ACED Simple Regression

```r
library(DescTools)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.2     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.3     v tibble    3.2.1
v lubridate 1.9.2     v tidyr     1.3.0
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

ACED Data

```r
ACEDextract <- read_csv("ACED_extract1.csv",na="-999")
```

```
Rows: 290 Columns: 29
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (7): SubjID, Session, Cond_code, Sequencing, Feedback, Gender, Level_Code
dbl (22): Correct, Incorrect, Reamaining, ElapsedTime, Race, pre_scaled, pos...

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.
```

```
ACEDextract$Session <- factor(ACEDextract$Session)
ACEDextract$Cond_code <- factor(ACEDextract$Cond_code)
ACEDextract$Sequencing <- factor(ACEDextract$Sequencing)
ACEDextract$Feedback <- factor(ACEDextract$Feedback)
ACEDextract$Gender <- factor(ACEDextract$Gender)
ACEDextract$Race <- factor(ACEDextract$Race,1:8)
ACEDextract$Level_Code <- factor(ACEDextract$Level_Code)
```

```
ACEDextract %>%
  mutate(gain=post_scaled-pre_scaled) ->
  ACEDextract
```

Research Questions

In this case study we will address the first research question.

1. Do the pretest, posttest and internal game measures measure the same thing? (Validity and Reliability)

Making Scatterplots

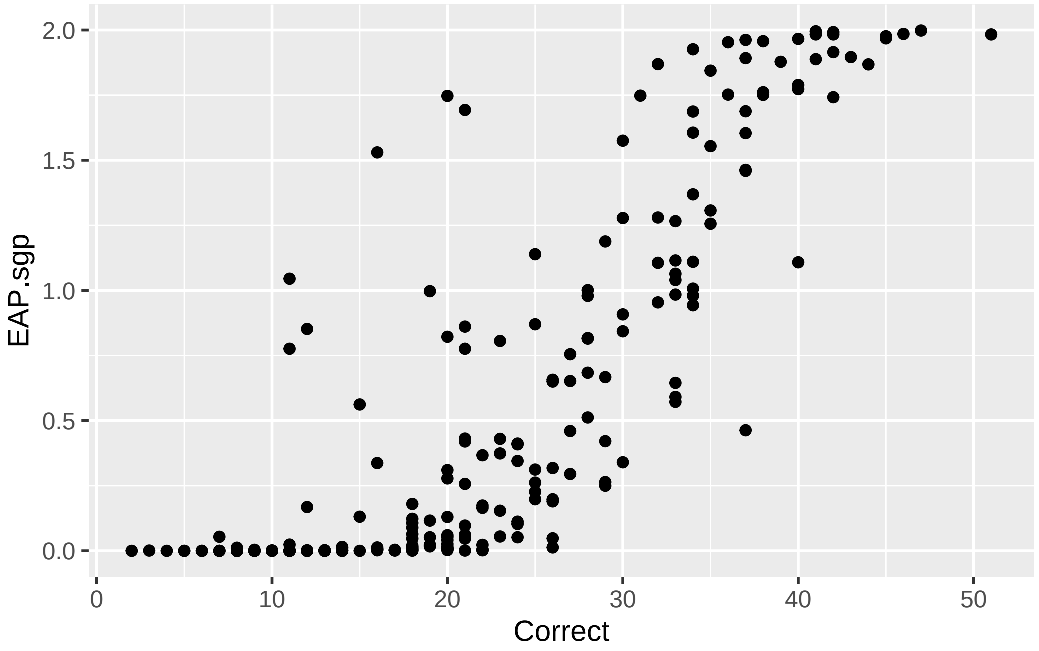
Use `geom_point()` with `ggplot()` to make a scatterplot.

Scatterplot

Here is a simple scatterplot.

```
EAPxCorrect <- ggplot(ACEDextract,aes(x=Correct,y=EAP.sgp)) +
  geom_point()
EAPxCorrect
```

```
Warning: Removed 60 rows containing missing values (`geom_point()`).
```



Adding lines and smooths

The function `geom_smooth()` adds a smooth line.

A few key arguments:

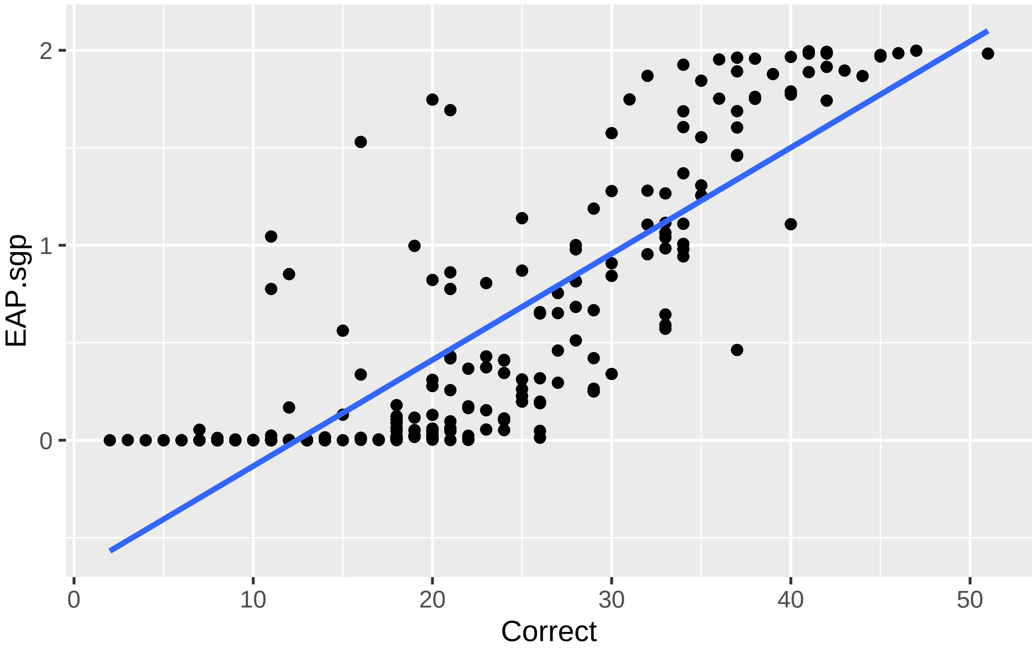
- `method` – “lm”, “loess”, “glm”, “gam”
- `formula` – This allows specifying other kinds of curves.
- `na.rm` – Logical, if `TRUE` then suppresses warning about NAs
- `se` – Logical, default `TRUE`, should standard errors be plotted.

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

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

```
Warning: Removed 60 rows containing non-finite values (`stat_smooth()`).
```

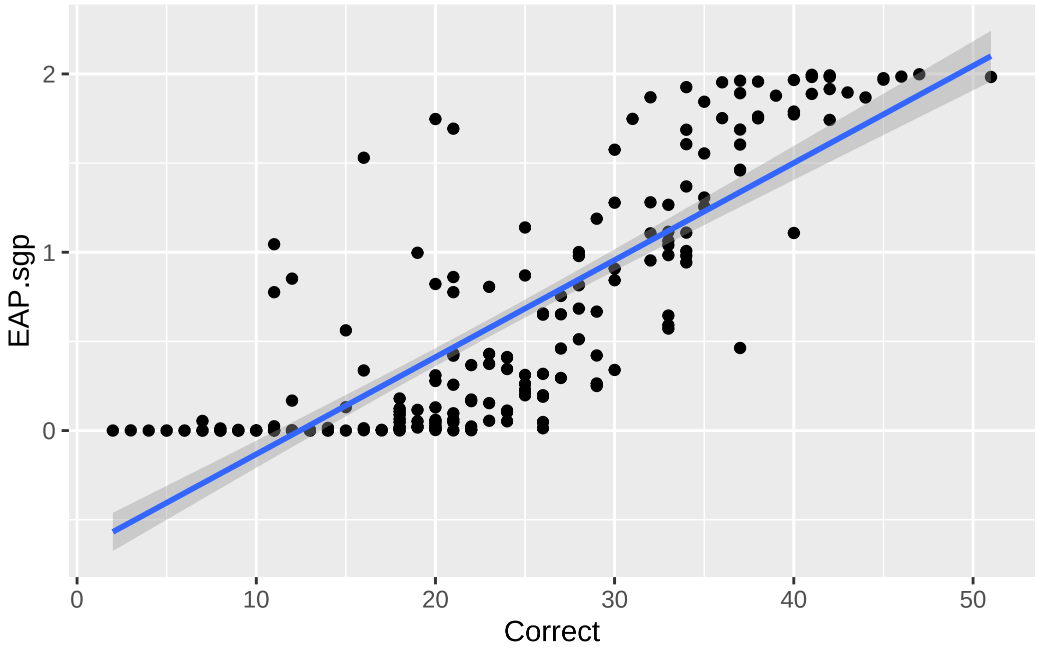
```
Warning: Removed 60 rows containing missing values (`geom_point()`).
```



```
EAPxCorrect + geom_smooth(method="lm")
```

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

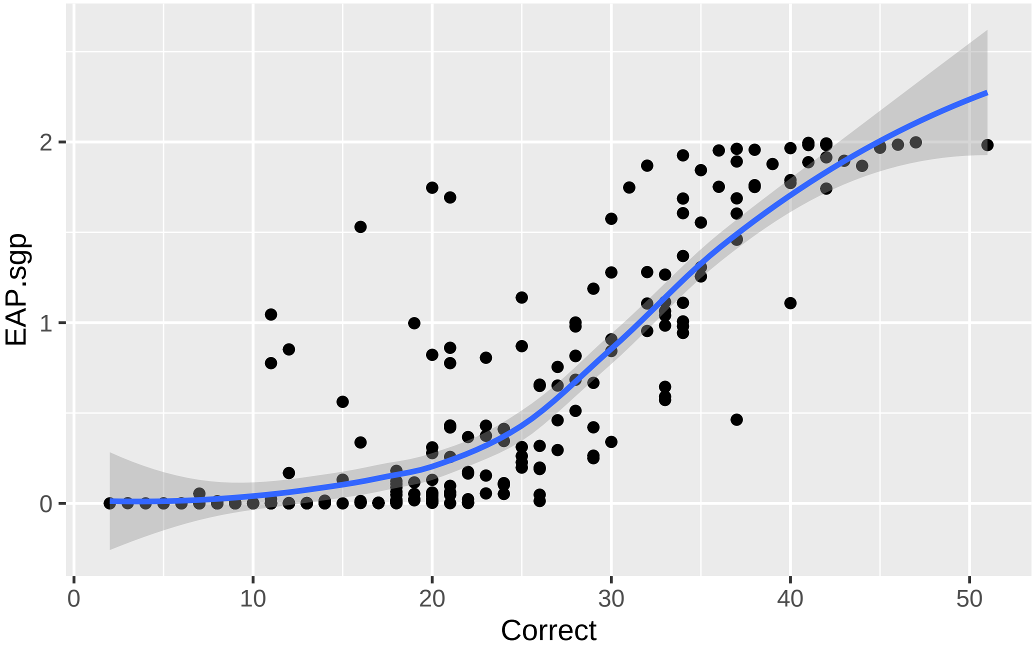
```
Warning: Removed 60 rows containing non-finite values (`stat_smooth()`).
Removed 60 rows containing missing values (`geom_point()`).
```



```
EAPxCorrect + geom_smooth(method="loess")
```

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

```
Warning: Removed 60 rows containing non-finite values (`stat_smooth()`).
Removed 60 rows containing missing values (`geom_point()`).
```

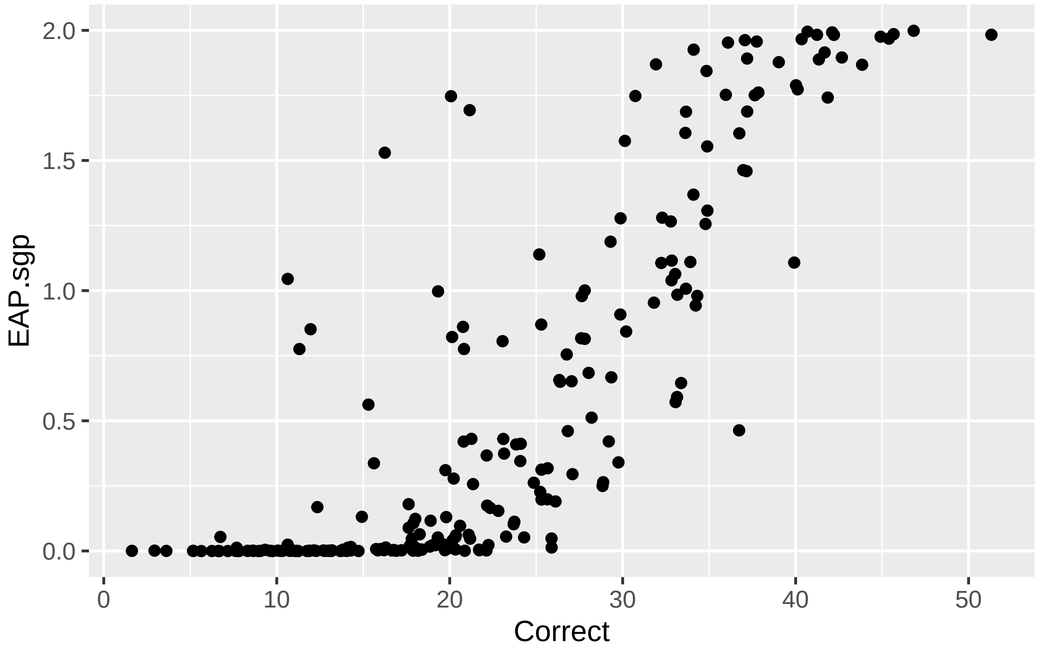


Jittering

When the data are integers (as in the count), sometimes points plot on top of each other. Jittering (adding a bit of random noise) can help.

```r
ggplot(ACEDextract,aes(x=Correct,EAP.sgp)) +
  geom_point(position="jitter")
```

```
Warning: Removed 60 rows containing missing values (`geom_point()`).
```



Coloring points

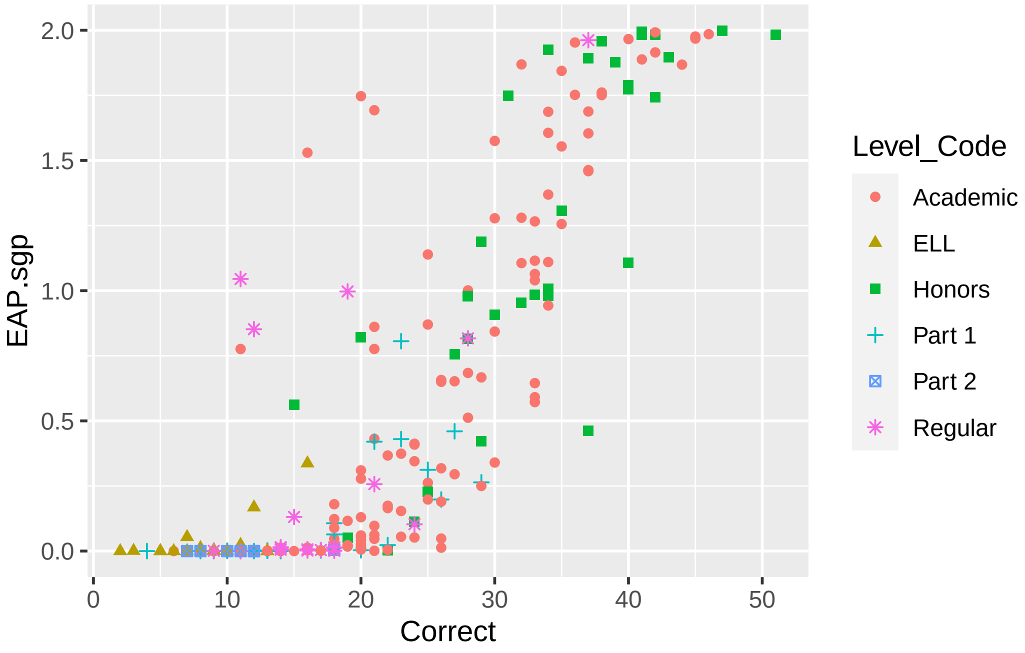
Attach a factor variable to

- `color` (line color) or `fill` (interior color)
- `shape` shape of plotting symbol
- `linetype` – type of the line (solid, dotted, dashed, &c).

Note: color can be a problem if (a) printing graph in black and white, or (b) show to somebody with limited color perception (about 8% of the population). Try to pair color with another aesthetic (e.g., shape or linetype).

```
ggplot(ACEDextract,aes(x=Correct,EAP.sgp,color=Level_Code,
                       shape=Level_Code)) +
  geom_point()
```

```
Warning: Removed 60 rows containing missing values (`geom_point()`).
```



Calclulating a correlation

Var and Cor

```
var(ACEDextract$Correct,ACEDextract$EAP.sgp)
```

```
[1] NA
```

```
cor(ACEDextract$Correct,ACEDextract$EAP.sgp)
```

```
[1] NA
```

```
select(ACEDextract,starts_with("EAP")) %>% cor()
```

```
        EAP.sgp EAP.cr EAP.dt EAP.eg EAP.exp EAP.ext EAP.mod EAP.rr EAP.tab
EAP.sgp       1     NA     NA     NA      NA      NA      NA     NA      NA
EAP.cr       NA      1     NA     NA      NA      NA      NA     NA      NA
EAP.dt       NA     NA      1     NA      NA      NA      NA     NA      NA
EAP.eg       NA     NA     NA      1      NA      NA      NA     NA      NA
EAP.exp      NA     NA     NA     NA       1      NA      NA     NA      NA
EAP.ext      NA     NA     NA     NA      NA       1      NA     NA      NA
EAP.mod      NA     NA     NA     NA      NA      NA       1     NA      NA
EAP.rr       NA     NA     NA     NA      NA      NA      NA      1      NA
EAP.tab      NA     NA     NA     NA      NA      NA      NA     NA       1
EAP.vr       NA     NA     NA     NA      NA      NA      NA     NA      NA
EAP.pic      NA     NA     NA     NA      NA      NA      NA     NA      NA
        EAP.vr EAP.pic
EAP.sgp     NA      NA
EAP.cr      NA      NA
EAP.dt      NA      NA
EAP.eg      NA      NA
EAP.exp     NA      NA
EAP.ext     NA      NA
EAP.mod     NA      NA
EAP.rr      NA      NA
EAP.tab     NA      NA
EAP.vr       1      NA
EAP.pic     NA       1
```

Handling missing data

Add `use="complete.obs"` or `use="pairwise.complete.obs"`

```
var(ACEDextract$Correct,ACEDextract$EAP.sgp,use="complete.obs")
```

```
[1] 6.059457
```

```
cor(ACEDextract$Correct,ACEDextract$EAP.sgp,use="complete.obs")
```

```
[1] 0.8359097
```

```
select(ACEDextract,starts_with("EAP")) %>% cor(use="pairwise.complete.obs")
```

```
          EAP.sgp    EAP.cr    EAP.dt    EAP.eg   EAP.exp   EAP.ext   EAP.mod
EAP.sgp 1.0000000 0.7924686 0.9967152 0.8617104 0.3638555 0.7353437 0.8675896
EAP.cr  0.7924686 1.0000000 0.8150896 0.6401405 0.2575187 0.7654101 0.6497430
EAP.dt  0.9967152 0.8150896 1.0000000 0.8461165 0.3501101 0.7598859 0.8617598
EAP.eg  0.8617104 0.6401405 0.8461165 1.0000000 0.2751036 0.5711900 0.7182038
EAP.exp 0.3638555 0.2575187 0.3501101 0.2751036 1.0000000 0.2124473 0.3428768
EAP.ext 0.7353437 0.7654101 0.7598859 0.5711900 0.2124473 1.0000000 0.6423054
EAP.mod 0.8675896 0.6497430 0.8617598 0.7182038 0.3428768 0.6423054 1.0000000
EAP.rr  0.5163624 0.4335190 0.5178691 0.4484167 0.3499196 0.4315655 0.4758091
EAP.tab 0.7000835 0.5831217 0.7020186 0.5854901 0.3349572 0.6244417 0.5934085
EAP.vr  0.7687073 0.5111927 0.7511615 0.6631146 0.4931869 0.5126654 0.7662889
EAP.pic 0.7055832 0.5067370 0.6867729 0.5971602 0.2377316 0.5008445 0.5979188
           EAP.rr   EAP.tab    EAP.vr   EAP.pic
EAP.sgp 0.5163624 0.7000835 0.7687073 0.7055832
EAP.cr  0.4335190 0.5831217 0.5111927 0.5067370
EAP.dt  0.5178691 0.7020186 0.7511615 0.6867729
EAP.eg  0.4484167 0.5854901 0.6631146 0.5971602
EAP.exp 0.3499196 0.3349572 0.4931869 0.2377316
EAP.ext 0.4315655 0.6244417 0.5126654 0.5008445
EAP.mod 0.4758091 0.5934085 0.7662889 0.5979188
EAP.rr  1.0000000 0.4800405 0.4960209 0.3914310
EAP.tab 0.4800405 1.0000000 0.4766964 0.4328901
EAP.vr  0.4960209 0.4766964 1.0000000 0.5630053
EAP.pic 0.3914310 0.4328901 0.5630053 1.0000000
```

Biserial and Polyserial correlations

polychor::polychor

Kendal's Tau-b

`DescTools::KendalTauB`

Fitting a Linear Model

The `lm()` function fits a linear model.

It returns an *object* of class "lm".

Can do interesting things with the object.

Formulas

The first argument to `lm()` is a formula.

A formula looks like `y ~ x`, where both x and y can be expressions with multiple variables.

`~` is a special character which makes a formula.

`y` is the dependent variable (what we want to predict)

`x` is the independent variable (what we are going to use to make the prediction)

Using the example above, `EAP.sgp ~ Correct`.

Generally, it will be the name of a variable, either in the data set or in the global environment.

Can also add a transformation, e.g., `log(x)` or `sqrt(x)`.

Sometimes use a `.` for special purposes.

Other arguments of `lm()`

- `data` – which data set are we using. Name of the data set, or "." if the data set is being piped in with "%>%.
- `subset` (optional) – either a vector of cases (row numbers) to use, or a logical vector same as number of rows in data which selects the cases to use.

Also can use `filter()` command on data before `lm()`

- `weights` – normally not used, but support complex survey designs.
- `na.action` – What to do with missing values.
  - "na.fail" – Generate an error
  - "na.omit" – Removes the missing values.
  - "na.exclude" – Removes the missing values, but pads the output so that the missing values can be predicted.
  - "na.pass" – passes the missing values through (result is likely to be `NA`, so usually not useful).

Can globally set the default by using `options()`

```
options("na.action")
```

```
$na.action
[1] "na.omit"
```

```
options(na.action=na.fail)
```

```
try(
lm_EAPxCorrect <- lm(EAP.sgp~Correct, data=ACEDextract)
)
```

```
Error in na.fail.default(structure(list(EAP.sgp = c(1.747, 0.000999999999999989,  :
  missing values in object
```

```
lm_EAPxCorrect <- lm(EAP.sgp~Correct, data=ACEDextract,
                     na.action="na.exclude")
```

Summaries

The result of running `lm` is an S3[1] object of class "lm".

```
class(lm_EAPxCorrect)
```

```
[1] "lm"
```

Generic functions do things slightly differently based on the, class of the [first] argument.

Methods of S3 generic functions are named `function.class`.

- `print.lm` – `print()` is an important generic function. The `print()` function is called when you just type the name of a variable in the console.

```
lm_EAPxCorrect
```

```


Call:
lm(formula = EAP.sgp ~ Correct, data = ACEDextract, na.action = "na.exclude")

Coefficients:
(Intercept)      Correct
   -0.67723      0.05446
```

[1] S3 objects, so called because they are described in the 3rd S book, Chambers and Hastie (1992) are lists with a special class attribute.

May want to change the `digits` argument.

```
print(lm_EAPxCorrect,digits=3)
```

```
Call:
lm(formula = EAP.sgp ~ Correct, data = ACEDextract, na.action = "na.exclude")

Coefficients:
(Intercept)      Correct
    -0.6772       0.0545
```

- `summary.lm` – The `lm` method of the summary function gives the statistics you commonly see in SPSS output.

```
summary(lm_EAPxCorrect)
```

```
Call:
lm(formula = EAP.sgp ~ Correct, data = ACEDextract, na.action = "na.exclude")

Residuals:
     Min       1Q   Median       3Q      Max
-0.87465 -0.28271 -0.03394  0.23909  1.33593

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.677235   0.058757  -11.53   <2e-16 ***
Correct      0.054456   0.002368   23.00   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.378 on 228 degrees of freedom
  (60 observations deleted due to missingness)
Multiple R-squared:  0.6987,    Adjusted R-squared:  0.6974
F-statistic: 528.8 on 1 and 228 DF,  p-value: < 2.2e-16
```

- `anova.lm` – This gives the ANOVA table

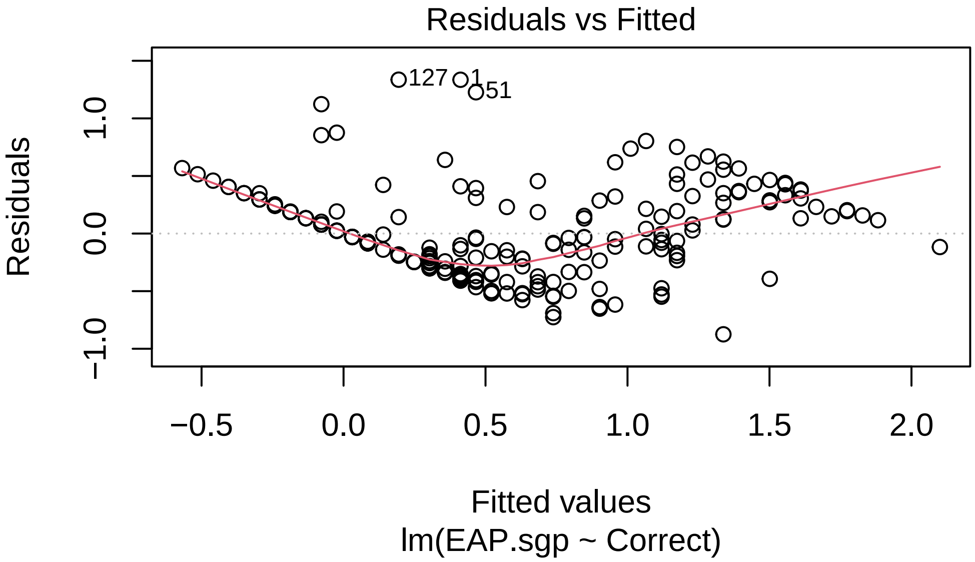
```
anova(lm_EAPxCorrect)
```

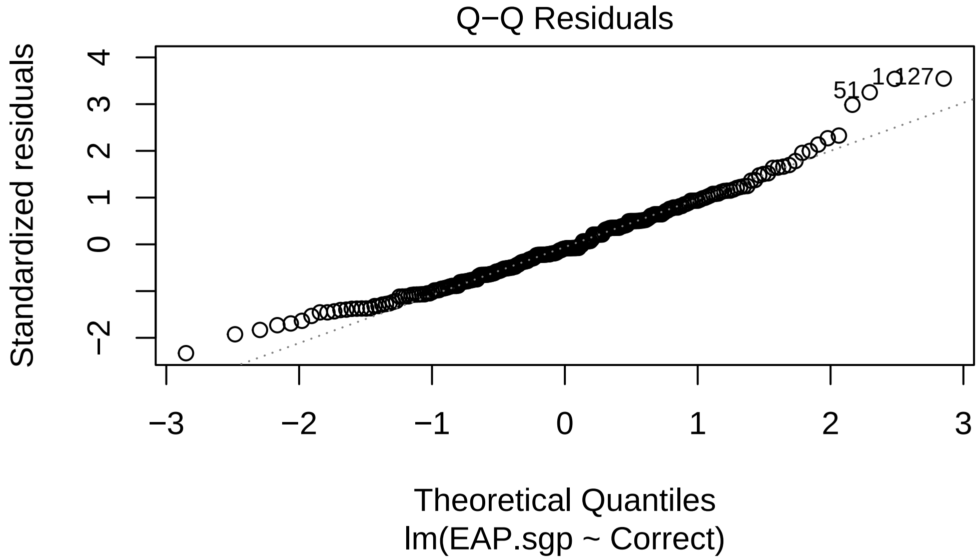
```
Analysis of Variance Table

Response: EAP.sgp
           Df Sum Sq Mean Sq F value    Pr(>F)    
Correct     1 75.564  75.564  528.83 < 2.2e-16 ***
Residuals 228 32.579   0.143                      
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

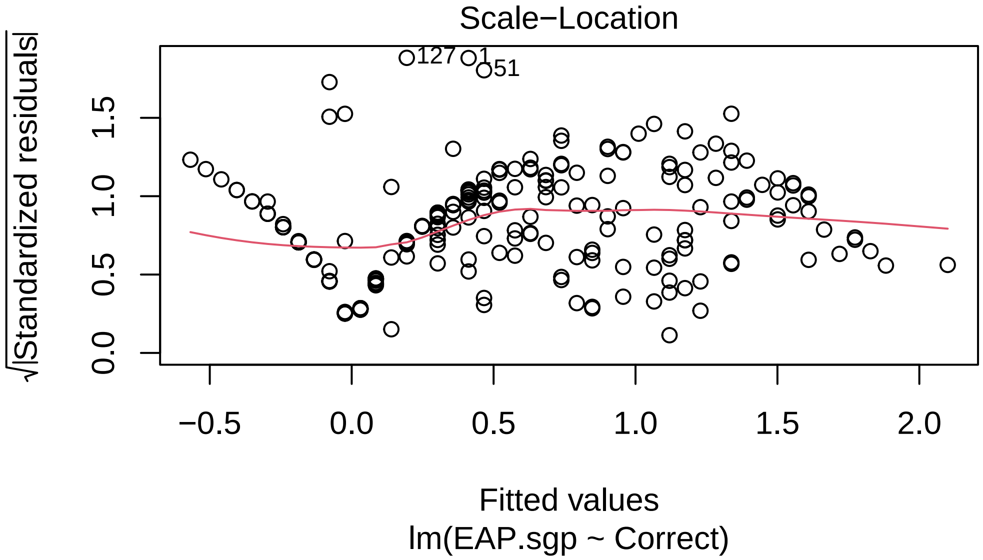
- `plot.lm` – This produces a number of diagnostic plots, more later.

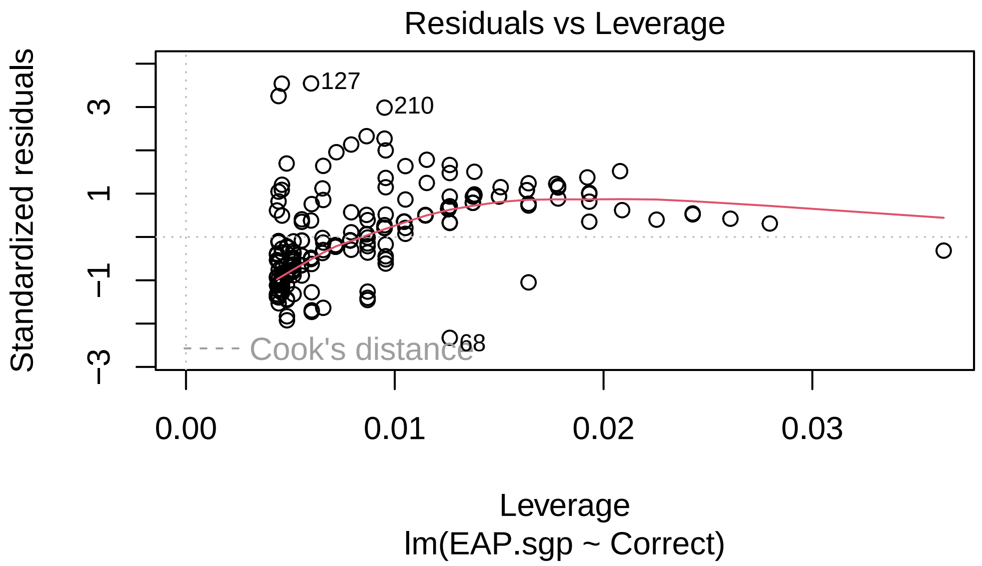
By default, the `plot.lm` method asks if you are ready before plotting the next plot. This is not necessary in RStudio, so add the option `ask=FALSE`.

```
plot(lm_EAPxCorrect,ask=FALSE)
```




Q−Q Residuals
Standardized residuals
−2
0
1
2
3
4
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(EAP.sgp ~ Correct)
51
1
127


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
−0.5
0.0
0.5
1.0
1.5
2.0
Fitted values
lm(EAP.sgp ~ Correct)
127
1
51



Note `help(plot)` gives help on the generic (any object) function, and `help(plot.lm)` gives help on the `lm` method for `plot`.

components

An S3 object is basically just a list. To access its components use the `$` operator

- `coefficients` – the slope and intercept
- `residuals` – the vector of residuals
- `fitted.values` – the vector of fitted values
- `df.residuals` – the degrees of freedom of the residuals.

```
lm_EAPxCorrect$coefficients
```

```
(Intercept)     Correct
-0.67723460  0.05445626
```

```
lm_EAPxCorrect$df.residual
```

```
[1] 228
```

```
head(lm_EAPxCorrect$residuals)
```

```
          1           2           3           4           5           6 
 1.33510938  0.02475946  0.51272172  0.24158451 -0.38589062  0.45482807 
```

```
head(lm_EAPxCorrect$fitted.values)
```

```
          1           2           3           4           5           6 
 0.41189062 -0.02375946  1.17427828 -0.24158451  0.41189062  0.68417193 
```

- `qr` – The Q and R matrixes from the QR decomposition.

Extracting bits

There are certain common extraction functions. (Usually better to use than the `$` operator.)

- `coef` – coefficients
- `effects` – effects, i.e., coefficients
- `vcov` – variance/covariance matrix
- `nobs` – number of [non-missing] observations.
- `variable.names` – names of variables used in model.

  ```
  coef(lm_EAPxCorrect)
  ```

  ```
  (Intercept)     Correct 
  -0.67723460  0.05445626 
  ```

  ```
  variable.names(lm_EAPxCorrect)
  ```

  ```
  [1] "(Intercept)" "Correct"
  ```

Extracting bits from the summary

- `summary()$sigma` – residual sd/standard error of the estimate
- `summary()$df` – degrees of freedom
- `summary()$fstatistic`
- `summary()$r.squared`, `summary()$adj.r.squared`

```
summary(lm_EAPxCorrect)$sigma
```

```
[1] 0.3780061
```

```
summary(lm_EAPxCorrect)$r.squared
```

```
[1] 0.6987451
```

Prediction

- `predict`
- `fitted`
- `residuals`, `rstandard`, `rstudent`
- `simulate`

Diagnostics

- `dfbeta`, `dfbetas`, `dffits`
- `cooks.distance`
- `influence`
- `hatvalues`

Model Fit

- `logLik`
- `deviance`

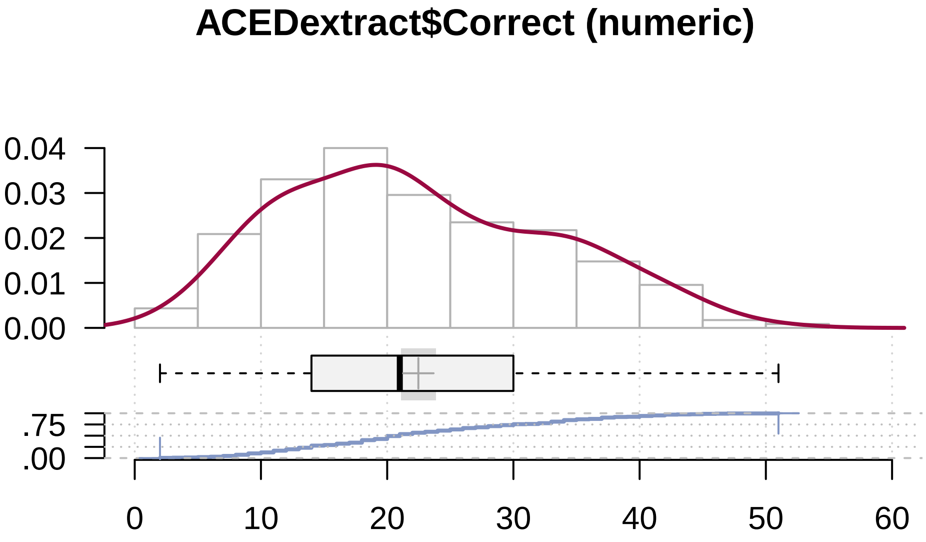
Tasks

1. Make marginal summaries for the following variables:

Correct,Incorrect,ElapsedTime,pre_scaled,post_scaled,EAP.sgp

```
Desc(ACEDextract$Correct)
```

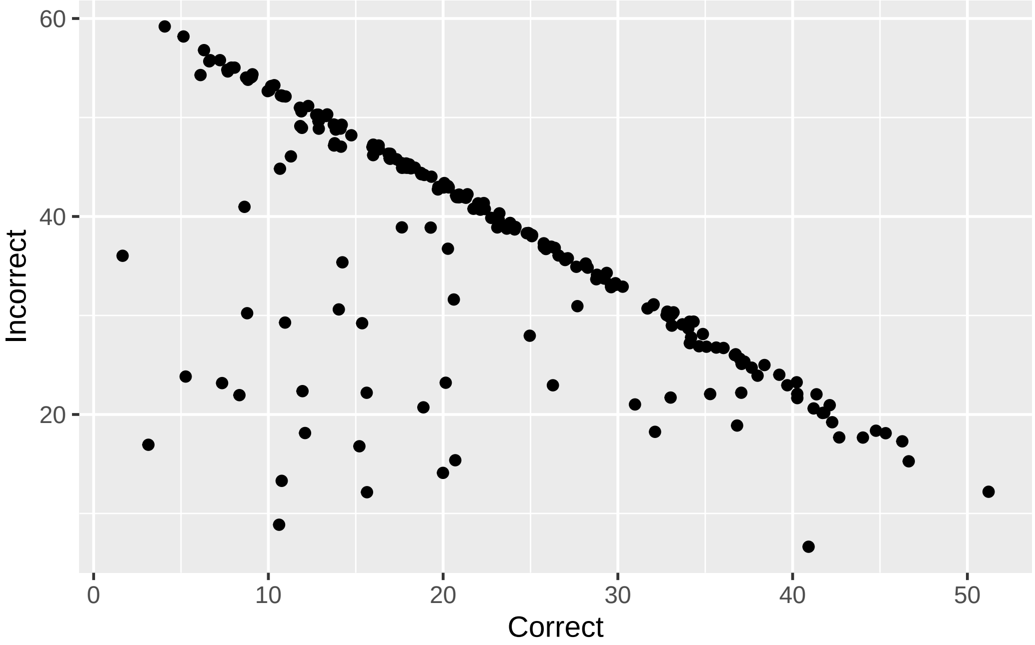
```
------------------------------------------------------------------------------
ACEDextract$Correct (numeric)

  length       n    NAs  unique      0s    mean  meanCI'
     290     230     60      47       0   22.47   21.10
           79.3%  20.7%            0.0%           23.84

     .05     .10    .25  median     .75     .90      .95
    8.00    9.00  14.00   21.00   30.00   37.00    41.00

   range      sd  vcoef     mad     IQR    skew     kurt
   49.00   10.55   0.47   11.86   16.00    0.38    -0.68

lowest : 2.0, 3.0, 4.0, 5.0 (2), 6.0 (2)
highest: 44.0, 45.0 (2), 46.0, 47.0, 51.0

' 95%-CI (classic)
```



2. Same as above, but break down by Level_code
3. Plot `Correct` against `Incorrect`. What is happening here?

```
ggplot(ACEDextract,aes(x=Correct,y=Incorrect)) +geom_point(position="jitter")
```

```
Warning: Removed 60 rows containing missing values (`geom_point()`).
```



4. Plot `EAP.sgp` against `post_scaled`. What is the correlation?
5. Plot `pre_scaled` against `post_scaled`. What is the correlation?
6. Regress `post_scaled` against `EAP.sgp`. Is `EAP.sgp` (the internal measure of ability from inside the game) a good predictor of `post_scaled` (the external predictor)?