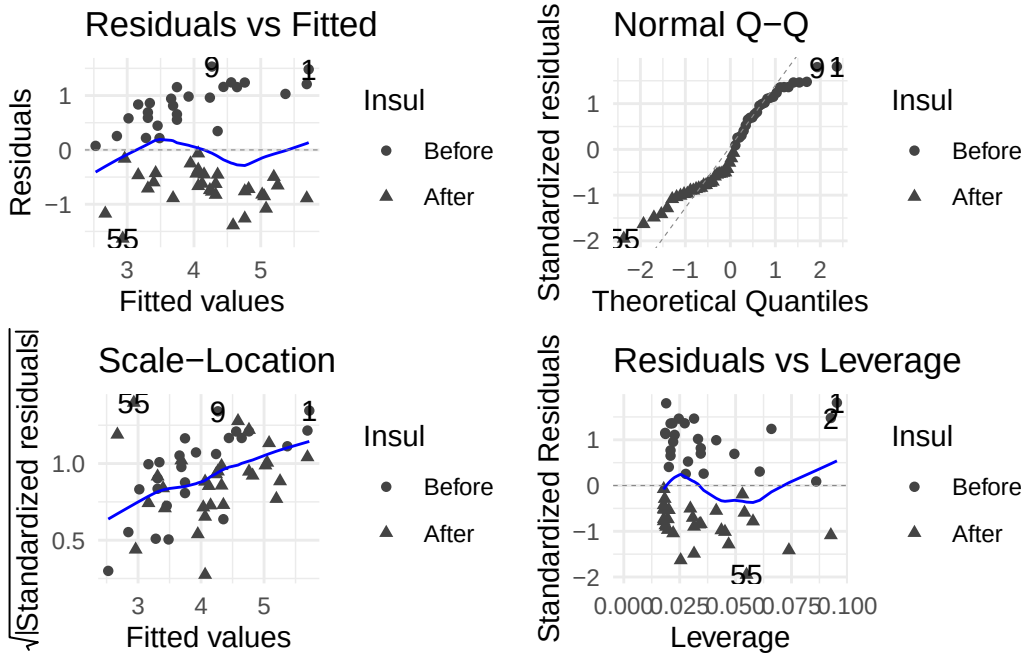
- Which regression should we trust?
- Can we build confidence interval for estimated coefficients?
- Can we estimate noise intensity?
- Can we trust the homoschedasticity assumption?
- Can we trust

Using diagnostic plots

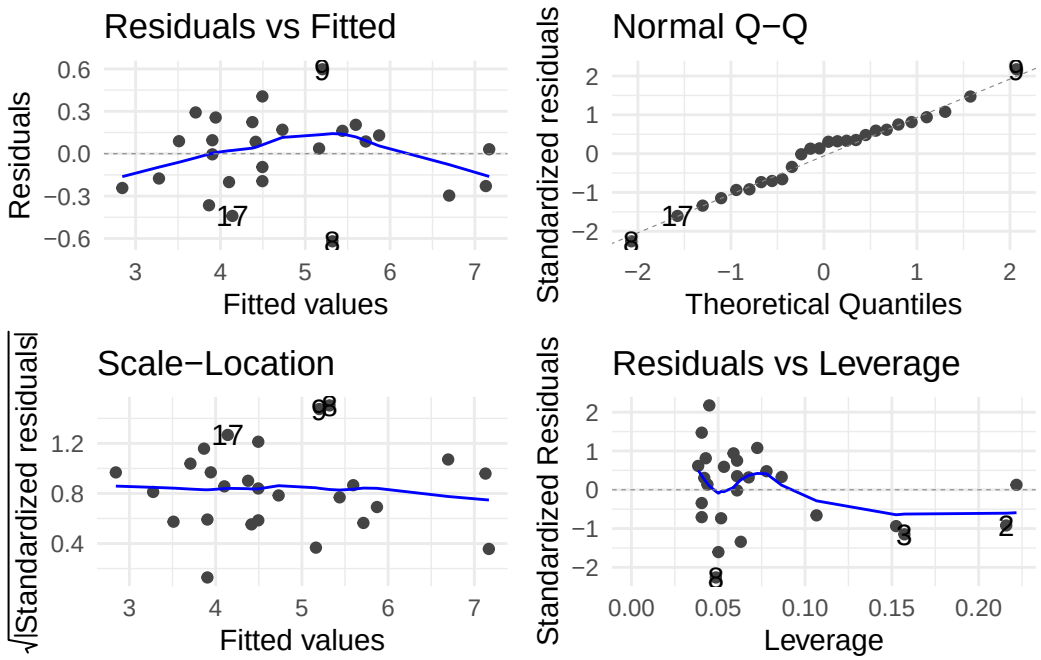
```
require(ggfortify)
```

```
Loading required package: ggfortify
```

```
autoplot(lm0, data=whiteside, shape='Insul')
```



```
autoplot(lm_before)
```




```
autoplot(lm_after)
```