The general linear model – Lecture 2
Adding a predictor

Jan Vanhove

https://janhove.github.io

Ghent, 15–17 July 2024

Having gained some understanding of how parameters and the uncertainty about those parameters can be estimated in the general linear model, we are now ready to consider a more interesting flavour of this model. What DeKeyser et al. (2010) wanted to find out wasn't so much the mean GJT value as the relationship between AOA and GJT. It's always a good idea to **draw a graph** first. For the type of question we're concerned with, a **scatterplot** is a reasonable choice.

Tip 2.1 (Plot, plot, plot)**.** Take out time to learn to draw plots, both to learn more about your data and to communicate your findings in talks and papers. This script will feature some basic examples, but for a better introduction, I recommend Kieran Healy's *Data visualization: A practical introduction*, which is available for free at https://socviz.co/. You'll find some further resources on my website (https://janhove.github.io). ♢

Let's first read in the data again:

```
library(tidyverse)
library(here)
d <- read_csv(here("data", "dekeyser2010.csv"))
```

Since it's impossible for GJT to influence AOA but quite likely that AOA influences GJT, we put the AOA values along the x-axis and the GJT values along the y-axis (Figure 1).

```
ggplot(data = d,
       aes(x = AOA,
           y = GJT)) +
  # shape = 1 draws empty circles
  geom_point(shape = 1) +
  xlab("Age of acquisition (years)") +
```

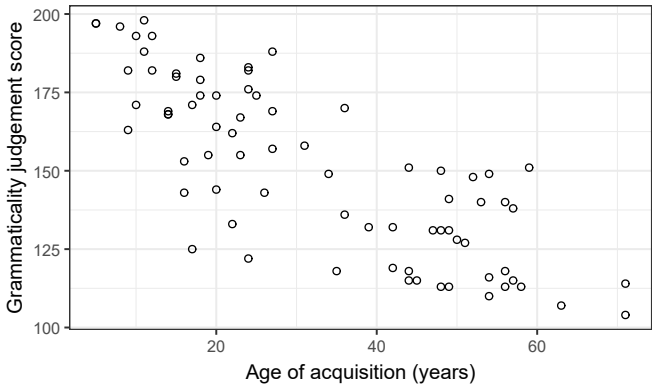

Figure 1: Scatterplot of the AOA-GJT relationship.

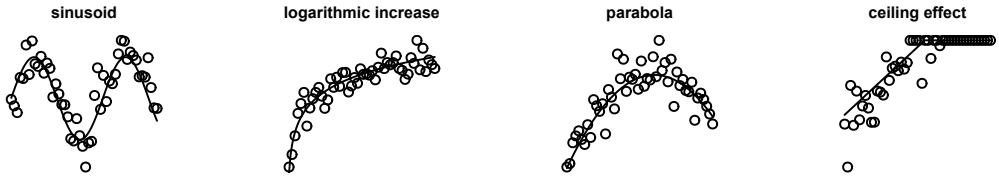

Figure 2: Examples of non-linear relationships.

```
  ylab("Grammaticality judgement score")
```

Note that there's a general tendency for the GJT to be lower for ever higher AOA values. Moreover, it seems as though this decrease is roughly linear. By way of contrast, Figure 2 shows four examples of non-linear relationships. Moreover, the scatterplot doesn't reveal any implausible data points that could be due to errors during data entry and the like: There are no 207-year-olds or GJT scores beyond the permissible range of [0, 204].

In what follows, we will apply the techniques learnt in the previous lecture in order to answer the following question: How does the GJT variable relate to the AOA variable? That is, once we know the value of someone's AOA what should be our best guess for that person's GJT score?

1 The simple linear regression equation

We proceed as in Lecture 1: we partition the outcome values into a part common to all outcome values and some discrepancy from the general trend. This time, however, we include some

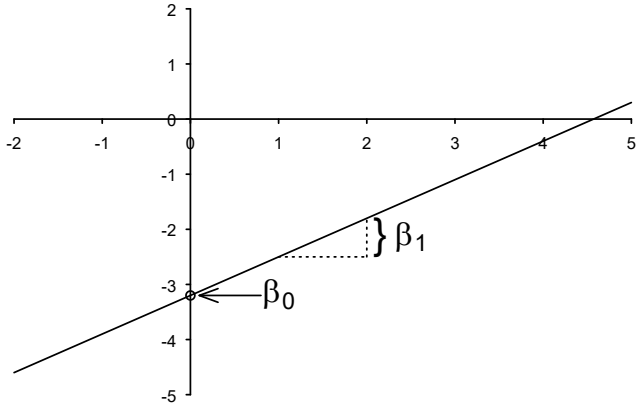

Figure 3: Intercept (β_0) and slope (β_1) of a straight line.

information about the link between AOA and GJT in the 'communality' term:

a specific outcome = communality between all outcomes (incl. link with AOA) + discrepancy.

Specifically, we will model this communality in terms of a straight line that is a function of AOA. Straight lines are parametrised by an intercept (we'll write β_0) and a slope (β_1):

$$f(x) = \beta_0 + \beta_1 x.$$

The intercept tells us the $f(x)$ value for $x = 0$; the slope tells us by how much $f(x)$ changes when x is increased by one unit; see Figure 3. From this we obtain the following **simple linear regression** equation:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i. \tag{1}$$

In our running example, y_i still refers to the i-th outcome, i.e., the i-th GJT score, and ε_i still refers to the i-th residual. What's new is the predictor, x (with values $x_1, x_2, \ldots, x_n$); in our example, these are the AOA values. The parameters β_0 and β_1 represent the intercept and the slope of the straight line that models the relationship between AOA and GJT.

Warning 2.2 (The Greek letter fallacy)**.** Both the model from Lecture 1 and the one in Equation 1 contain the term β_0. But these two β_0s refer to different things (mean vs. intercept). The meaning of a model parameter hinges crucially on the other parameters as well as the variables included in the model, so don't equate parameters in different models just because they have the same name. Lectures 4 and 5 in particular will drive this point home. ♢

Incidentally, it's called 'simple linear regression' because we're modelling the outcome in terms of a single predictor (hence simple, as opposed to multiple), and the outcome is modelled as

a weighted sum of the predictor values (i.e., as a 'linear combination'). The 'regression' bit stems from its origins in studying the phenomenon of 'regression to the mean', but that'd take us too far.

2 Estimating the parameters

If we just pick our estimates for the β_0, β_1 parameters by hand, we could come up with any number of estimates that work sort of okay, just like in Lecture 1. We need a more principled approach.

The estimation approaches we discussed in Lecture 1 still work. We will content ourselves with the least squares method as this is the one you'll encounter in practice: we choose $\widehat{\beta}_0, \widehat{\beta}_1$ so as to minimise the sum of the estimated residuals. (Equivalently, we could choose them so as to maximise the likelihood of the data assuming normally distributed residuals.) In principle, we could work through a number of suggestions for $\widehat{\beta}_0$ and $\widehat{\beta}_1$ and then choose the pair of estimates that minimises $\sum_{i=1}^{n} \widehat{\varepsilon}_i^2$. Figure 4 shows that estimates of $\widehat{\beta}_0 \approx 190$ and $\widehat{\beta}_1 \approx -1.2$ are optimal in this sense.

We could also derive these estimates mathematically. Conceptually, the maths are fairly easy: we want to solve the same problem as in the previous lecture, but for two parameter estimates simultaneously rather than for just one. But to spell this out precisely, we would need some first-year university mathematics. So let's skip straight to computing the OLS estimates in R:

```
aoa.lm <- lm(GJT ~ AOA, data = d)
coef(aoa.lm)
```

```
(Intercept)         AOA
    190.409      -1.218
```

Having estimated these parameters, we can add the estimated regression line to the scatterplot (Figure 5).

```
ggplot(data = d,
       aes(x = AOA,
           y = GJT)) +
  geom_point(shape = 1) +
  geom_abline(intercept = coef(aoa.lm)[[1]],
              slope = coef(aoa.lm)[[2]])
```

We could also directly use the `geom_smooth()` function:

```
# not shown
ggplot(data = d,
```

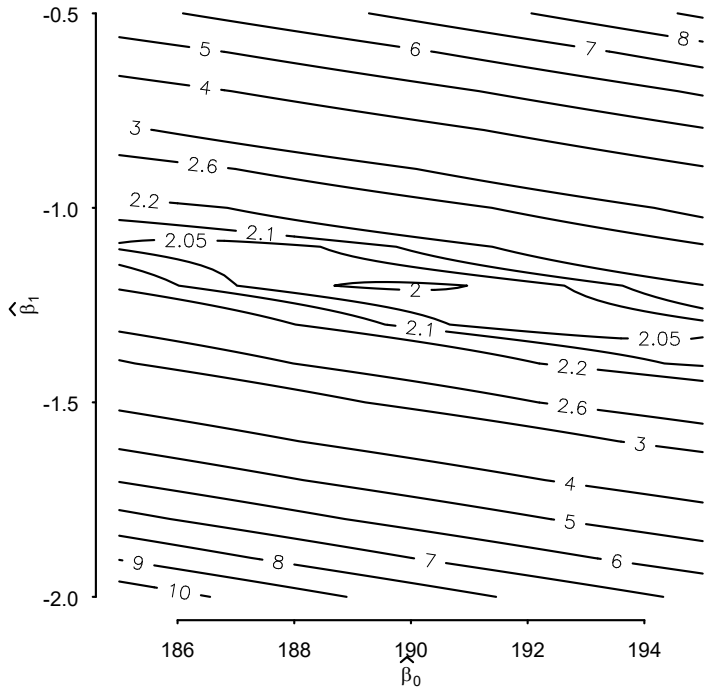

Figure 4: The sum of the squared residuals (divided by 10,000) of the GJT data for different combinations of parameter estimates. You can read this plot like a topographic map: the lines are contour lines. The sum is minimised for an intercept estimate near 190 and a slope estimate near -1.2.

```
      aes(x = AOA,
          y = GJT)) +
  geom_point(shape = 1) +
  geom_smooth(method = "lm", se = FALSE)
```

I set the `se` parameter in the `geom_smooth()` layer to `FALSE`. Setting it to `TRUE` would plot a 95% confidence band – which is a concept we haven't yet discussed.

Remark 2.3 (Other optimality criteria)**.** Apart from the least squares method, other methods exist for estimating the parameters in the general linear model. In Lecture 1, the method of least absolute deviations was already mentioned, which leads to **median regression** (a form of **quantile regression**). Some methods seek to minimise some combination of the sum of the squared deviations and the size of the β estimates. Depending on the precise criterium, these techniques are known as **lasso regularisation**, **ridge regularisation** or the **elastic net**. The basic idea behind these techniques is to purposefully introduce some bias in the estimation of

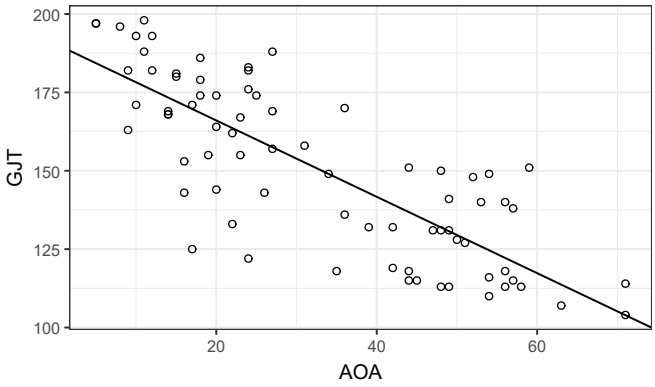

Figure 5: Scatterplot with the estimated regression line.

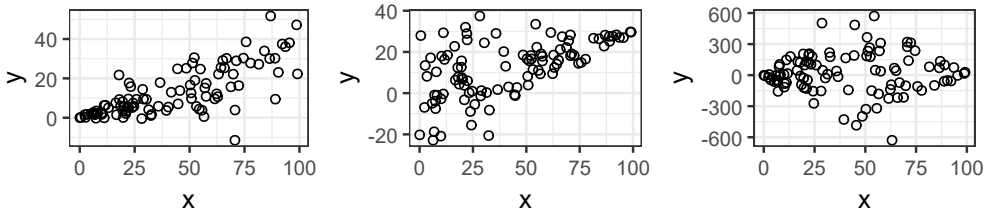

Figure 6: In all scatterplots, the spread of the data varies considerably with the x values.

the β parameters but by doing so reduce the sample-to-sample variability of the estimates. $\diamondsuit$

3 Quantifying uncertainty

Our estimates for β_0, β_1 are based on a sample and are hence inherently uncertain. Again, we can estimate the degree of this uncertainty if we are willing to make some assumptions about the residuals. As in Lecture 1, we assume that the residuals are i.i.d. We already covered the 'independence' part of this assumption. The 'identical' bit (i.e., the homoskedasticity assumption) can be illustrated by taking a look at what some crass violations of it look like: Figure 6 shows three examples where the distribution of the residuals, more specifically the variance of this distribution, changes with x. If we're willing to make the i.i.d. assumption, we can again obtain uncertainty measures by means of bootstrapping or by assuming normality.

3.1 The semi-parametric bootstrap

The logic is identical to that of Lecture 1. The only difference is that we now estimate two parameters. Instead of generating a vector with bootstrapped estimates, we need to write pairs of estimates in the rows of a two-column matrix. As a result, the code changes slightly.

But the logic is the same.

```
d$Prediction <- predict(aoa.lm)
d$Residual <- resid(aoa.lm)

n_bootstrap <- 20000
# preallocate 20000-by-2 matrix
bs_b <- matrix(nrow = n_bootstrap, ncol = 2)

for (i in 1:n_bootstrap) {
  # Step 2
  bs_resid <- sample(d$Residual, replace = TRUE)
  # Steps 3
  d <- d |>
    mutate(bs_outcome = Prediction + bs_resid)
  # Step 4
  bs_mod <- lm(bs_outcome ~ AOA, data = d)
  # store both estimates in i-th row
  bs_b[i, ] <- coef(bs_mod)
}
```

Let's inspect the first couple of rows of this matrix; if you run this code yourself, you'll obtain different results due to the randomness in step 2:

```
head(bs_b)
```

```
       [,1]    [,2]
[1,] 194.94 -1.3246
[2,] 189.45 -1.2201
[3,] 190.08 -1.1643
[4,] 197.57 -1.3748
[5,] 188.43 -1.1633
[6,] 192.70 -1.2742
```

The first column contains the bootstrapped $\widehat{\beta}_0^*$ values; the second column the bootstrapped $\widehat{\beta}_1^*$ values. If you draw a histogram of these bootstrap estimates, you'll notice that both distributions look pretty normal:

```
# not shown
hist(bs_b[, 1])
hist(bs_b[, 2])
```

The standard deviations of these distributions serve as estimates of the standard errors for

$\widehat{\beta}_0, \widehat{\beta}_1$:

```
# apply(x, 2, function) applies function to x by columns
apply(bs_b, 2, sd)
```

```
[1] 3.84815 0.10384
```

```
# alternatively:
sd(bs_b[, 1])
sd(bs_b[, 2])
```

That is, we estimate β_0 to be 190.4 ± 3.8 and β_1 to be -1.2 ± 0.1.

Confidence intervals can be obtained like before using the percentile method or by referring to a standard normal distribution. So for 95% confidence intervals:

```
# percentile method
apply(bs_b, 2, quantile, probs = c(0.025, 0.975))
```

```
        [,1]    [,2]
2.5%  182.83 -1.4242
97.5% 197.87 -1.0151
```

```
# standard normal
mean(bs_b[, 1]) + qnorm(c(0.025, 0.975)) * sd(bs_b[, 1])
```

```
[1] 182.85 197.93
```

```
mean(bs_b[, 2]) + qnorm(c(0.025, 0.975)) * sd(bs_b[, 2])
```

```
[1] -1.4209 -1.0138
```

Because the distributions of the bootstrapped estimates are both pretty much normal, both methods yield essentially the same results.

Note, incidentally, how we made use of the i.i.d. assumption in this bootstrap. In step 2, we resampled the estimated residuals with replacement, but without any further constraints. That is, all draws were independent of one another and all were sampled with the same probability from the empirical residual distribution.[1]

[1] Alternatives that do not assume homoskedasticity are easy to imagine. We could, for instance, have specified that we only resample the residuals for participants with AOA values over 40 from participants with AOA values over 40, and similarly for participants with lower AOA values. This would correspond to the assumption that the residuals for participants with high vs. low AOA values are sampled from separate distributions. The code to run this bootstrap variation would only be slightly more complicated.

3.2 The parametric bootstrap

Alternatively, we could assume that the residuals are drawn from a normal distribution, i.e.,

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_i + \varepsilon_i, \\ \varepsilon_i &\overset{\text{i.i.d.}}{\sim} \text{Normal}(0, \sigma_\varepsilon^2), \end{aligned} \tag{2}$$

for $i = 1, 2, \ldots, n$. Note that this notation encapsulates the homoskedasticity assumption: the variance of the normal distribution (σ_ε^2) is the same for all $i = 1, 2, \ldots, n$.

The R code to run the bootstrap doesn't contain anything new:

```
sigma_aoa.lm <- sigma(aoa.lm)

n_bootstrap <- 20000
bs_b <- matrix(nrow = n_bootstrap, ncol = 2)

for (i in 1:n_bootstrap) {
  # Step 2
  bs_resid <- rnorm(n = nrow(d), sd = sigma_aoa.lm)
  # Step 3
  d <- d |>
    mutate(bs_outcome = Prediction + bs_resid)
  # Step 4
  bs_mod <- lm(bs_outcome ~ AOA, data = d)
  bs_b[i, ] <- coef(bs_mod)
}
```

Histograms show that the bootstrapped β estimates are normally distributed:

```
# not shown
hist(bs_b[, 1])
hist(bs_b[, 2])
```

The confidence intervals and estimated standard errors we obtain are essentially the same as before.

```
apply(bs_b, 2, quantile, probs = c(0.025, 0.975))
```

```
        [,1]    [,2]
2.5%  182.62 -1.4201
97.5% 197.88 -1.0096
```

```
apply(bs_b, 2, sd)
```

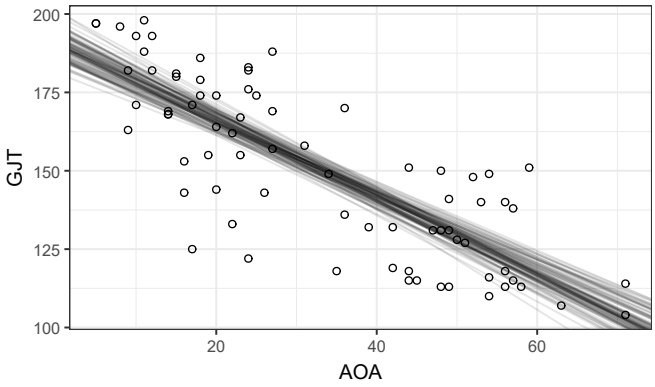

Figure 7: Scatterplot with 100 bootstrapped regression lines.

```
[1] 3.91546 0.10514
```

Now's a good time to visualise the uncertainty about the location of the regression line. We have 20,000 pairs of bootstrapped estimates of β_0, β_1. To gauge the uncertainty about the location of the regression line, we can draw a handful of the straight lines implied by these estimates. In the code below, the first 100 pairs of estimates are used; Figure 7 shows the result.

```
plot_bs <- ggplot(data = d,
                  aes(x = AOA,
                      y = GJT)) +
  geom_point(shape = 1)

for (i in 1:100) {
  plot_bs <- plot_bs +
    geom_abline(intercept = bs_b[i, 1],
                slope = bs_b[i, 2],
                # use alpha to make the lines a bit transparent
                alpha = 1/10)
}

plot_bs
```

In practice, people don't draw these bootstrapped regression lines. Instead, they colour in the region in which most of these lines fall. This region is known as a (pointwise) **confidence band**. Let's see how we can draw generate such confidence bands ourselves as this'll give us some further insights into the general linear model. First, we generate a sequence of predictor

values for which we want to plot the confidence band. Here, we just take all integers between the minimum and maximum AOA values in our sample:

```
# Step 1
new_aoa <- seq(from = min(d$AOA), to = max(d$AOA), by = 1)
# that is, 5, 6, 7, ..., 69, 70, 71
```

Note that there are 67 values in `new_aoa`. We have already generated 20,000 pairs of bootstrapped parameter estimates. For each of these pairs, we can compute the location of the bootstrapped regression line at each of the newly created predictor values. For instance, we can evaluate the regression line for the 37th bootstrap run like so:

```
bs_b[37, 1] + bs_b[37, 2] * new_aoa
```

```
 [1] 186.88 185.71 184.54 183.37 182.20 181.03 179.86 178.68
 [9] 177.51 176.34 175.17 174.00 172.83 171.65 170.48 169.31
[17] 168.14 166.97 165.80 164.63 163.45 162.28 161.11 159.94
[25] 158.77 157.60 156.42 155.25 154.08 152.91 151.74 150.57
[33] 149.40 148.22 147.05 145.88 144.71 143.54 142.37 141.20
[41] 140.02 138.85 137.68 136.51 135.34 134.17 132.99 131.82
[49] 130.65 129.48 128.31 127.14 125.97 124.79 123.62 122.45
[57] 121.28 120.11 118.94 117.76 116.59 115.42 114.25 113.08
[65] 111.91 110.74 109.56
```

Note that this yields 67 values – one for each `new_aoa` value. We now carry out this computation not only for the 37th bootstrap run but for all 20,000 runs. We store the results into a 20,000-by-67 matrix:

```
# Step 2
bs_y_hat <- matrix(nrow = n_bootstrap,
                   ncol = length(new_aoa))
for (i in 1:n_bootstrap) {
  bs_y_hat[i, ] <- bs_b[i, 1] + bs_b[i, 2]*new_aoa
}
```

Each row of this matrix contains the location of a different bootstrapped regression line corresponding to all 67 different values of `new_aoa`. We now look up the 2.5th and 97.5th percentile of the generated values at each `new_aoa` value; at each point, 95% of the bootstrapped regression lines lie between these percentiles:

```
# Step 3
lo_95 <- apply(bs_y_hat, 2, quantile, probs = 0.025)
hi_95 <- apply(bs_y_hat, 2, quantile, probs = 0.975)
```

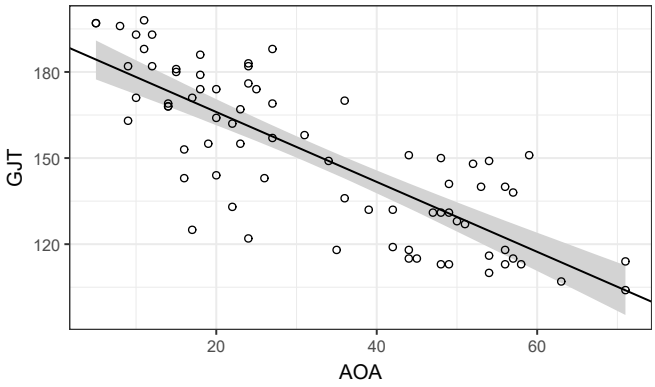

Figure 8: Scatterplot with a bootstrap-based 95% confidence band for the regression line.

Finally, we combine the predictor values and these percentile values into a tibble and plot them.[2] See Figure 8.

```
# Step 4
confidence_band_tbl <- tibble(new_aoa, lo_95, hi_95)

ggplot(data = confidence_band_tbl,
       aes(x = new_aoa)) +
  geom_ribbon(aes(ymin = lo_95,
                  ymax = hi_95),
              fill = "lightgrey") +
  geom_point(data = d,
             aes(x = AOA, y = GJT),
             shape = 1) +
  geom_abline(intercept = coef(aoa.lm)[[1]],
              slope = coef(aoa.lm)[[2]]) +
  xlab("AOA") +
  ylab("GJT")
```

3.3 t-distributions

As long as we're assuming that the residuals are i.i.d. draws from a normal distribution, we might as well estimate the standard errors and compute the confidence intervals for the parameter estimates directly.

[2] Tibbles are the counterpart of standard data frames when using the tidyverse package. But you can use `data.frame()` instead of `tibble()` if you prefer.

```
summary(aoa.lm)


Call:
lm(formula = GJT ~ AOA, data = d)

Residuals:
   Min     1Q Median     3Q    Max
-44.70  -9.54  -0.26  13.02  32.45

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  190.409      3.904    48.8   <2e-16
AOA           -1.218      0.105   -11.6   <2e-16

Residual standard error: 16.4 on 74 degrees of freedom
Multiple R-squared:  0.645, Adjusted R-squared:  0.64
F-statistic:  134 on 1 and 74 DF,  p-value: <2e-16
```

```
confint(aoa.lm)
```

```
               2.5 %   97.5 %
(Intercept) 182.6297 198.1876
AOA          -1.4275  -1.0085
```

We can also directly obtain a confidence interval for the location of the regression line some prespecified predictor values using the `predict()` function. The following command generates a 95% confidence interval for the location of the regression line at each of the 67 values in `new_aoa`.

```
conf_band_t <- predict(aoa.lm, newdata = tibble(AOA = new_aoa),
                       interval = "confidence")
head(conf_band_t) # 1st row: AOA = 5, 2nd: AOA = 6, etc.
```

```
     fit    lwr    upr
1 184.32 177.44 191.20
2 183.10 176.40 189.81
3 181.88 175.35 188.41
4 180.66 174.30 187.03
5 179.45 173.25 185.64
6 178.23 172.20 184.26
```

We can plot the result of these computations, the result of which is hardly discernible from the one obtained previously:

```
# not shown
conf_band_t <- as.tibble(conf_band_t)
conf_band_t$AOA <- new_aoa
ggplot(data = conf_band_t,
       aes(x = AOA)) +
  geom_ribbon(aes(ymin = lwr,
                  ymax = upr),
              fill = "lightgrey") +
  geom_point(data = d,
             aes(x = AOA, y = GJT),
             shape = 1) +
  geom_line(aes(y = fit)) +
  xlab("AOA") +
  ylab("GJT")
```

Once we've understood what we're really doing when drawing a confidence band, we can do so without all the rigmarole:

```
# not shown
ggplot(data = d,
       aes(x = AOA, y = GJT)) +
  geom_point(shape = 1) +
  geom_smooth(method = "lm")
```

4 Interpreting regression lines

Equation 2 on page 9 is useful for getting a conceptual grasp of the regression line. According to this equation, we assume that the residuals are drawn i.i.d. from a normal distribution with mean 0 and variance σ_ε^2. For a fixed x value, the expected distribution of the y values, then, is a normal distribution centred on $\beta_0 + \beta_1 x$ and with variance σ_ε^2. Plugging in our estimates for $\beta_0, \beta_1, \sigma_\varepsilon^2$, we obtain the estimated expected distribution of y for a given x value. The estimated regression line, then, shows the estimated **conditional means** of y for the different values of x. Figure 9 illustrates this concept graphically. The cross-section of the 95% confidence band at a given x value is the 95% confidence interval of the corresponding conditional mean.

Let's look at a couple of examples of how we can interpret our model.

- According to the `aoa.lm` model, the estimated β parameters are 190.4 and -1.22. So

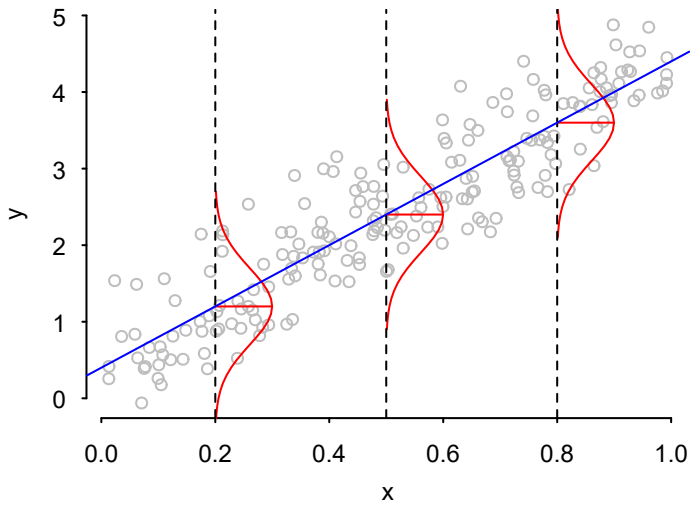

Figure 9: If we assume that the residuals are i.i.d., then the regression line connects the estimated conditional means for the distribution of y at different x-values. In this illustration, the residuals are all assumed to be normally distributed, but this is not a necessary assumption for making sense of the regression line. If the residuals aren't normally distributed, however, it's possible that the conditional means don't capture what's interesting about how x and y relate to each other.

according to this model, if we were to sample lots of participants with an AOA of 15, the best guess for their mean GJT score would be $190.4 - 1.22 \cdot 15 = 172.1$. The same answer can be obtained using `predict()`:

```
predict(aoa.lm, newdata = tibble(AOA = 15))

     1
172.14
```

Note, however, that there are two participants in the data set with an AOA of 15, and their mean GJT score isn't 172.1:

```
d |> filter(AOA == 15)

# A tibble: 2 x 6
  Participant   AOA   GJT Prediction Residual bs_outcome
  <chr>       <dbl> <dbl>      <dbl>    <dbl>      <dbl>
1 P49            15   180       172.     7.86       163.
2 P59            15   181       172.     8.86       160.
```

The extent to which our model-based estimate of the conditional GJT mean for an AOA

value of 15 is more accurate than the mean of these two observations depends on the accuracy of our modelling assumptions – especially the assumption that the AOA–GJT relationship is roughly linear. This doesn't seem too much of a stretch, and, conceptually, this assumption allows us to estimate the conditional mean for a given AOA value more accurately by leveraging the information about the relationship between AOA and GJT that we can extract from the other data points.

- There are no participants with an AOA of 21 in our sample. But according to our model, if we were to sample lots of participants with this AOA, their mean GJT would be about 165:

```
predict(aoa.lm, newdata = tibble(AOA = 21))
```

```
     1
164.83
```

This is an example of **intrapolation**, as we have both participants with lower and with higher AOA values.

- There are no participants with an AOA of 82 in our sample. But according to our model, if we were to sample lots of participants with this AOA, their mean GJT would be about 91:

```
predict(aoa.lm, newdata = tibble(AOA = 82))
```

```
     1
90.535
```

This is an example of **extrapolation**, since the maximum AOA in our sample is 71.

- The mean AOA in our sample is about 32.5. The estimated GJT mean for this AOA corresponds exactly to the GJT mean for the entire sample:

```
mean(d$GJT)
```

```
[1] 150.78
```

```
predict(aoa.lm, newdata = tibble(AOA = mean(d$AOA)))
```

```
     1
150.78
```

This is not a coincidence but a general phenomenon, the proof of which is provided in the footnote.[3]

Use your common sense when intra- and extrapolating. If we have a sample of participants

[3]The estimated conditional mean for the mean predictor value equals the sample mean of the outcome.

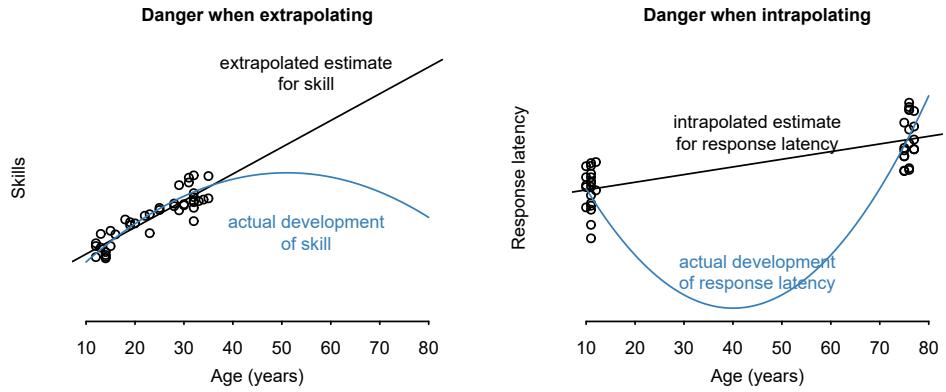

Figure 10: Dangers when extra- or intrapolating.

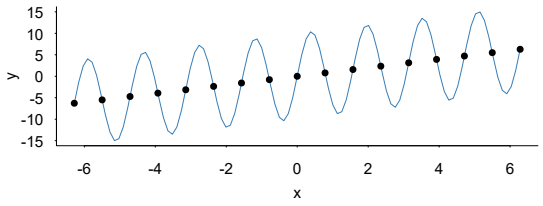

Figure 11: If we only observed the black dots, we'd probably think the relationship was linear. How much we'd be wrong when intrapolating between the measuring points depends on the amplitude of the wave.

aged 8–26, we'd be on thin ice extrapolating to participants aged 5 or 45; see Figure 10, left. The plot on the right illustrates the dangers with intrapolation. That said, even with densely sampled data it is still possible that a seemingly linear relationship is strongly nonlinear; see Figure 11.

Finally, a word on the meaning of the estimated intercept, $\widehat{\beta}_0$. The intercept represents the estimated conditional mean for participants with a predictor value of 0. However, zero lies outside the range of AOA values in our sample. A common trick to render the estimated intercept more informative about the data is to **centre** the predictor at some sensible value, usually the mean or the median. Centring just means subtracting this sensible value from all the predictor values before fitting the model:

Proof. According to the regression model, $y_i = \widehat{\beta}_0 + \widehat{\beta}_1 x_i + \widehat{\varepsilon}_i$ for $i = 1, 2, \ldots, n$. Hence we have

$$\frac{1}{n}\sum_{i=1}^{n} y_i = \frac{1}{n}\sum_{i=1}^{n}(\widehat{\beta}_0 + \widehat{\beta}_1 x_i + \widehat{\varepsilon}_i) = \frac{1}{n} n\widehat{\beta}_0 + \frac{1}{n}\widehat{\beta}_1 \sum_{i=1}^{n} x_i + \frac{1}{n}\sum_{i=1}^{n}\widehat{\varepsilon}_i.$$

We know that the mean of the estimated residuals is always zero when using OLS, so this simplifies to

$$\frac{1}{n}\sum_{i=1}^{n} y_i = \widehat{\beta}_0 + \widehat{\beta}_1 \left(\frac{1}{n}\sum_{i=1}^{n} x_i\right). \qquad \square$$

```r
# centre AOA
d$c.AOA <- d$AOA - mean(d$AOA)

# fit model again
aoa.lm <- lm(GJT ~ c.AOA, data = d)

# model coefficients
summary(aoa.lm)$coefficients
```

```
            Estimate Std. Error t value   Pr(>|t|)
(Intercept)  150.776    1.88073  80.169 1.1205e-73
c.AOA         -1.218    0.10514 -11.584 2.7282e-18
```

The advantage of centring is that the estimated intercept now represents the conditional mean of the outcome for the predictor value around which was centred (including the estimated standard error for this mean). If we centre around the mean predictor value, then we've already seen that this yields the sample mean.

Note that in order to compute the conditional mean for an AOA value of 35, we have to subtract the AOA mean from this predictor value, too, when we centred the predictor for our model:

```r
predict(aoa.lm, newdata = tibble(c.AOA = 35 - mean(d$AOA)))
```

```
     1
147.78
```

So not (!):

```r
predict(aoa.lm, newdata = tibble(c.AOA = 35))
```

```
     1
108.15
```

5 Assumptions and relevance, Episode 2

The assumptions we've made throughout this section (linear relationship, homoskedasticity, normality) almost never hold literally. Instead of worrying whether these assumptions literally hold (they don't), ask yourself whether your model is relevant to the questions you want to answer. The simple linear model seeks to characterise the linear relationship between the predictor and the outcome. If the relationship between predictor and outcome isn't approximately linear, the model's output will still be literally correct in many situations – but it may not help you find out what you want to know. In such cases, purely graphical data analyses or models capable of capturing nonlinearities (e.g., generalised additive models) may be of interest.

Similarly, if the spread around the regression line is much larger at one end of the scale than at the other, your model may still be literally correct. But since it assumes that the spread is homogeneous, it won't be able to pick up on this kind of difference. If you think such differences in spread are relevant to your research project, you may want to resort to models that not only capture conditional means but changes in the spread as well (e.g., generalised least-squares models).

As always, plotting the data prevents you and your readership from flying blind.

Summary

- The procedures introduced in Lecture 1 can be extended quite easily in order to model the relationship between two numeric variables. In the next lectures, we'll broaden the scope even more.
- The intercept represents the estimated outcome value when the predictor variable is fixed at 0. Depending on your data and research question, this may not be relevant information. But you can translate your predictor data to make the intercept more meaningful.
- The slope is the estimated difference between the outcome distributions of observations that differ by one unit in the predictor variable.

Exercises

1. (Paper and pencil.) Let's fit another model to DeKeyser et al.'s data:

```
gjt.lm <- lm(AOA ~ GJT, data = d)
summary(gjt.lm)$coefficients
```

```
             Estimate Std. Error t value   Pr(>|t|)
(Intercept) 112.33280   6.998618  16.051 8.2540e-26
GJT          -0.52922   0.045683 -11.584 2.7282e-18
```

   (a) What do the parameter estimates for `(Intercept)` and `GJT` mean – literally?

   (b) Which of the two models (`aoa.lm` or `gjt.lm`) is the most relevant in the context of DeKeyser et al.'s study? Why?

2. (With R.) The dataset `vanhove2014_cognates.csv` contains a summary of some data I collected for my PhD thesis (Vanhove, 2014). 163 speakers of German were asked to translate 45 written and 45 spoken Swedish words into German. The columns `CorrectSpoken` and `CorrectWritten` contain the number of correct translations per participants in both

modalities. The dataset `vanhove2014_background.csv` contains some background information about the participants, including their performance on a handful cognitive tests.

Let's first combine these datasets. (In addition to learning how to visualise data, learning to summarise, transform, and combine datasets is incredibly useful, but unfortunately out of the scope of this lecture series. See `https://r4ds.had.co.nz/` if you're interested in learning more about these topics.)

```
cognates <- read_csv(here("data", "vanhove2014_cognates.csv"))
background <- read_csv(here("data", "vanhove2014_background.csv"))
all_data <- cognates |>
  left_join(background, by = "Subject")
```

Now try to answer the following questions.

(a) The column `DS.Span` contains the participants' score on a working memory task. How is their performance on this task associated with `CorrectSpoken`?

(b) How is their performance on an English proficiency test (`English.Overall`) associated with `CorrectWritten`?

(c) How does their performance on the Swedish translation tasks vary with `Age` in both modalities?

References

DeKeyser, Robert, Iris Alfi-Shabtay & Dorit Ravid. 2010. Cross-linguistic evidence for the nature of age effects in second language acquisition. *Applied Psycholinguistics* 31. 413–438. doi:10.1017/S0142716410000056.

Vanhove, Jan. 2014. *Receptive multilingualism across the lifespan: Cognitive and linguistic factors in cognate guessing*: University of Fribourg dissertation. `http://doc.rero.ch/record/210293`.