Analysing the Effect of Age and Narrative Identity on Well-being

Dr Mark Andrews, Associate Professor of Statistical Methods, Nottingham Trent University, UK

Answer 1: Load and view the data

Our first problem above is to load up the data set from the file `ylc_df.csv`, look at the data frame to see how many columns (variables), how many rows (observations), and try to assess what each of the variables represent.

In R, we load the data frame from the csv file as follows:

```
ylc_df <- readCSV('ylc.csv')
```

Having loaded the data frame and assigned it to the name `ylc_df`, we can simply type this name to get a sense of the structure of the data:

```
ylc_df

# A tibble: 1,413 × 8
      ID sex      age ethnicity education         year wellbeing ac_value
   <dbl> <fct>  <dbl> <fct>     <fct>            <dbl>     <dbl>    <dbl>
 1     1 female  55.5 white     some-undergrad       1    -0.564  -0.580
 2     1 female  55.5 white     some-undergrad       2    -0.676  -0.580
 3     1 female  55.5 white     some-undergrad       3    -1.30   -0.580
 4     1 female  55.5 white     some-undergrad       4    -1.52   -0.580
 5     1 female  55.5 white     some-undergrad       5    -1.52   -0.580
 6     1 female  55.5 white     some-undergrad       6    NA      -0.580
 7     1 female  55.5 white     some-undergrad       7    NA      -0.580
 8     1 female  55.5 white     some-undergrad       8    NA      -0.580
 9     1 female  55.5 white     some-undergrad       9    NA      -0.580
10     2 female  56.0 white     some-undergrad       1     0.878   0.0630
# ℹ 1,403 more rows
```

We can see immediately (at the top of the output) that the data frame consists of 1413 rows and 8 columns. We can also see the first 10 values of each column and also that the data type of these columns are either `dbl`, which means numeric (`dbl` stands for "double" or "double precision", which is the coding scheme for decimal numbers in many coding languages), and `fct`, which means it is factor, or categorical variable.

In general, if there are a relatively large number of columns in the data frame, more than can be displayed comfortably in the R console, when we type the data frame's name like we just did, all columns won't be visible. Instead, at the bottom, we will see just the names of the columns. In this situation, we won't be able to see any of those columns' values. If we would like to see some of these values, however, we can use the command `glimpse` like this:

```
glimpse(ylc_df)
```

```
Rows: 1,413
Columns: 8
$ ID        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, …
$ sex       <fct> female, female, female, female, female, female, female, fema…
$ age       <dbl> 55.47814, 55.47814, 55.47814, 55.47814, 55.47814, 55.47814, …
$ ethnicity <fct> white, white, white, white, white, white, white, white, whit…
$ education <fct> some-undergrad, some-undergrad, some-undergrad, some-undergr…
$ year      <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, …
$ wellbeing <dbl> -0.5640279, -0.6759404, -1.3002230, -1.5161813, -1.5155756, …
$ ac_value  <dbl> -0.57989254, -0.57989254, -0.57989254, -0.57989254, -0.57989…
```

Now, the columns' values are displayed as rows. No matter how many columns there are in the data frames, their first few values will always be visible with `glimpse`. Therefore, `glimpse` is a useful and widely used tool to get a sense of the data frame that will work even with large data frames.

What each column represents is quite obvious in many cases. For example, we see a unique identifier for each participant (`ID`), and columns indicating the participants sex, age, ethnicity, level of education. We also have a column named `year`. The data is longitudinal so `year` indicates the year when each value of well-being (`wellbeing`) is measured for each participant. We see that we have 9 years of data for participant `ID=1`. In general, each participant was measured each year (`year`) for 9 years, although sometimes their data for any given year is missing. The column `ac_value` is a measure of the participant's average level of fullfilled motivated narrative identity. As described in Lind et al. (2024), a coding scheme was used on each participants' free text response data each year where they were asked to elaborate on life challenges the previous year. The authors coded this data for motivational themes of agency and communion and whether those motivation were fullfilled or thwarted. Participants with higher values of `ac_value` mean their narrative identities contain more fullfilled motivational themes.

In summary then, the variables in the data frame are as follows:

- `ID`: A unique identifier for each participant
- `sex`: Whether the participant is male or female
- `age`: The age of the participant in the first year of the study
- `ethnicity`: Their ethnicity using some major ethnic categories
- `education`: The highest level of education obtain by the participant
- `year`: The year, from 1 to 9, when the measurements were taken
- `welbeing`: A measure of the participant's mental well-being each year, measured using a standardized psychometric scale
- `ac_value`: The participant's average fulfilled motivation narrative identity score of the course of the study

Answer 2: Summarize the data, particularly demographic variables

In general, a very useful tool for quickly summarizing the variables in any data frame is `skim` from the `skimr` package, which we loaded already.

```
skim(ylc_df)
```

Data summary

Name	ylc_df
Number of rows	1413
Number of columns	8
Column type frequency:	
factor	3
numeric	5
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
sex	0	1	FALSE	2	fem: 909, mal: 504
ethnicity	0	1	FALSE	4	whi: 792, bla: 594, oth: 18, int: 9
education	0	1	FALSE	4	som: 630, com: 387, som: 333, hig: 63

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
ID	0	1.00	81.13	47.54	1.00	40.00	80.00	122.00	164.00	
age	18	0.99	56.41	0.95	54.79	55.62	56.20	57.08	59.09	
year	0	1.00	5.00	2.58	1.00	3.00	5.00	7.00	9.00	
wellbeing	66	0.95	0.01	0.89	-3.36	-0.59	0.20	0.68	1.53	
ac_value	0	1.00	0.01	0.41	-1.08	-0.28	0.05	0.31	0.92	

As you can see, `skim` provides a general overview of all the variables. We see how many variables we have, how many are numeric and how many are categorical. For each variable, we see how many missing values there are. For the categorical variables, we get a rough idea of the numbers of each different possible value. For the numeric variables, we get common descriptive statistics like mean, standard deviation, minimum value (`p0`; the 0th percentile), first quartile (`p25`; 25th percentile),the median (`p50`; 50th percentile), third quartile (`p75`; 75th percentile), maximum value (`p100`; the 100th percentile). We also get a small histogram for each numeric variable.

While `skim` is a great tool that probably should always be used to do a first quick look at the data, it is not always sufficient and so we may need to use other tools. For example, note how the `skim` output says that there are 909 values of `females` for `sex`. This does not mean that the data has 909 female participants. Because for each participant, we get up to 9 data points, the number of females or males in the `sex` variables overcounts these values, and likewise for the other demographic variables. We can, however, use many other tools to calculate the summary statistics for these demographics.

The following command will select the participant identifier, their sex, age, ethnicity, education. Because each participant has exactly one value for each of these, we can then remove any duplicates with the `distinct()` command.

```
ylc_demo_df <- select(ylc_df, ID, sex, age, ethnicity, education) |>
  distinct()

ylc_demo_df

# A tibble: 157 × 5
      ID sex      age ethnicity education
   <dbl> <fct>  <dbl> <fct>     <fct>
 1     1 female  55.5 white     some-undergrad
 2     2 female  56.0 white     some-undergrad
 3     3 female  55.5 white     some-postgrad
 4     4 female  56.4 white     completed-undergrad
 5     5 female  55.5 white     completed-undergrad
 6     6 female  56.0 white     some-postgrad
 7     7 male    56.1 white     completed-undergrad
 8     8 female  55.9 black     completed-undergrad
 9     9 female  55.6 white     some-undergrad
10    10 female  56.1 black     some-undergrad
# ℹ 147 more rows
```

Looking at `ylc_demo_df`, we see it has only 157 rows, which is the total number of participants. This is because the repetition over the multiple years in the study were removed.

As an aside, above we used the native R *pipe operator*: |>. This is obtained by typing a | symbol followed immediately by >. In RStudio and other IDEs, however, there are key shortcuts, usually CTRL + shift + m (command shift on Mac OS). It is a extremely useful operator when coding in R because it usually leads to much cleaner and more readable code. However, it does take some getting used to. There are countless guides online the pipe operators in R. There is, in fact, another pipe operator `%>%`, which in most respects works identically to |>, and so guides on either one are generally useful when learning how to use the pipe. For now, a quick summary is that the |> takes the value or object to its left and passes it to the function on the right. For example, the integers 1 to 10 can be obtained in R with `1:10`. If we wanted to calculate the mean of these numbers we could do `mean(1:10)`, which gives us 5.5. But we could also do the following:

```
1:10 |> mean()

[1] 5.5
```

The way to read this is that we send the data `1:10` to the function `mean()`, so you can read |> as *sends to*, or simply *to*.

In our `ylc_demo_df` code above, we first select five variables and then remove any duplicate rows by passing the resulting data frame to `distinct` (a command from the `dplyr` package, loaded when you load `tidyverse`) that removes any duplicate rows.

Now, using `ylc_demo_df` we can calculate some descriptive statistics for the demographic variables. For this, we will first use `count` (also from `dplyr`). For example, to count the number of males and females, we can do this (again using the |>):

```
ylc_demo_df |> count(sex)

# A tibble: 2 × 2
  sex        n
  <fct>  <int>
1 male      56
2 female   101
```

If we want the percentage of men and women, we can add a new column named `percnt`. For this, we will use the `mutate` function from `dplyr`, which is used to add new columns to a data-frame or edit existing ones.

```
ylc_demo_df |> count(sex) |> mutate(percnt = round(100 * n/sum(n), 1))

# A tibble: 2 × 3
  sex        n percnt
  <fct>  <int>  <dbl>
1 male      56   35.7
2 female   101   64.3
```

What the `mutate` command does here is it divides the values of `n` (giving the count of each sex) by the sum of the `n` column, then multiplies by 100, and rounds to one decimal place. As such, we can see that the percentage of females is 64.3%.

We can now follow the same procedure for `ethnicity` and `education`:

```
ylc_demo_df |> count(ethnicity) |> mutate(percnt = round(100 * n/sum(n), 1
))

# A tibble: 4 × 3
  ethnicity       n percnt
  <fct>       <int>  <dbl>
1 white          88   56.1
2 black          66   42
3 inter-racial    1    0.6
4 other           2    1.3

ylc_demo_df |> count(education) |> mutate(percnt = round(100 * n/sum(n), 1
))

# A tibble: 4 × 3
  education              n percnt
  <fct>              <int>  <dbl>
1 high-school            7    4.5
2 some-undergrad        37   23.6
3 completed-undergrad   43   27.4
4 some-postgrad         70   44.6
```

For the numeric age variable, we can calculate the mean, median, standard deviation, and any other summary statistics, using the `summarize` function:

```
ylc_demo_df |>
  summarize(avg = mean(age, na.rm = TRUE),
            median = median(age, na.rm = TRUE),
            stdev = sd(age, na.rm = TRUE))
```

```
# A tibble: 1 × 3
    avg median stdev
  <dbl>  <dbl> <dbl>
1  56.4   56.2 0.956
```

Here, we calculate the mean, median, and standard deviation of age with the functions `mean()`, `median()`, and `sd()`, respectively. Note that in each case, we use `na.rm = TRUE` in the function call. This removes any missing values (`NA` values) from the age column before doing the summary statistic calculation. If there are missing values in a column, and we can see from `skim` that there are missing values in the age column, any summary statistic will return a value of `NA` unless we first do `na.rm = TRUE`.

From this summary, we can see that our participants consist of middle-aged people, about 2 in every 3 of whom are female, mostly either white or black ethnicity, and relatively well educated.

Answer 3: Visualizing trends

We start by visualizing the trend in well-being scores over the 9 years of the study, which is shown in Figure Figure 1. In particular, for each year, we make a Tukey boxplot showing the distribution of scores of `wellbeing`. A Tukey boxplot shows the median (central horizontal line), upper and lower quartiles (upper and lower edges of box). The thin whiskers give the range of values out to what it defines as non-outliers, which are values within 1.5 times inter-quartile range above or below the upper or lower quartiles:

```
ggplot(ylc_df, aes(x=year, y=wellbeing, group = year)) + geom_boxplot() +
  scale_x_continuous(breaks = 1:9)
```

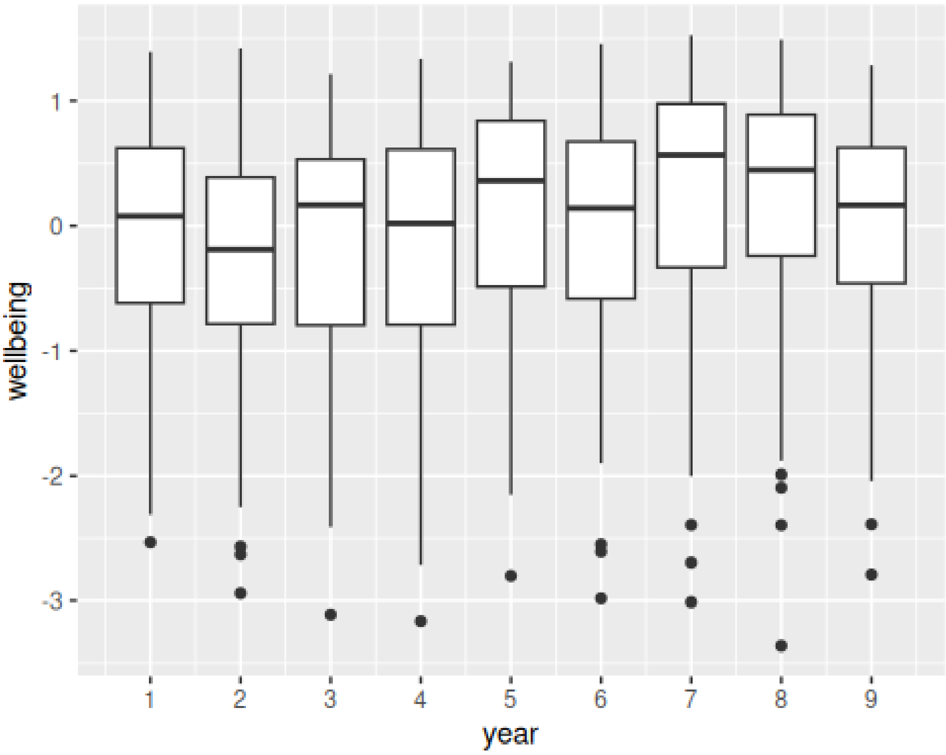

Figure 1: For each year of the study, Tukey boxplot distribution of well-being scores.

The code here uses the powerful `ggplot` command. There is a lot that this can do but it does take some practice and experience to get comfortable with it. In brief, here we are telling `ggplot` to plot `year` on the x-axis and `wellbeing` on the y-axis. We indicate `group = year` to inform it that we want one boxplot per year. The second line with `scale_x_continuous` simply indicates that we would like a break point at each year value from 1 to 9.

This plot in Figure Figure 1 is very informative. We see a potential upward trend in well-being scores as participants age. This upward trend is not overwhelmingly clear, however, so we will need to do careful statistical analysis before we conclude anything more definitively. Nonetheless, so far, we do see a potential upward trend in well-being with age.

Now let us look at how a participant's average well-being score varies by their `ac_value` score. For this, we need to first calculate each person's average well-being scores, which we do below with `summarize`. Note that we also add each person's `ac_value` score, which is already their mean over time, to the data frame that is returned. The `.by = ID` means that we calculate one mean per each value of `ID`. We then pipe this data frame to `ggplot` to produce a scatterplot (with `geom_point`) and a line of best fit (with `stat_smooth(method = 'lm')`):

```
ylc_df |>
  summarize(wellbeing = mean(wellbeing, na.rm=T), ac_value = ac_value, .by
= ID) |>
  ggplot(aes(x = ac_value, y = wellbeing)) + geom_point() +
  stat_smooth(method = 'lm')
```

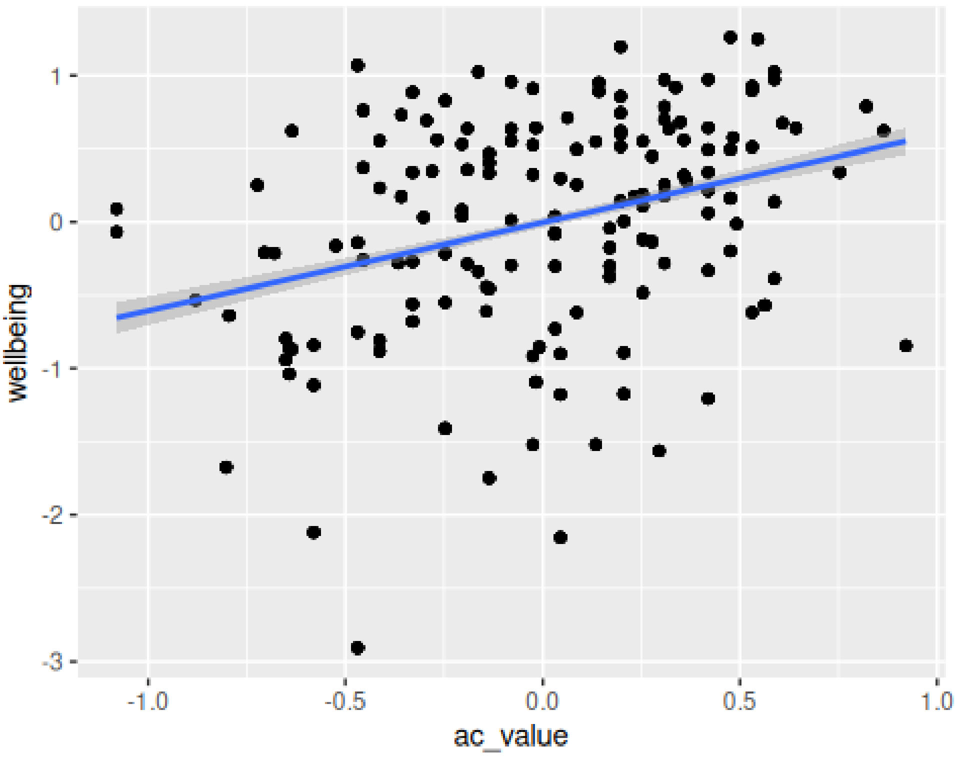

Figure 2: Scatterplot of participants' well-being and `ac_value` scores.

Clearly we can see a potential upward trend: higher values of ac_value correspond to higher average values of well-being.

Another way of looking at the relationship between motivational narrative identity and well-being is to first calculate whether each participant's ac_value is above or below the median value of ac_value in the sample. We can do this as follows:

```
ylc_df <- mutate(ylc_df, motivated = ac_value > median(ac_value))
ylc_df

# A tibble: 1,413 × 9
      ID sex      age ethnicity education      year wellbeing ac_value mot
ivated
   <dbl> <fct>  <dbl> <fct>     <fct>         <dbl>     <dbl>    <dbl> <lg
l>
 1      1 female  55.5 white     some-undergr…     1    -0.564   -0.580  FAL
SE
 2      1 female  55.5 white     some-undergr…     2    -0.676   -0.580  FAL
SE
 3      1 female  55.5 white     some-undergr…     3    -1.30    -0.580  FAL
SE
 4      1 female  55.5 white     some-undergr…     4    -1.52    -0.580  FAL
SE
 5      1 female  55.5 white     some-undergr…     5    -1.52    -0.580  FAL
SE
 6      1 female  55.5 white     some-undergr…     6    NA       -0.580  FAL
SE
 7      1 female  55.5 white     some-undergr…     7    NA       -0.580  FAL
```

```
SE
 8      1 female  55.5 white     some-undergr…     8     NA       -0.580  FAL
SE
 9      1 female  55.5 white     some-undergr…     9     NA       -0.580  FAL
SE
10      2 female  56.0 white     some-undergr…     1      0.878   0.0630 TRU
E
# ℹ 1,403 more rows
```

The value of `motivated` takes on values of `TRUE` or `FALSE` depending on whether the participant's `ac_value` is above the overall median. With this variable, we can now look at the distribution of the average well-being scores, using a boxplot, separately for the participants above and below the median value of `ac_value`. We can do this as follows, and is shown in Figure Figure 3.

```
ylc_df |>
  summarize(wellbeing = mean(wellbeing, na.rm=T), motivated=motivated, .by
= ID) |>
  ggplot(aes(x = motivated, y = wellbeing)) + geom_boxplot()
```

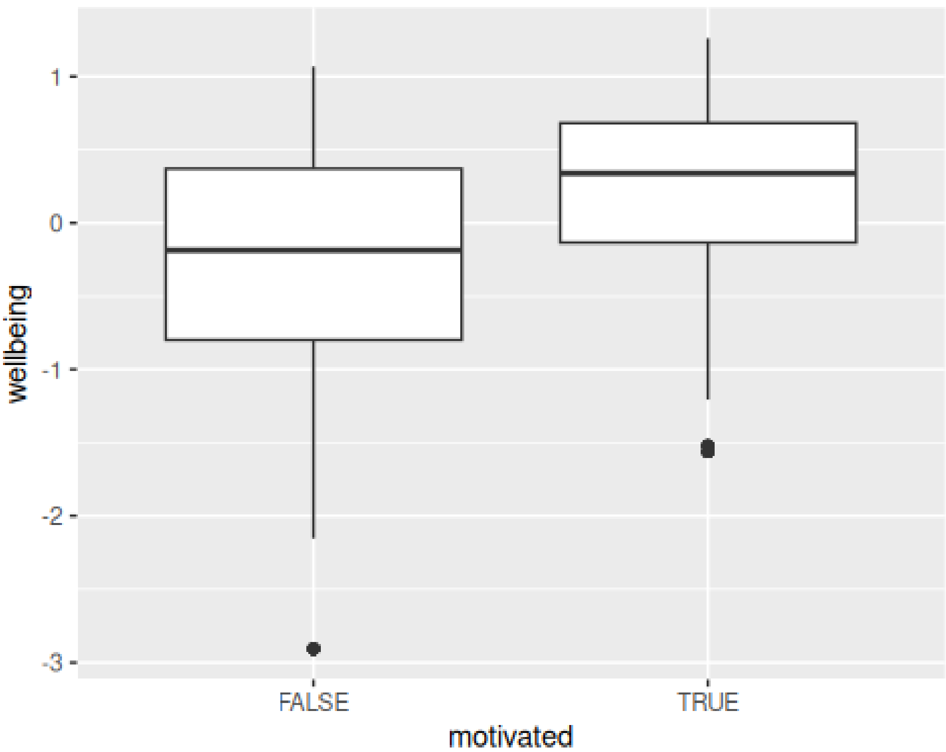

Figure 3: Tukey boxplot distribution of well-being scores depending on whether the participant's `ac_value` score is above or below the overall median.

From this plot, it is apparent that participants with above median value of `ac_value` tend to have higher average scores of well-being.

We can now look at trends by age and by the value of `motivated` simultaneously. The following code, shown in Figure Figure 4, plots two boxplots each year, one for those participants whose value of ac_value is above the median and another for those whose ac_value is below the median.

```
ggplot(ylc_df, aes(x=year,
                   y=wellbeing,
                   fill=motivated,
                   group = interaction(motivated, year))) + geom_boxplot()
+
  scale_x_continuous(breaks = 1:9) +
  theme(legend.position = 'bottom')
```

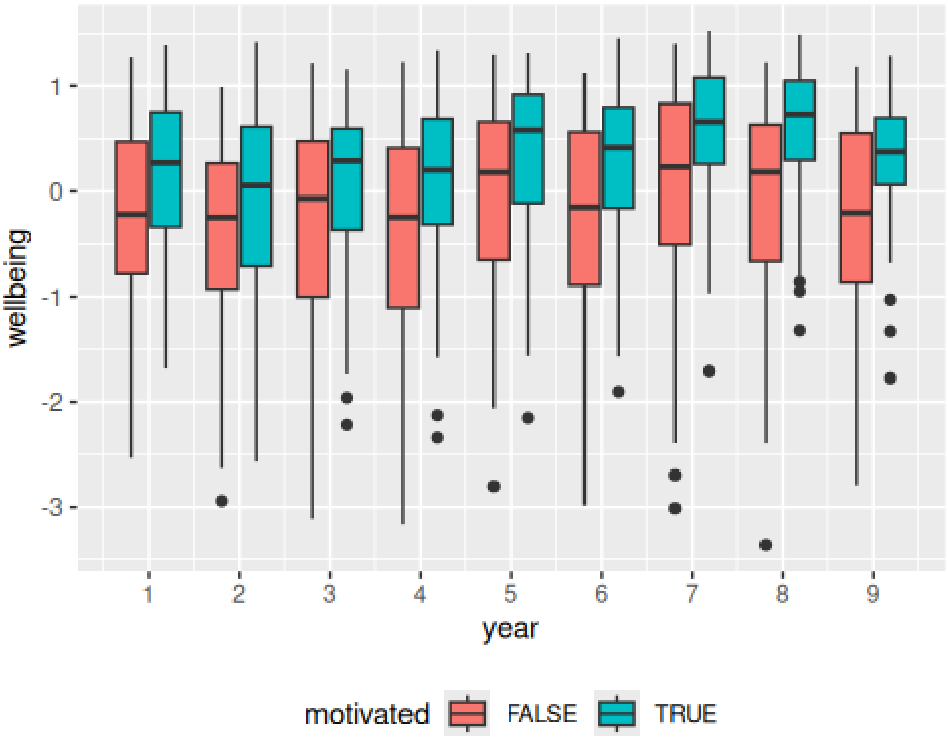

Figure 4: For each year of the study, Tukey boxplot distribution of well-being scores depending on whether the participant's `ac_value` score is above or below the overall median.

The code here uses `fill=motivated` to colour code the boxplot depending on the value of `motivated`. We also group by both the `motivated` and the `year` variable using the `interaction` function, which is used when we combine different grouping variables. The last line of the code put the legend below the plot, rather than to the side.

From this plot, it appears as if a) well-being scores increase with age b) well-being scores are higher if the `ac_value` is above the median and c) the increase in well-being with age is slightly higher/steeper for those participants with above median values of `ac_value`. Of course, this is just how things roughly appear from the visualization. Important as visualizations are, to be able to draw more definitive conclusions, we need to conduct a statistical analysis that tests hypotheses about the effect of age, motivated narrative identity, and their interaction.

Answer 4: Statistical analysis

Recall that Problem 4 asked us to perform any analysis to address the following: 1. Does well-being vary by age? 2. Does well-being vary by narrative identity? 3. Is there an interaction between age and narrative identity?

Repeated measures ANOVA

One possible analysis method that we can use to address these questions a repeated-measures two-way ANOVA. A two-way ANOVA will allow us to test if there is change in well-being scores with age, a change in well-being scores with the value of `motivated` (i.e. above or below median values of `ac_value`), and also to test their interaction. We must, however, take into account the fact that each participant gives us multiple well-being scores, i.e. we have repeated measures of well-being, hence we must do a repeated-measures two-way ANOVA.

We can do this repeated-measures ANOVA using the `aov_car` command from the `afex` package, which we loaded above.

```
M_anova <- aov_car(wellbeing ~ motivated + Error(ID/factor(year)), data =
ylc_df)
```

Before we examine the results, let us discuss thec command's code. The `Error` function is used to specify the repeated measures. In this case, with `Error(ID/factor(year))`, we are saying that for each participant (`ID`), we have multiple values of well-being each `year`. Note, we state `factor(year)` instead of `year` only because ANOVA requires that the independent variables are categorical and `year` is a number and so we must convert it to a categorical variable first.

We can view the results by typing the name of the saved result:

```
M_anova

Anova Table (Type 3 tests)

Response: wellbeing
           Effect           df  MSE         F  ges p.value
1       motivated       1, 139 4.88 16.04 *** .079   <.001
2            year 5.92, 823.43 0.28 20.63 *** .037   <.001
3 motivated:year 5.92, 823.43 0.28    2.37 * .004    .029
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1

Sphericity correction method: GG
```

There are three hypothesis being tested: the main effect of age (`year`) on well-being, the main effect of `motivated` on well-being, and the interaction of these two variables. The hypothesis test is based on a F statistic, listed under `F`, which has the two degrees of freedom listed under `df`. These degrees of freedom have been corrected for sphericity violations using the Greenhouse-Geisser (GG) epsilon correction. Sphericity is a rather technical assumption in repeated measures ANOVA, but if there are any violations of this assumption, they are automatically fixed with the epsilon correction.

The corresponding p-value is in the last column. We can see that each of the three hypothesis is significant. From this, we can conclude that well-being changes with age, well-being changes with the value of `motivated`, and that there is also an interaction of age and `motivated`. In particular, the interaction means that the change in well-being over time is different for the two different values of `motivated`.

We can calculate the estimated value of `wellbeing` for each value of year, separately for each value of `motivated` as follows:

```
emmeans(M_anova, specs = ~ year | motivated)

motivated = FALSE:
 year   emmean     SE  df lower.CL upper.CL
 X1   -0.25427 0.0959 139  -0.4439  -0.0646
 X2   -0.39542 0.1020 139  -0.5972  -0.1937
 X3   -0.32781 0.1030 139  -0.5319  -0.1238
 X4   -0.42123 0.1060 139  -0.6298  -0.2126
 X5   -0.06963 0.1050 139  -0.2763   0.1371
 X6   -0.29648 0.0993 139  -0.4928  -0.1002
 X7    0.00891 0.1110 139  -0.2097   0.2276
 X8   -0.10834 0.0960 139  -0.2981   0.0815
 X9   -0.27403 0.0932 139  -0.4582  -0.0898

motivated = TRUE:
 year   emmean     SE  df lower.CL upper.CL
 X1    0.14813 0.0966 139  -0.0429   0.3392
 X2   -0.05035 0.1030 139  -0.2535   0.1528
 X3    0.07299 0.1040 139  -0.1325   0.2785
 X4    0.09658 0.1060 139  -0.1135   0.3067
 X5    0.32324 0.1050 139   0.1151   0.5314
 X6    0.28924 0.1000 139   0.0915   0.4870
 X7    0.54143 0.1110 139   0.3212   0.7616
 X8    0.59203 0.0967 139   0.4009   0.7832
 X9    0.31894 0.0938 139   0.1334   0.5045

Confidence level used: 0.95
```

Usually after we perform an ANOVA, we do pairwise comparisons. In this current analysis, this will allow us to identity how average well-being is different between any two years and for any value of `motivated`. One difficulty with this analysis is that there are 9 values of `year` so there 36 pairwise year comparisons for each value of `motivated` and that gives us 72 separate tests. Even though these tests will be corrected for multiple comparisons, it is quite challenging to see how well-being changes over time for the different values of `motivated`.

Linear mixed effects

One way to allow us to better understand the change in well-being with age and how this differs with the values of `motivated` is perform an alternative analysis called a linear mixed effects model. Linear mixed effects models are in fact related to repeated measures ANOVA but have, in general, much greater flexibility.

The linear mixed effects model that we will perform assumes that there is, on average, a linear relationship between well-being and age. Of course, by looking at plots like Figure Figure 1 and Figure Figure 4, the trends are not clearly exactly linear, but by modelling it as linear, we can capture the average year-on-year increase of well-being. However, for any on participant, their linear trend in well-being may vary randomly from the average participant: their trend line may be slightly steeper or less steep, for example. Linear mixed effects models allow us to simultaneously measure population average effects, such as how for the average participant, well-being increases with age, and also how these trends vary randomly between different participants.

For linear mixed effects analysis, sometimes it helps to centre or scale continuous variables and so we will create a new variable `year_c`, which is the value of `year` minus 5. Thus, -4 of `year_c` is the first year of the study and +4 is the final year.

```
ylc_df <- ylc_df |> mutate(year_c = year - 5)
```

The analysis is the done as follows, using `lmer` from the `lmerTest` package (which is a thin wrapper around `lmer` from the `lme4` package, on which `lmerTest` is built):

```
M_lme <- lmer(wellbeing ~ year_c * motivated + (year_c|ID), data = ylc_df)
```

The `(year_c|ID)` code described the so-called *random effects* of the model. In this case, it means that we assume the linear relationship betweeen `year_c` and well-being varies randomly across the different participants. In other words, each participant has a different value of the intercept and slope describing the relationship between `year_c` and well-being. The variation in these random intercepts and slopes across participants is assumed to be normally distributed.

The `year_c * motivated` code described the *fixed effects* of the model. Another way to describe this is that is specifies the population average effects of the model. In other words, it models how for the average participant, well-being variables by `year_c` and `motivated` and their interaction. Note that `year_c` is a continuous variable with values from -4 to +4. An interaction of categorical variable (`motivated`) and a continuous variable (`year_c`) effectively means that the slope and intercept of the effect of the continuous variable changes with each different value of the categorical variable.

The summary of this analysis can be obtained as follows:

```
summary(M_lme)

Linear mixed model fit by REML. t-tests use Satterthwaite's method [
lmerModLmerTest]
Formula: wellbeing ~ year_c * motivated + (year_c | ID)
   Data: ylc_df

REML criterion at convergence: 2265

Scaled residuals:
    Min      1Q  Median      3Q     Max
-5.6624 -0.4947  0.0477  0.5588  3.9480

Random effects:
```

```
 Groups   Name        Variance Std.Dev. Corr
 ID       (Intercept) 0.503739 0.70975
          year_c      0.002953 0.05434  -0.01
 Residual             0.202210 0.44968
Number of obs: 1347, groups:  ID, 157

Fixed effects:
                       Estimate Std. Error         df t value Pr(>|t|)
(Intercept)           -0.221535   0.080445 156.050141  -2.754  0.00659 **
year_c                 0.025939   0.009068 149.940066   2.861  0.00483 **
motivatedTRUE          0.478619   0.116295 155.608013   4.116 6.25e-05 ***
year_c:motivatedTRUE   0.034801   0.013032 149.147805   2.670  0.00842 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            (Intr) year_c mtTRUE
year_c       0.013
motivtdTRUE -0.692 -0.009
yr_c:mtTRUE -0.009 -0.696  0.010
```

Let's first focus on the table of coefficients under `Fixed effects:`. We interpret this as a multiple linear regression coefficients table, albeit one with one categorical variable, one continuous variable, and their interaction. The estimate value of `year_c`, 0.0259389, gives the slope of the linear effect of `year_c` on well-being when `motivated = FALSE`. In other words, each year, well-being increases on average by 0.0259389 for participants whose value of `motivated` is `FALSE`, i.e. those below median `ac_value`. The estimate value of `year_c:motivatedTRUE`, 0.0348007, gives the *change* in the slope, relative to when `motivated` is `FALSE`, of the linear effect of `year_c` on well-being when `motivated = TRUE`. In other words, each year, well-being increases on average by 0.0259389 + 0.0348007, or 0.0607396, for participants whose value of `motivated` is `TRUE`, i.e. those above the median `ac_value`. The estimate value of `(Intercept)`, -0.2215351, gives the intercept, which is the average value of well-being when `year_c` is 0, or year 5, when `motivated = FALSE`. In other words, on average for participants with below median values of `ac_value`, their average well-being score on year 5 is -0.2215351. Finally, the estimate value of `motivatedTRUE`, 0.4786189, gives the *change* in the intercept when from when `motivated = FALSE` to when `motivated = TRUE`. In other words, on average for participants with above median values of `ac_value`, their average well-being score on year 5 is -0.2215351 + 0.4786189 or 0.2570838.

Note how all these effects are significant. In particular, the interaction effect, `year_c:motivatedTRUE` tells us that there is a significant increase in the slope of the effect of age on well-being when `motivated = TRUE` compared to when `motivated = FALSE`. The significant `motivatedTRUE` effect effect tells us there is a significant increase in average well-being when `motivated = TRUE` compared to when `motivated = FALSE`. The significant `year_c` effect tells us that even when `motivated = FALSE`, there is still a non-zero slope in the linear relationship between year and well-being. In other words, even when `motivated = FALSE`, well-being increases on average with age.

We can use emmeans to calculate the estimated value of well-being each year for both values of motivated as follows:

```
emmeans(M_lme, specs = ~ year_c + motivated, at = list(year_c = seq(-4, 4)
))

 year_c motivated  emmean     SE  df lower.CL upper.CL
     -4 FALSE     -0.3253 0.0878 155  -0.4987 -0.15185
     -3 FALSE     -0.2994 0.0846 155  -0.4664 -0.13228
     -2 FALSE     -0.2734 0.0822 155  -0.4358 -0.11098
     -1 FALSE     -0.2475 0.0808 155  -0.4072 -0.08779
      0 FALSE     -0.2215 0.0804 155  -0.3805 -0.06262
      1 FALSE     -0.1956 0.0811 155  -0.3558 -0.03543
      2 FALSE     -0.1697 0.0827 155  -0.3330 -0.00627
      3 FALSE     -0.1437 0.0853 155  -0.3122  0.02474
      4 FALSE     -0.1178 0.0887 154  -0.2930  0.05745
     -4  TRUE      0.0141 0.0917 155  -0.1671  0.19532
     -3  TRUE      0.0749 0.0884 155  -0.0997  0.24944
     -2  TRUE      0.1356 0.0859 155  -0.0341  0.30534
     -1  TRUE      0.1963 0.0844 155   0.0295  0.36315
      0  TRUE      0.2571 0.0840 154   0.0912  0.42299
      1  TRUE      0.3178 0.0846 154   0.1508  0.48489
      2  TRUE      0.3786 0.0862 154   0.2083  0.54880
      3  TRUE      0.4393 0.0887 153   0.2640  0.61462
      4  TRUE      0.5000 0.0922 152   0.3179  0.68218

Degrees-of-freedom method: kenward-roger
Confidence level used: 0.95
```

More useful is if we plot these values by piping the data outputted by emmeans to ggplot. In the following code, shown in Figure Figure 5, will plot the linear relationship between year_c and well-being separately for each value of motivated. Shown also are the confidence intervals.

```
emmeans(M_lme, specs = ~ year_c + motivated, at = list(year_c = seq(-4, 4)
)) |>
  as_tibble() |>
  ggplot(aes(x = year_c, y = emmean, colour = motivated)) +
    geom_ribbon(aes(ymin = lower.CL, ymax = upper.CL, fill = motivated), a
lpha = 0.25) +
    geom_point() +
    geom_line() +
    theme(legend.position = 'bottom')
```

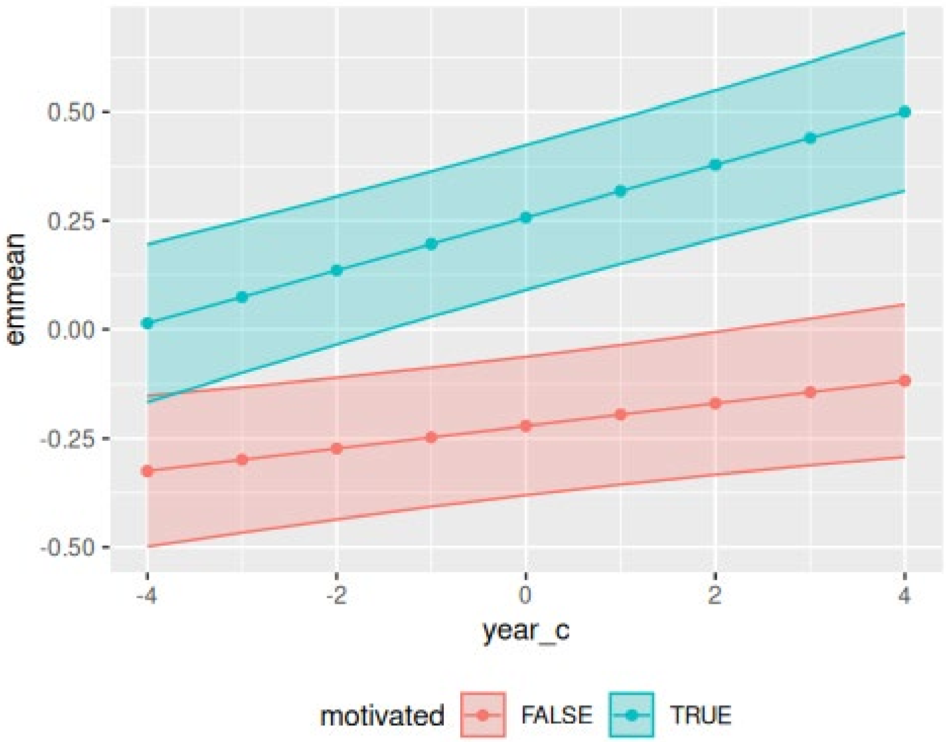

Figure 5: The change in the intercept and slope of the linear relationship between age and well-being depending on the value of `motivated`

In thr `M_lme`, there are also *Random Effects*. The main thing that these results tell us is how much much variability across individuals there is in general linear relationship between age and narrative identity on well-being. In particular, the standard deviation of the normal distribution of variability of the intercepts across different participants is 0.7097454, while the standard deviation of the normal distribution of the variability in slopes across participants is 0.0543432.