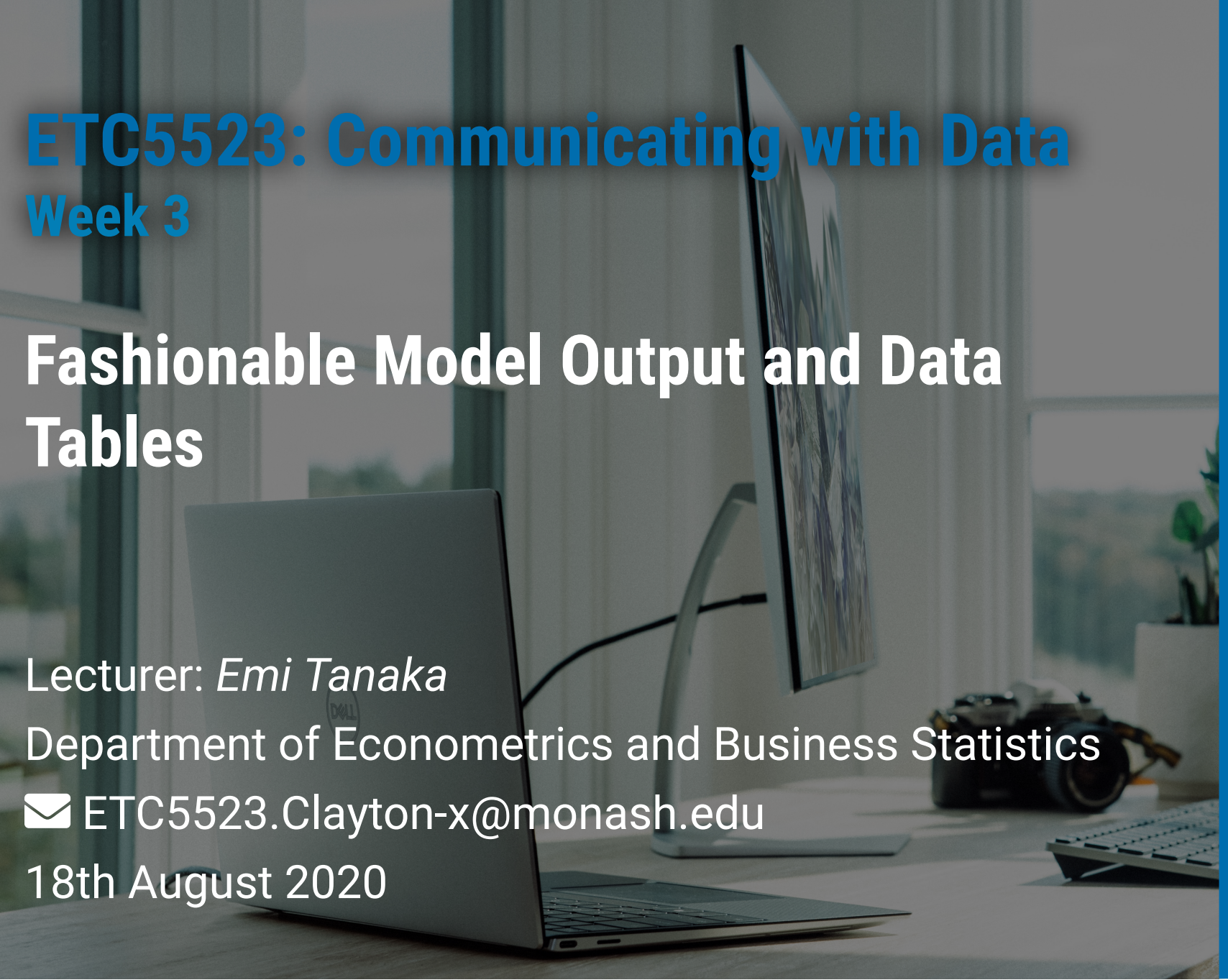


ETC5523: Communicating with Data

Week 3

Fashionable Model Output and Data Tables

Lecturer: *Emi Tanaka*
Department of Econometrics and Business Statistics
ETC5523.Clayton-x@monash.edu
18th August 2020

Housekeeping

- Take-home assessment will be released **Thu 20 Aug 7PM** on Moodle.
- Instructor's (i.e. me) consultation hour will be shifted to **12.30-1.30PM Thu** for *this week alone* so any troubleshooting can be sorted out prior to the start of the assessment.

🎯 What today is *not* about

- In this lecture, we are *not* overly concerned with whether the model is the best, good or bad; nor about the correctness of the interpretations of the statistical outputs.
- You learn about:
  - Bayesian models in ETC5242,
  - forecasting models in ETC5550,
  - machine learning models in ETC5250 and
  - advanced statistical models in ETC5580.

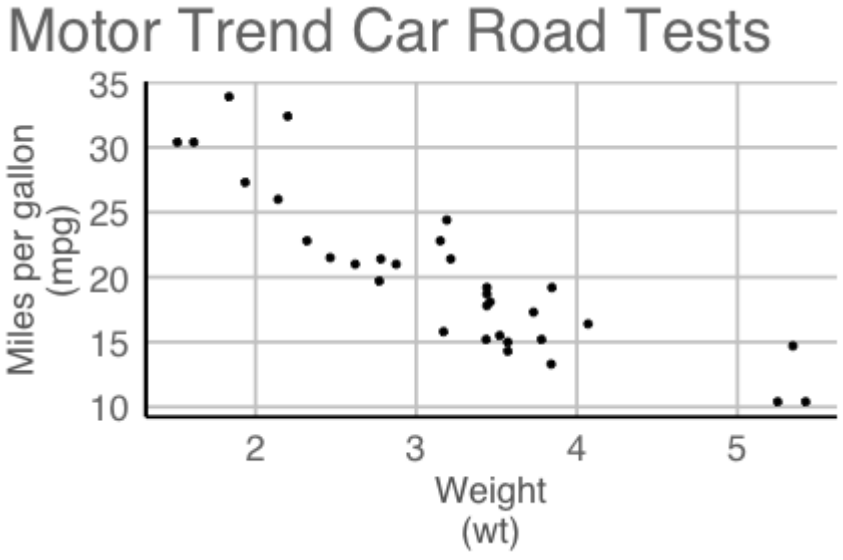
📅 So what is today about?

1. Working with model objects
2. Presenting information as a table

Statistical models

- All models are approximations of the unknown data generating process - whether how good of an approximation depends on the collected data and the model choice.

Example 1



Aim: characterise `mpg` in terms of `wt`.

- We fit the model:

$$\texttt{mpg}_i = \beta_0 + \beta_1 \texttt{wt}_i + e_i$$

  - ▶ Parameter details

- In R we fit this as

```
fit <- lm(mpg ~ wt, data = mtcars)
```

which is the same as

```
fit <- lm(mpg ~ 1 + wt, data = mtcars)
```

Extracting information from the fitted model Part 1/4

- When you fit a model, there would be a number of information you will be interested in extracting from the fit including:
  - the model parameter estimates,
  - model-related summary statistics, e.g. R^2, AIC and BIC,
  - model-related values, e.g. residuals, fitted values and predictions.
- So how do you extract these values from the `fit`?
- What does `fit` even contain?
- Any programming language will usually have:
  - a function to see the structure of the object, in R this is `str()`, and
  - methods that can be applied to classes of that object.

Extracting information from the fitted model Part 2/4

```r
fit <- lm(mpg ~ 1 + wt, data = mtcars)
fit
```

```
##
## Call:
## lm(formula = mpg ~ 1 + wt, data = mtcars)
##
## Coefficients:
## (Intercept)           wt
##      37.285       -5.344
```

LIVE DEMO

```r
str(fit)
```

```
## List of 12
##  $ coefficients : Named num [1:2] 37.29 -5.3
##   ..- attr(*, "names")= chr [1:2] "(Intercep
##  $ residuals    : Named num [1:32] -2.28 -0.
##   ..- attr(*, "names")= chr [1:32] "Mazda RX
##  $ effects      : Named num [1:32] -113.65 -
##   ..- attr(*, "names")= chr [1:32] "(Interce
##  $ rank         : int 2
##  $ fitted.values: Named num [1:32] 23.3 21.9
##   ..- attr(*, "names")= chr [1:32] "Mazda RX
##  $ assign       : int [1:2] 0 1
##  $ qr           :List of 5
##   ..$ qr   : num [1:32, 1:2] -5.657 0.177 0.
##   ..- attr(*, "dimnames")=List of 2
```

scroll

Extracting information from the fitted model Part 3/4

Accessing model parameter estimates:

```
fit$coefficients

## (Intercept)          wt
##   37.285126   -5.344472

# OR using
coef(fit)

## (Intercept)          wt
##   37.285126   -5.344472
```

This gives us the estimates of β_0 and β_1. But what about σ^2?

```
sigma(fit)^2

## [1] 9.277398
```

Extracting information from the fitted model Part 4/4

```
summary(fit)
```

```
##
## Call:
## lm(formula = mpg ~ 1 + wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 ' ' 0.1
```

scroll

So how do I get these summary values out?

Model objects to tidy data

```
broom::tidy(fit)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic
##   <chr>          <dbl>     <dbl>     <dbl>
## 1 (Intercept)    37.3      1.88      19.9  8
## 2 wt             -5.34     0.559     -9.56 1
```

```
broom::glance(fit)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p
##       <dbl>         <dbl> <dbl>     <dbl>
## 1     0.753         0.745  3.05      91.4 1.
## # … with 3 more variables: deviance <dbl>, d
```

```
broom::augment(fit)
```

```
## # A tibble: 32 x 9
##    .rownames          mpg    wt .fitted .res
##    <chr>            <dbl> <dbl>   <dbl>  <db
##  1 Mazda RX4         21    2.62    23.3 -2.2
##  2 Mazda RX4 Wag     21    2.88    21.9 -0.9
##  3 Datsun 710        22.8  2.32    24.9 -2.0
##  4 Hornet 4 Drive    21.4  3.22    20.1  1.3
##  5 Hornet Sportabo…  18.7  3.44    18.9 -0.2
##  6 Valiant           18.1  3.46    18.8 -0.6
##  7 Duster 360        14.3  3.57    18.2 -3.9
##  8 Merc 240D         24.4  3.19    20.2  4.1
##  9 Merc 230          22.8  3.15    20.5  2.3
## 10 Merc 280          19.2  3.44    18.9  0.3
## # … with 22 more rows
```

Modelling categorical variables Part 1/3

Experiment data: tooth growth in guinea pigs with **two supplement types** (orange juice of ascorbic acid) with **three different doses** (0.5, 1 and 2 mg/day).

```
df <- ToothGrowth %>% mutate(dose = as.factor(dose))
glimpse(df)

## Rows: 60
## Columns: 3
## $ len  <dbl> 4.2, 11.5, 7.3, 5.8, 6.4, 10.0, 11.2, 11.2, 5.2, 7.0, 16.5,
## $ supp <fct> VC, VC, VC, VC, VC, VC, VC, VC, VC, VC, VC, VC, VC, VC, VC,
## $ dose <fct> 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 1, 1, 1,
```

Suggested model and its fit:

```
fitc <- lm(sqrt(len) ~ 1 + supp + dose + supp:dose, data = df)
```

Modelling categorical variables Part 2/3

```
broom::tidy(fitc)
```

```
## # A tibble: 6 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     3.59      0.139    25.8   4.94e-32
## 2 suppVC         -0.806     0.197    -4.09  1.44e- 4
## 3 dose1           1.15      0.197     5.86  2.83e- 7
## 4 dose2           1.51      0.197     7.65  3.64e-10
## 5 suppVC:dose1    0.144     0.279     0.518 6.07e- 1
## 6 suppVC:dose2    0.800     0.279     2.87  5.82e- 3
```

- Where is dose 0.5? And supplementary OJ?
- The first levels of dose (0.5) and supp (OJ) have been constrained to zero; in another words, these are the **reference level** (also called **base-line category**).

Modelling categorical variables Part 3/3

```
broom::tidy(fitc) %>%
  tidycat::tidy_categorical(fitc)
```

```
## # A tibble: 12 x 9
##    term   estimate std.error statistic  p.value variable  level effect r
##    <chr>     <dbl>     <dbl>     <dbl>    <dbl> <chr>     <fct> <chr>  <
##  1 (Inte…    3.59      0.139    25.8   4.94e-32 (Interce… (Int… main   N
##  2 <NA>      0         0         0     0.       supp      OJ    main   B
##  3 suppVC   -0.806     0.197    -4.09  1.44e- 4 supp      VC    main   N
##  4 <NA>      0         0         0     0.       dose      0.5   main   B
##  5 dose1     1.15      0.197     5.86  2.83e- 7 dose      1     main   N
##  6 dose2     1.51      0.197     7.65  3.64e-10 dose      2     main   N
##  7 <NA>      0         0         0     0.       supp:dose OJ:0… inter… B
##  8 <NA>      0         0         0     0.       supp:dose VC:0… inter… N
##  9 <NA>      0         0         0     0.       supp:dose OJ:1  inter… N
## 10 suppV…    0.144     0.279     0.518 6.07e- 1 supp:dose VC:1  inter… N
## 11 <NA>      0         0         0     0.       supp:dose OJ:2  inter… N
## 12 suppV…    0.800     0.279     2.87  5.82e- 3 supp:dose VC:2  inter… N
```

Why have constraints/reference/base-line categories?

- Let's consider these simultaneous equations:

$$\begin{aligned} 20\beta_0 + 10\beta_1 + 10\beta_2 &= 50 \\ 10\beta_0 + 10\beta_1 &= 20 \\ 10\beta_0 + 10\beta_2 &= 30 \end{aligned} \quad \text{or} \quad \begin{bmatrix} 20 & 10 & 10 \\ 10 & 10 & 0 \\ 10 & 0 & 10 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 50 \\ 20 \\ 30 \end{bmatrix}$$

- A solution is $\beta_0 = \beta_1 = 1$ and $\beta_2 = 2$. But so is $\beta_0 = -1$, $\beta_1 = 3$ and $\beta_2 = 4$. And so is $\beta_0 = 2, \beta_1 = 0$ and $\beta_2 = 1$.
- There are infinite number of solutions.
- 💡 The idea is the same when you have a categorical variable and the effects for each level of the categorical variables are estimated using least squares estimate (or maximum likelihood estimate) then without constraints, there are infinite number of solutions.

Modelling in R

- There are many R-packages that fit all kinds of models, e.g.
  - `mgcv` fits generalized additive models,
  - `glmnet` fits a generalised linear models with elastic net,
  - `rstanarm` fits Bayesian regression models using Stan,
  - `BLR` also fits Bayesian linear regression models, and
  - `fable` (written by your tutor Mitch O'Hara-Wild) fits forecast models,
  - many other contributions by the community.
- There are a lot of new R-packages contributed; some implementing the latest research results. This means that if you want to use the state-of-the-art research, then you need to work with model objects beyond the standard `lm` and `glm`.

Example with Baysian regression

```
library(rstanarm)
fit_stan <- stan_lm(mpg ~ 1 + wt, data = mtcars,
                    prior = R2(0.7528, what = "mean"))

broom::tidy(fit_stan)

## Error: $ operator is invalid for atomic vectors
```

Huh?

```
coef(fit_stan)

## (Intercept)          wt
##   30.477078   -3.227948
```

Okay this works, but why not for `broom::tidy`?

S3 Object classes

- So how do you find out the functions that work with model objects?
- First notice the class of the object `fit`:

```
class(fit)

## [1] "lm"
```

- The methods associated with this can be found using:

```
methods(class = "lm")

##  [1] add1           alias          anova          case.names     coer
##  [6] confint        cooks.distance deviance       dfbeta         dfbe
## [11] drop1          dummy.coef     effects        extractAIC     fami
## [16] formula        fortify        hatvalues      influence      init
```

scroll

Where is `coef()`?

Object oriented programming (OOP) Part 1/3

- You need to know a little bit about OOP if you want to know more about how model objects work and how methods functions extract information from the model objects.
- The ***S3 system*** is the most widely used OOP system in R but there are other OOP systems in R, e.g. the S4 system is used for model objects in `lme4` R-package, but it will be out of scope for this course.
- Let's have a look inside of `coef` function:

```
coef

## function (object, ...)
## UseMethod("coef")
## <bytecode: 0x7ffe573673f8>
## <environment: namespace:stats>
```

Try also looking inside `lm`

S3 Object oriented programming (OOP) Part 2/3

- Suppose I create a new class called `student`:

```
obj <- structure(list(who = "🤪", quiz = 29, exam = 30),
                 class = "student")
class(obj)

## [1] "student"
```

- Here I create a simple generic method, called `quiz`, for the `student` class to extract the student object's quiz mark:

```
# the generic
get_quiz <- function(object, ...) { UseMethod("get_quiz") }
# the method
get_quiz.student <- function(object, ...) { return(object$quiz) }
# implementation of the method
get_quiz(obj)
```

S3 Object oriented programming (OOP) Part 3/3

- A method is created by using the form `generic.class`.
- When using a method for `class`, you can omit the `.class` from the function.
- E.g. `residuals(fit)` is the same as `residuals.lm(fit)` since the class of `fit` is `lm`.
- So does that mean `coef.lm` exists?
- `coef.lm` doesn't exist, instead there is `coef.default`.
- `default` is a special class and when a generic doesn't have a method, it falls back to `generic.default`.
- `coef.default` is hidden from view though and you need to look deeper to find it: `stats:::coef.default`. LIVE DEMO

Case study ❶ `broom::tidy` Part 1/2

```
class(fit_stan)

## [1] "stanreg" "glm"     "lm"

broom::tidy

## function (x, ...)
## {
##     UseMethod("tidy")
## }
## <bytecode: 0x7ffe15c14d80>
## <environment: namespace:generics>
```

There is no `tidy.stanreg` method so uses the `broom::tidy.glm` instead.

Case study ❶ broom::tidy Part 2/2

```
library(broom)

tidy.stanreg <- function(x, ...) {
  est <- x$coefficients
  tibble(term = names(est),
         estimate = unname(est),
         std.error = x$ses)
}

tidy(fit_stan)
```

```
## # A tibble: 2 x 3
##   term        estimate std.error
##   <chr>          <dbl>     <dbl>
## 1 (Intercept)    30.5      1.42
## 2 wt             -3.23     0.359
```

Working with model objects

- Is this only for R though? How do you work with model objects in general?

Python

```python
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
x = np.array(r.mtcars['wt']).reshape(-1, 1)
y = np.array(r.mtcars['mpg'])
model = LinearRegression().fit(x, y)

[model.intercept_, model.coef_]

## [37.28512616734204, array([-5.34447157])]
```

```
## (Intercept)          wt
##   37.285126   -5.344472
```

Model objects are usually a `list` returning multiple output from the model fit.

When working with model objects, check the object structure and find the methods associated with it

(and of course check the documentation).

Some common tables

Descriptive summary statistics tables

	4 (N=11)	6 (N=7)	8 (N=14)	Overall (N=32)
mpg				
Mean (SD)	26.7 (4.51)	19.7 (1.45)	15.1 (2.56)	20.1 (6.03)
Median [Min, Max]	26.0 [21.4, 33.9]	19.7 [17.8, 21.4]	15.2 [10.4, 19.2]	19.2 [10.4, 33.9]
wt				
Mean (SD)	2.29 (0.570)	3.12 (0.356)	4.00 (0.759)	3.22 (0.978)
Median [Min, Max]	2.20 [1.51, 3.19]	3.22 [2.62, 3.46]	3.76 [3.17, 5.42]	3.33 [1.51, 5.42]
vs				
0	1 (9.1%)	3 (42.9%)	14 (100%)	18 (56.2%)
1	10 (90.9%)	4 (57.1%)	0 (0%)	14 (43.8%)

Regression results tables

	Dependent variable:
	mpg
wt	-5.344***
	(0.559)
Constant	37.285***
	(1.878)
Observations	32
R^2	0.753

What statistics to present?

This depends on what you want to convey and your audience.

There are two key purposes of the table:

1. *display* information; and
2. *communicate* information.

Descriptive summary statistics Part 1/2

- The main goal is to give a numerical summary to give a "feel" of what the data contains. These generally should convey:
  - variables in the data,
  - the number of observations for each variable,
  - missing values (if any),
  - the distribution, e.g. in the form of five number of summaries or counts/percentages for categorical variables.
- For numerical variables, you may have a **correlation table** which displays the correlation coefficients of every pair of variables. Because it is symmetrical, you can omit out the upper triangle.

Variables	Mean	SD	1.	2.
1. Miles per gallon	20.09	6.03		
2. Weight	3.22	0.98	-0.87	

Descriptive summary statistics Part 2/2

- You may have **cross-tabulations** (also called **contingency tables**)

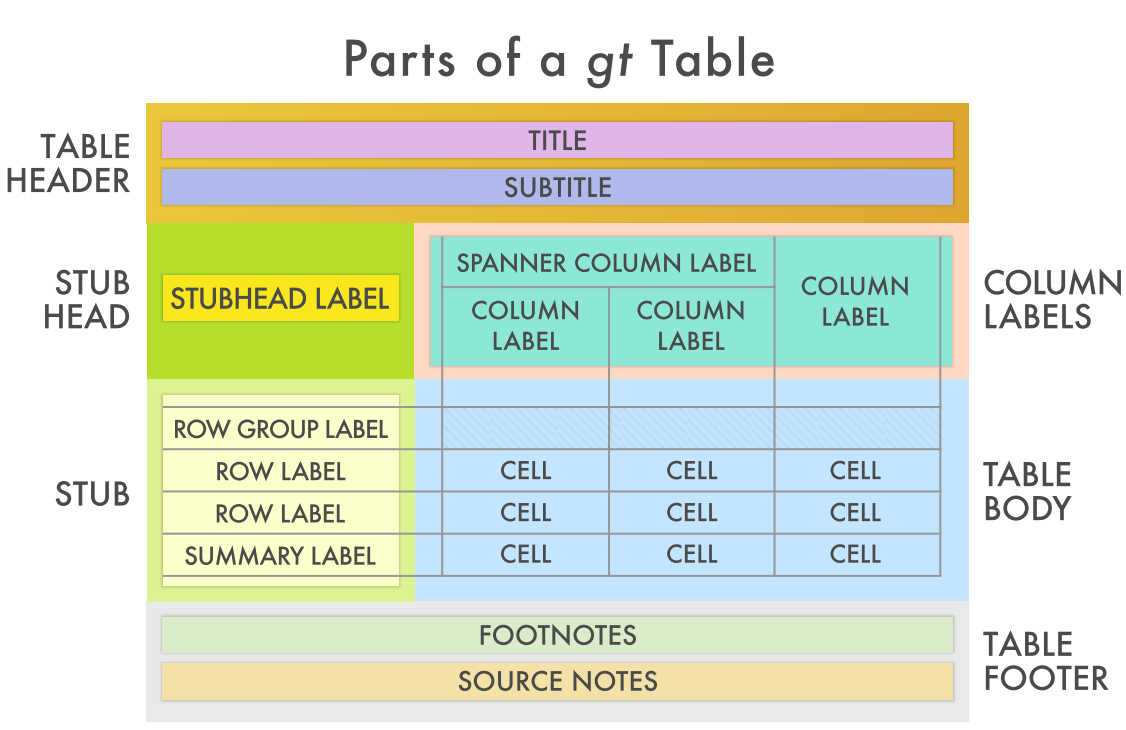
cyl/gear	3	4	5	Total
4	3.1% (1)	25.0% (8)	6.2% (2)	34.4% (11)
6	6.2% (2)	12.5% (4)	3.1% (1)	21.9% (7)
8	37.5% (12)	0.0% (0)	6.2% (2)	43.8% (14)
Total	46.9% (15)	37.5% (12)	15.6% (5)	100.0% (32)

▶ R code

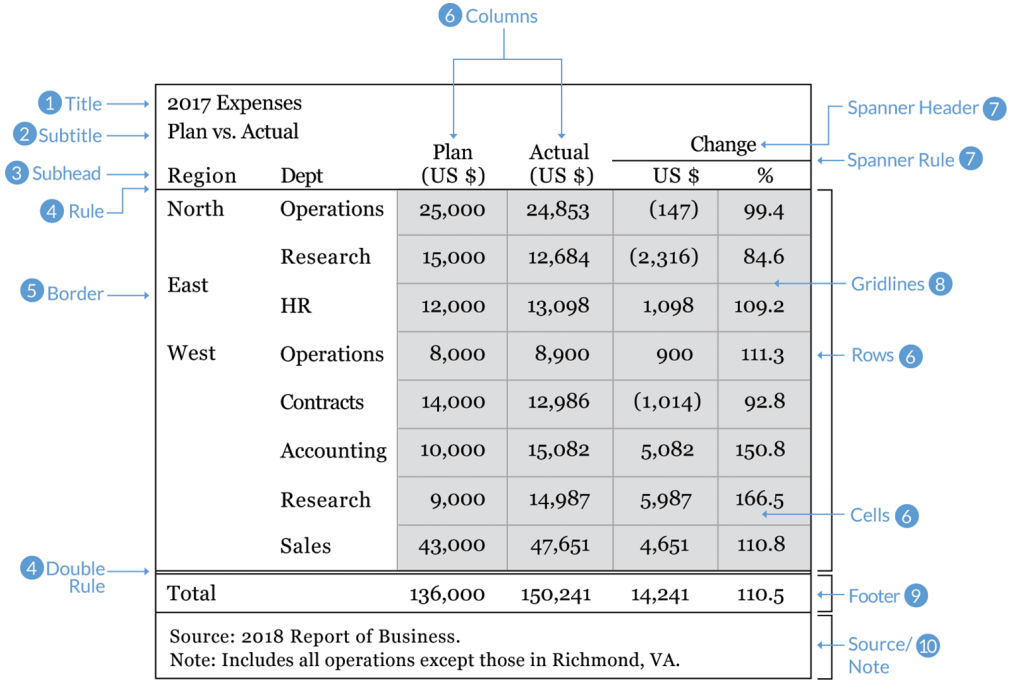
Regression results

- Or more generally, the important numerical characteristics of your model. This may include:
  - parameter estimates of your model,
  - the standard error or confidence/credible interval of your estimates,
  - model fit quality, e.g. R^2, AIC, BIC and so on.
- You see often the inclusion of p-value, usually from the significance test of the corresponding variable and some indicate the significance level by the amount of $*$.
- It is important to ***convey the uncertainty*** for the estimates or predictions from your model.
- There is also what is called the **ANOVA table** that shows the variation according to different sources.

Anatomy of the table



Source: https://gt.rstudio.com/



Source: Schwabish (2020) Ten Guidelines for Better Tables. *Journal of Benefit-Cost Analysis* **11** (2) 151-178

Numerical precision Part 1/2

- Select an appropriate precision for your goal and audience:

	Weight		
Car brand	**5 d.p.**	**2 d.p.**	**0 d.p.**
Mazda RX4	2.61937	2.62	3
Mazda RX4 Wag	2.87518	2.88	3
Datsun 710	2.31916	2.32	2
Hornet 4 Drive	3.21660	3.21	3

- Display trailing zeroes to match selected precision of the column:

Trailing zeroes	
Yes	**No**
0.233	0.233
0.320	0.32
0.400	0.4

Numerical precision Part 2/2

- Change and display units as appropriate:

Car brand	Weight (lbs)	Weight (1000 lbs)	Weight (kg)	Weight (g)	Weight (mg)
Mazda RX4	2620	2.620	1188	1190000	1,188,000,000
Mazda RX4 Wag	2875	2.875	1304	1300000	1,304,000,000
Datsun 710	2320	2.320	1052	1050000	1,052,000,000
Hornet 4 Drive	3215	3.215	1458	1460000	1,458,000,000

- Show comma every 3 digits (or other marks as needed). E.g. `1000000` is easier to read with `1,000,000`.

```
scales::comma(c(439024, 4900343), accuracy = 1000)

## [1] "439,000"   "4,900,000"
```

Column alignment

- Spanner labels are usually aligned in center.
- Right-align numbers
- Left-align texts

Car brand			Horsepower		
Left	**Center**	**Right**	**Left**	**Center**	**Right**
Mazda RX4	Mazda RX4	Mazda RX4	110	110	110
Mazda RX4 Wag	Mazda RX4 Wag	Mazda RX4 Wag	110	110	110
Datsun 710	Datsun 710	Datsun 710	93	93	93
Hornet 4 Drive	Hornet 4 Drive	Hornet 4 Drive	110	110	110

 How to make tables in R?

- There are many packages that make table in R, including ones that wrangle the data for you to make specialised table output. E.g. `knitr::kable`, `kableExtra`, `formattable`, `gt`, `DT`, `pander`, `xtable`, `stargazer`.
- You can read the documentation for each packages to make the table you want (I mainly use `knitr::kable`, `kableExtra` and `DT`).
- Whatever package you use, it's important that you understand the output and how it works with the medium you are trying to display the table.

Table in Markdown

In markdown file:

```
First Header  | Second Header
------------- | -------------
Content Cell  | Content Cell
Content Cell  | Content Cell
```

Possible display:

First Header	Second Header
Content Cell	Content Cell
Content Cell	Content Cell

Table in HTML & PDF

HTML → Web browser

```
<table>
<thead>
<tr>
<th>First Header</th> <th>Second Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Cell</td> <td>Content Cell</td>
</tr>
<tr>
<td>Content Cell</td> <td>Content Cell</td>
</tr>
</tbody>
</table>
```

LaTeX → PDF

```
\begin{tabular}{cc}
\hline
First Header & Second Header\\
\hline
Content Cell & Content Cell\\
Content Cell & Content Cell\\
\hline
\end{tabular}
```

- HTML is for HTML output
- LaTeX is for PDF output
- Most HTML elements do *not* work in LaTeX and vice versa.
- What is the R-package output?

LIVE DEMO

Case study ❷ interactive tables with DT

```
DT::datatable(mtcars, options = list(pageLength = 4))
```

Show 4 entries Search:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21	6	160	110	3.9	2.62	16.46	0	1	4	4
Mazda RX4 Wag	21	6	1								
Datsun 710	22.8	4	1								
Hornet 4 Drive	21.4	6	2								

Showing 1 to 4 of 32

- `DT::datatable` works best for HTML documents.
- For PDF documents, it takes a `webshot` of the HTML table and inserts it as an image.
- It's useful for data exploration where the main goal of the table is display of information rather than communication.

When do you make tables over plots?

When you want to ***show exact values*** or the ***accuracy of the values are important*** to convey.

You *can* combine plots with tables.

Case study ❸ heatmap table with `formattable`

```
library(formattable)
mtcars %>%
  select(mpg, disp, wt, hp) %>%
  cor() %>%
  as.data.frame() %>%
  rownames_to_column("Variables") %>%
  formattable(list(area(col = 2:5) ~ color_tile("#F5B7B1", "#7DCEA0")))
```

Variables	mpg	disp	wt	hp
mpg	1.0000000	-0.8475514	-0.8676594	-0.7761684
disp	-0.8475514	1.0000000	0.8879799	0.7909486
wt	-0.8676594	0.8879799	1.0000000	0.6587479
hp	-0.7761684	0.7909486	0.6587479	1.0000000

Case study 4 inline plots with `sparkline`

- **Sparkline** refers to a small chart drawn without axes or coordinates.
- There is a jQuery sparkline that is provided as a `htmlwidgets` in the `sparkline` R-package.
- This is a `htmlwidgets` object so it only works for HTML output and requires the JS dependencies to work.

LIVE DEMO

Today's main lessons

Working with model objects

- Understanding about the OOP system
- How to tease apart the model objects and methods associated with it

Presenting information as a table

- Various common tables that present information
- Some guidelines for best way to communicate with tables

That's it!

Lecturer: Emi Tanaka

Department of Econometrics and Business Statistics

ETC5523.Clayton-x@monash.edu