Tutorial: Multilevel and Hierarchical Modeling (Part 1)

Chao-Yo Cheng
28 February 2021

1 Introduction

By the end of today, you will know how to:

- Fit a (linear) multi-level model.
- Allow the intercepts and/or slopes to vary in a linear multi-level model.

2 Packages

Today we will use the `lme4` package, which includes a variety of functions to fit linear and non-linear multilevel models.

The `lme4` package does not report p-values in model summary tables (recall that there is some controversies around the use of p-values). If we want to report p-values for our models, we can supplement `lme4` with the `lmerTest` package.

Finally, we will use the `lattice` and `ggplot2` packages to visualize the results.

We will also need `dplyr` for data wrangling. Alternatively, you can load the `tidyverse` package.

```
library(lme4) # for regression
library(lmerTest) # for p-value
library(lattice) # for visualization
library(ggplot2) # for visualization
library(tidyverse) # for data wrangling
```

3 Data

We will study **World Values Survey (WVS)** today – one of the largest survey projects in about 100 countries. WVS started as the 1981 European Values Study (EVS) (led b Jan Kerkhofs and Ruud de Moory) and was later expanded under the leadership of Ronald Inglehart at the University of Michigan.

WVS "explores people's values and beliefs, how they change over time, and what social and political impact they have." For more information about WVS, click here.

Today we will be studying Wave 7 (2017-2020). It is a huge dataset and includes variables from other cross-national political and socioeconomic datasets (e.g., Varieties of Democracy). You can find the complete codebook on Moodle along with other tutorial materials.

For the purpose of demonstration, we will focus on the following variables in this tutorial.

- `happy` (ordinal): Happiness; the variable ranges from 4 to 1; "4" means *very happy* and "1" means *not at all happy*.
- `edu` (ordinal): Highest educational level of the respondent; the variable ranges between 1 and 8, with 8 referring to the highest level of education (doctoral or equivalent). This will be our main **individual-level** predictor today. This variable is recoded based on `Q275` in the original dataset.

- `regime_type` (categorical): Regime type of the country. This variable is provided by the Polity V dataset. Each country is assigned into one of the following categories: Autocracy, closed anocracy, open anocracy, and democracy. Anocracy, by definition, refers to political regimes that exhibit both some democratic and autocratic charateristics. This will be our main **group-level** predictor today.

  Question: How else can we group the respondents in the survey? And why?

Let's read in the data and take a look at the variables. We will use `readRDS()` to import the `.RData` file.

```
dta <- readRDS("WVS_7.RData")
```

Now we can carry out some data wrangling.

```
dta_new <- dta %>%
  dplyr::select(Q46P, Q275, regtype) %>%
  rename(edu = Q275) %>%
  mutate(Q46P = ifelse(Q46P <= 0, NA, Q46P),
         edu = ifelse(edu <= 0, NA, edu),
         regtype = ifelse(regtype <= 0, NA, regtype)) %>%
  mutate(regtype = as.character(regtype),
         Q46P = as.character(Q46P)) %>%
  mutate(happy = recode(Q46P,
                        "4" = "1",
                        "3" = "2",
                        "2" = "3",
                        "1" = "4"),
         reg_type = recode(regtype,
                           "5" = "full democracy",
                           "4" = "democracy",
                           "3" = "open anocracy",
                           "2" = "closed anocracy",
                           "1" = "autocracy")) %>%
  mutate(happy = as.numeric(happy)) %>%
  tidyr::drop_na(regtype, happy)
#table(dta_new$happy, dta_new$Q46P)
```

Question: Can you walk through the data wrangling process verbally? Any suggestion to make the process more efficiently?

For more information – you can consult the book *R for Data Science* (https://r4ds.had.co.nz/index.html). Cheatsheets are also available here: https://www.rstudio.com/resources/cheatsheets/.

Below is a step-by-step explanation:

- Use `select()` to choose the variables we need from `dta`.
- Use `rename()` to rename `Q275`.
- Use `mutate()` four times
  - first, for all variables we change the negative values to `NA` (see the codebook)
  - next, we change the variables to "character" (from factor) so it is easier for the next manipulation
  - third, change the variables into ordinal ones
  - finally, convert `happy` from character to "numeric" objects so we can use it for the dependent variable.

Question: What does `tidyr::drop_na()` do?

We use this to drop all observations with NA for any of these three variables in `dta_new`.

Let's check if the re-coding is done properly.

```
dta_new$reg_type <- factor(dta_new$reg_type,
                           levels=c("autocracy", "closed anocracy", "open anocracy", "democracy", "full
table(dta_new$reg_type, dta_new$regtype)
```

```
##
##                       1     2     3     4     5
##   autocracy        8143     0     0     0     0
##   closed anocracy     0  9508     0     0     0
##   open anocracy       0     0  6635     0     0
##   democracy           0     0     0 32880     0
##   full democracy      0     0     0     0 14078
```

Question: What does `factor()` do?

Question: How may respondents from different political regimes act differently in the survey?

Everything looks good. Let's proceed.

```
head(dta_new, 10)
```

```
##    Q46P edu regtype happy       reg_type
## 1     3   1       5     2 full democracy
## 2     4   1       5     1 full democracy
## 3     3   4       5     2 full democracy
## 4     3   3       5     2 full democracy
## 5     2   3       5     3 full democracy
## 6     3   3       5     2 full democracy
## 7     3   2       5     2 full democracy
## 8     4   3       5     1 full democracy
## 9     3   1       5     2 full democracy
## 10    2  NA       5     3 full democracy
```

We can check the object type of each variable by calling them out separately.

```
class(dta_new$regtype)
```

```
## [1] "character"
```

```
class(dta_new$reg_type)
```

```
## [1] "factor"
```

Or, we can use the `sapply()` function to do the same in one go.

```
sapply(dta_new, class)
```

```
##        Q46P         edu     regtype       happy    reg_type
## "character"   "integer" "character"   "numeric"    "factor"
```

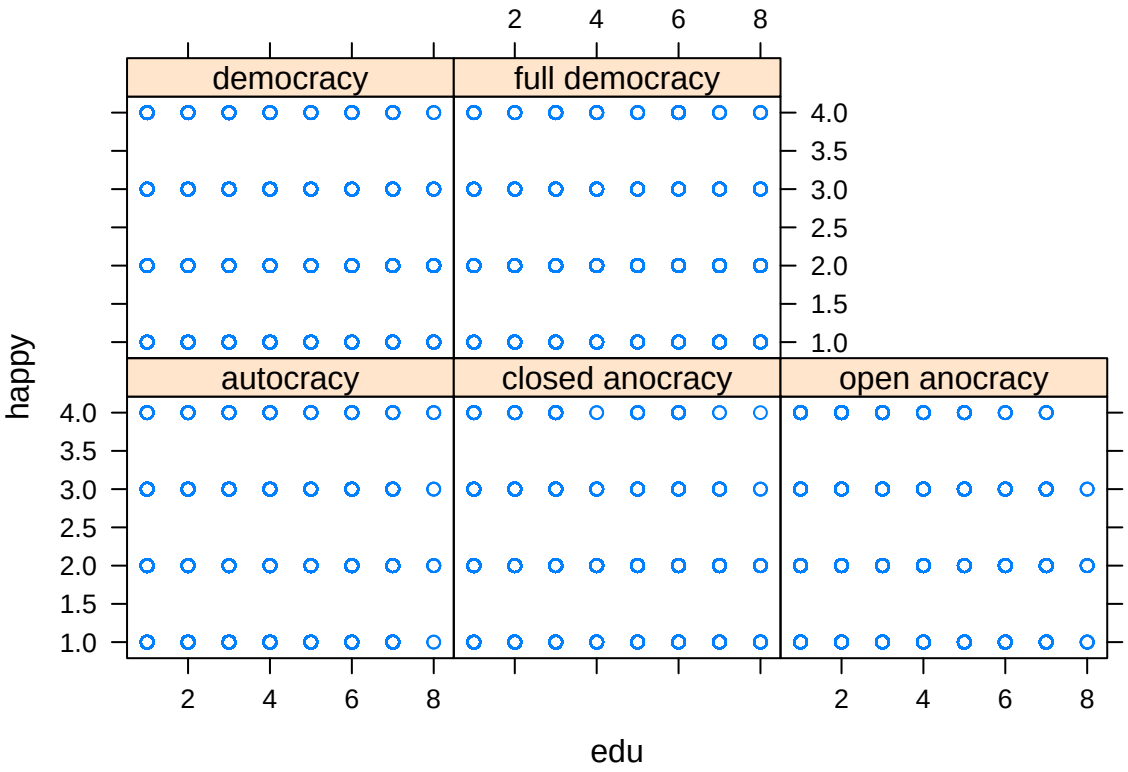
Question: We enter two arguments into `sapply()` – what does each of them refer to?

Question: What is the difference between "character" and "factor"?

4 Visualization

Let's plot the data using the `xyplot()` function from the `lattice` package to see how the relationship between `edu` and `happyion` varies by different regime types.

```
xyplot(happy ~ edu | reg_type, dta_new)
```

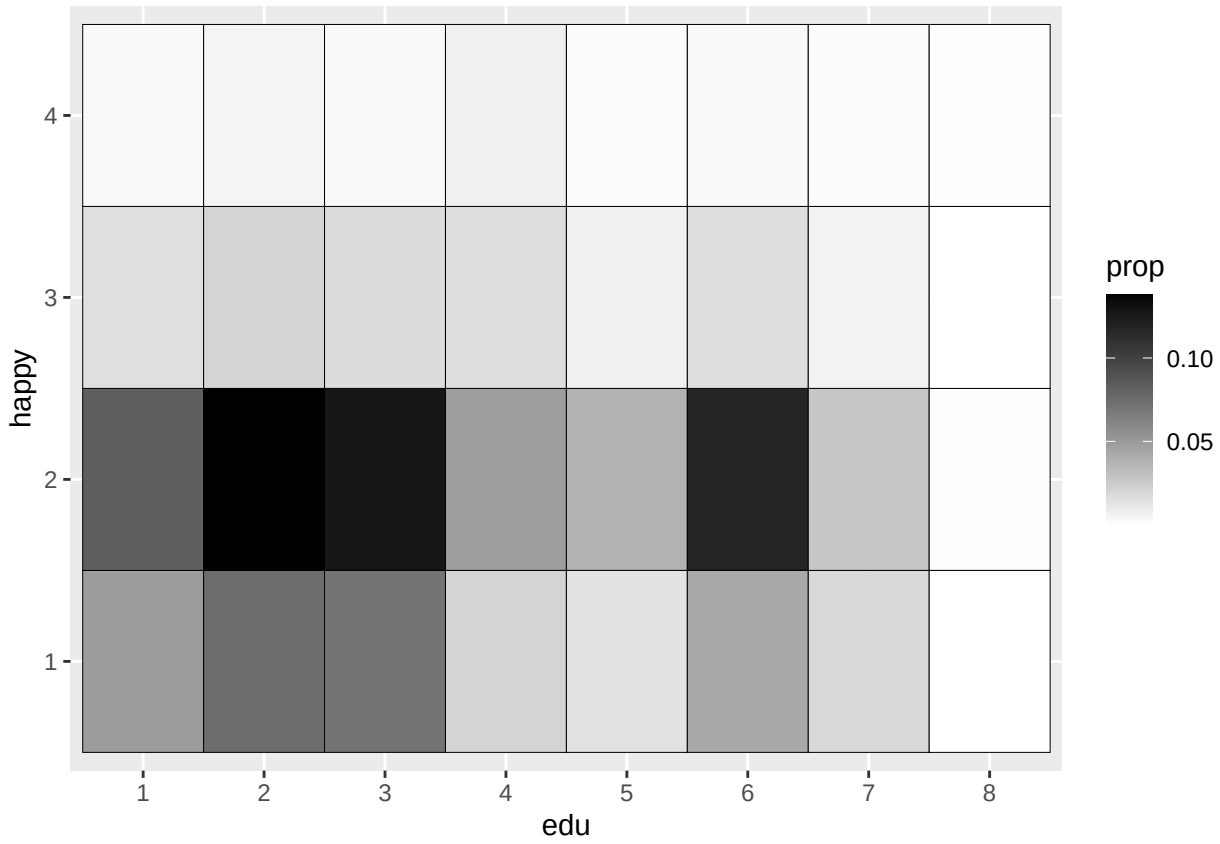


Question: They do not look too informative – what is going on here?

Sometimes, we can try heatmaps to visualize the relationship between two **ordinal** variables. For more discussion, check out this page and this CrossValidated post.

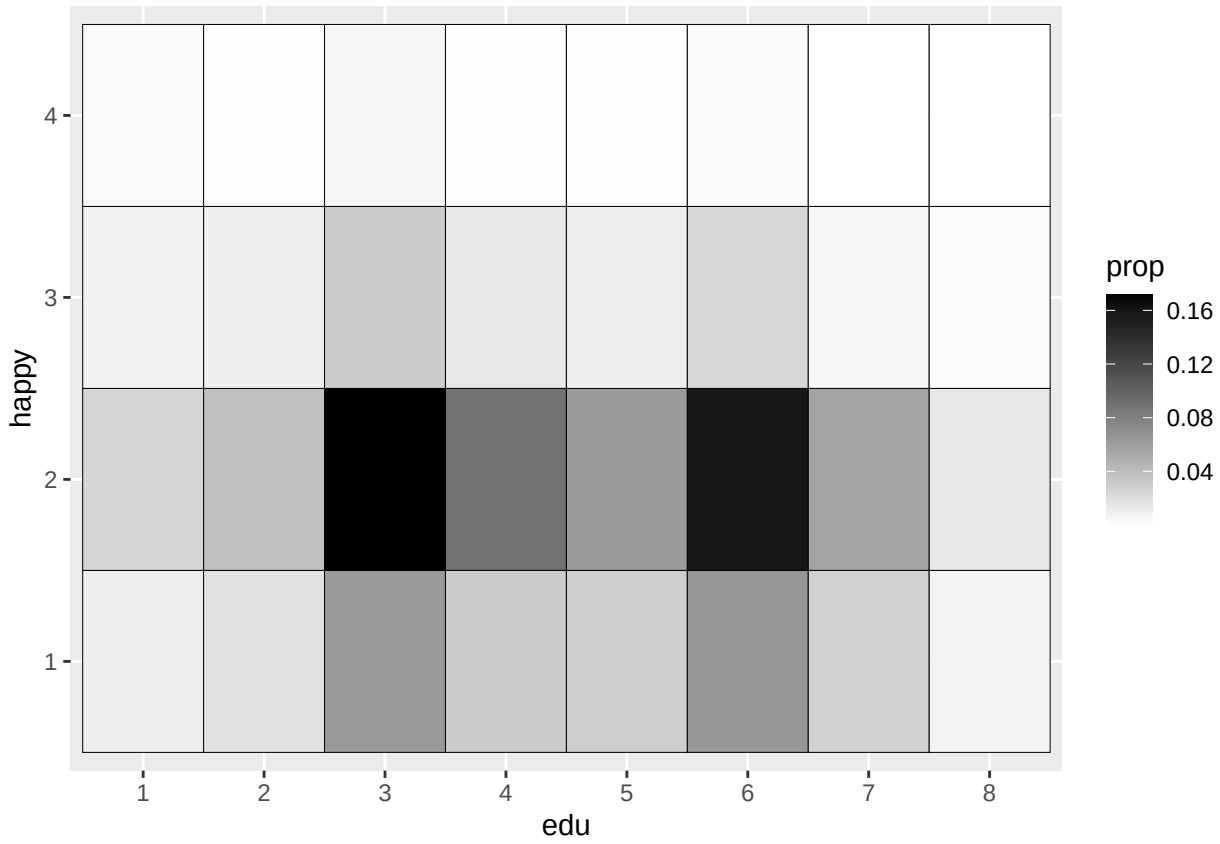
```
dta_autocracy <- dta_new %>% filter(regtype == 1) %>% drop_na(edu, happy)
heat_autocracy <- table(dta_autocracy$edu, dta_autocracy$happy)/nrow(dta_autocracy)
heat_autocracy <- data.frame(heat_autocracy)
colnames(heat_autocracy) <- c("edu", "happy", "prop")

ggplot(heat_autocracy, aes(edu, happy)) +
   geom_tile(aes(fill = prop), colour = "black") +
   scale_fill_gradient(low = "white", high = "black")
```




```
dta_democracy <- dta_new %>% filter(regtype == 5) %>% drop_na(edu, happy)
heat_democracy <- table(dta_democracy$edu, dta_democracy$happy)/nrow(dta_democracy)
heat_democracy <- data.frame(heat_democracy)
colnames(heat_democracy) <- c("edu", "happy", "prop")

ggplot(heat_democracy, aes(edu, happy)) +
   geom_tile(aes(fill = prop), colour = "black") +
   scale_fill_gradient(low = "white", high = "black")
```

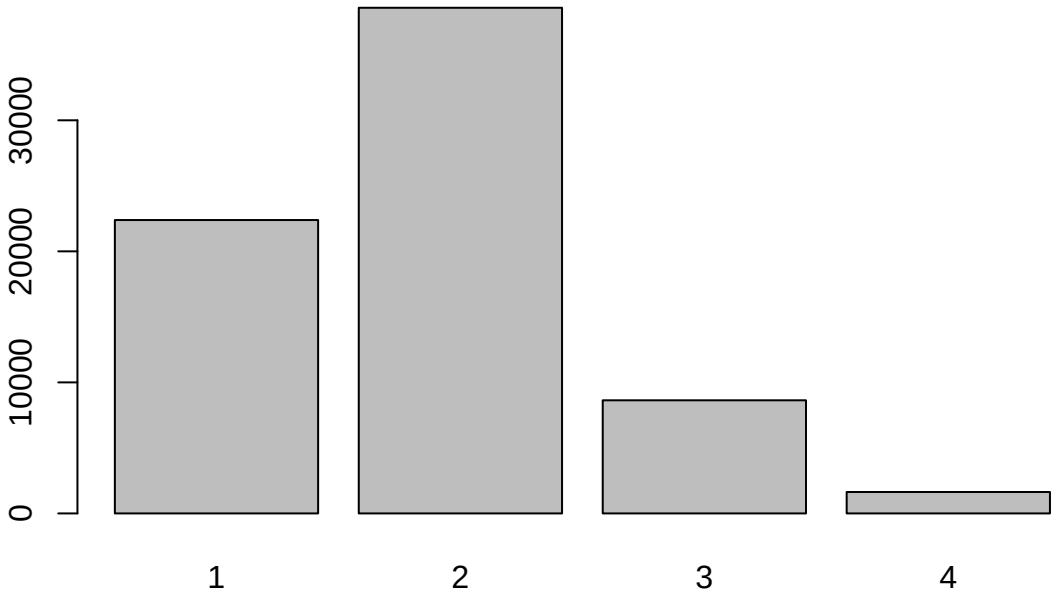


Question: Does the relationship between education and perceived happyion vary by regime type?

5 Analysis

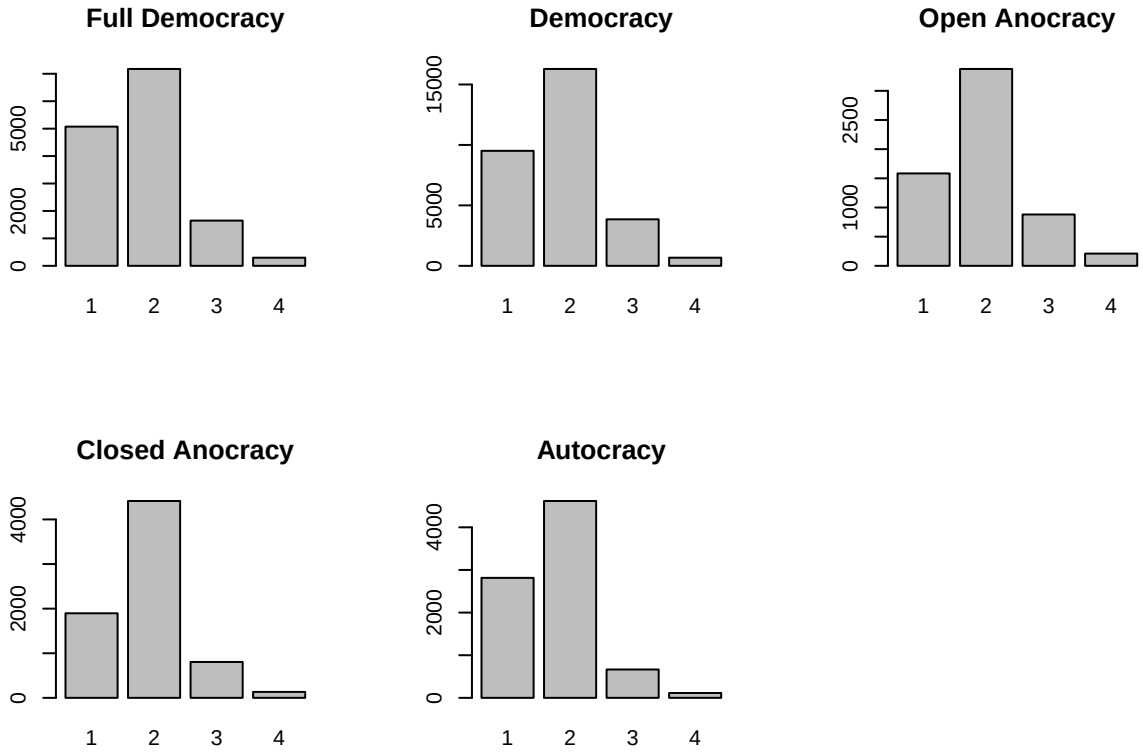
Let's take a look at the outcome variable to check for any potential outliers. Given that both variables are ordinal, we will use **bar charts** rather than histograms.

```
count <- table(dta_new$happy)
barplot(count)
```



Let's check if the distribution varies by political regimes.

```
par(mfrow=c(2,3))
barplot(table(dta_new$happy[dta$regtype == 5]), main="Full Democracy")
barplot(table(dta_new$happy[dta$regtype == 4]), main="Democracy")
barplot(table(dta_new$happy[dta$regtype == 3]), main="Open Anocracy")
barplot(table(dta_new$happy[dta$regtype == 2]), main="Closed Anocracy")
barplot(table(dta_new$happy[dta$regtype == 1]), main="Autocracy")
```



Question: What do you think? Is what you see here consistent with your expectations?

As the starter, let's use our old friend `lm()`.

```r
options(scipen=999)

mod_lm <- lm(happy ~ 1, data=dta_new)
summary(mod_lm)
```

```
##
## Call:
## lm(formula = happy ~ 1, data = dta_new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8528 -0.8528  0.1472  0.1472  2.1472
##
## Coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 1.852788   0.002664   695.6 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.711 on 71243 degrees of freedom
```

```r
mod_lm <- lm(happy ~ edu, data=dta_new)
summary(mod_lm)
```

```
##
## Call:
## lm(formula = happy ~ edu, data = dta_new)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8769 -0.8247  0.1440  0.1753  2.1962
##
## Coefficients:
##              Estimate Std. Error t value             Pr(>|t|)
## (Intercept)  1.887362   0.005949 317.268 < 0.0000000000000002 ***
## edu         -0.010445   0.001440  -7.252    0.000000000000416 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7041 on 66645 degrees of freedom
##   (4597 observations deleted due to missingness)
## Multiple R-squared:  0.0007884,  Adjusted R-squared:  0.0007734
## F-statistic: 52.59 on 1 and 66645 DF,  p-value: 0.0000000000004163
```

There seems to be a **negative** correlation between `edu` and `happy` (huh?).

Now we will attempt several types of multilevel modeling approaches (using `reg_type` to group respondents across different countries), which can depend on several things.

- Should we include any predictors?
- Should we allow the intercept to vary by group?
- Should we allow the slope to vary by group?

  Question: How do you make decisions on these questions?

5.1 No Predictor; Varying Intercept and Fixed Slope

If we only allow the intercept to vary by group without including any predictor, basically we are trying to fit the following models in one go with `lmer()`:

$$\text{happy} = \alpha_i + \epsilon,$$

in which we use i to denote a given regime type (e.g., autocracy, closed anocracy, open anocracy, democracy, full democracy).

Since we have five groups, that means our goal is to find the estimated intercept of each type (in other words, in total we want to have five αs) while keeping the estimated slope the same regardless of the regime type.

Let's fit the model.

```
options(scipen=999)
mod_lmm_1 <- lmer(happy ~ 1 + (1|reg_type), data=dta_new)
summary(mod_lmm_1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: happy ~ 1 + (1 | reg_type)
##    Data: dta_new
##
## REML criterion at convergence: 153161.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.3717 -1.1274  0.1622  0.2835  3.1052
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  reg_type (Intercept) 0.003883 0.06232
##  Residual             0.502358 0.70877
## Number of obs: 71244, groups:  reg_type, 5
##
## Fixed effects:
##             Estimate Std. Error      df t value    Pr(>|t|)
## (Intercept)  1.88304    0.02804 4.00082   67.15 0.000000294 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The regression output does not tell us the exact estimated intercept of each regime type, so we have to get them manually. To do so,

- Step 1: in the regression output, you can find the **average** of all estimated intercepts across political regimes listed under `Fixed effects`. You can use `fixef()` to obtain it. This is also known as the **fixed** part of the estimated intercept of each regime type.
- Step 2: Now we know the average, the question now is how far away each intercept is from the average. This is the **random** part of the estimated intercept of each regime type. We can use the function `ranef()` to call out them out.
- Step 3: The sum of `fixef()` and `ranef()` is the estimated intercept of each group (or regime type).

Let's proceed.

```
fixef(mod_lmm_1) # fixed part of each intercept
```

```
## (Intercept)
##     1.88304
```

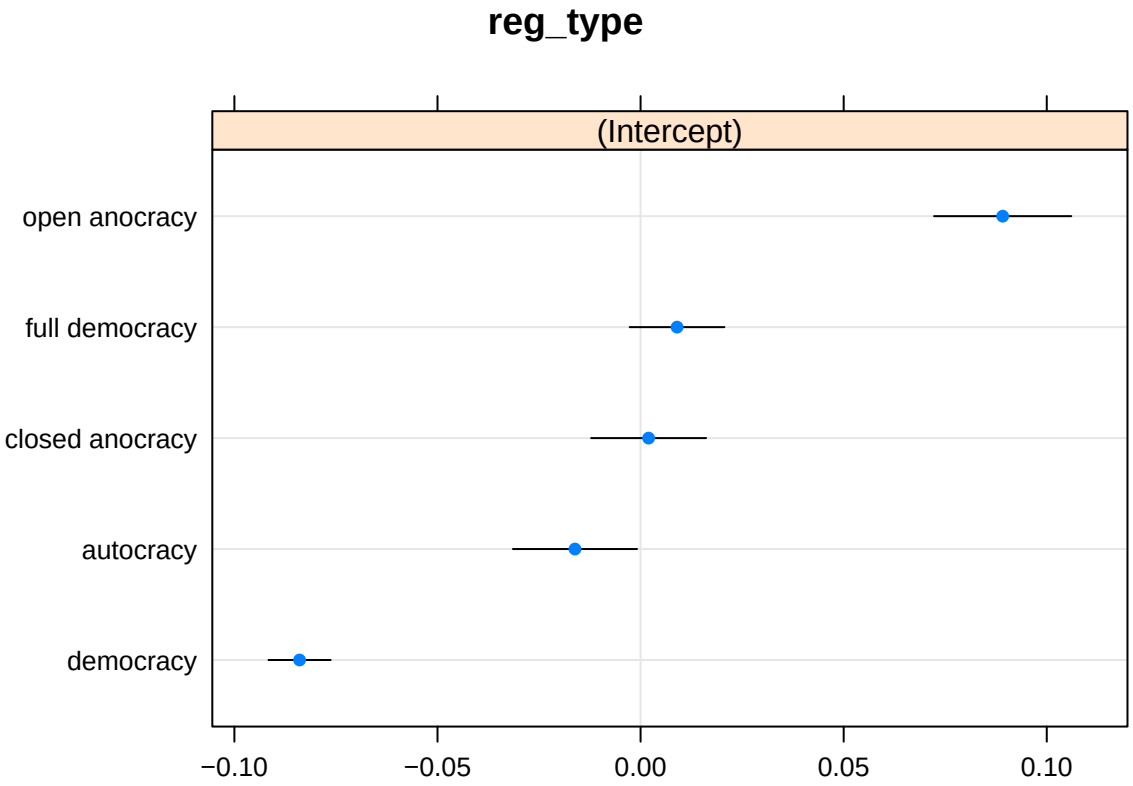
```
ranef(mod_lmm_1) # random part of each intercept
```

```
## $reg_type
##                   (Intercept)
## autocracy       -0.016149079
## closed anocracy  0.001977716
## open anocracy    0.089148460
## democracy       -0.083956189
## full democracy   0.008979092
##
## with conditional variances for "reg_type"
```

You can also plot the random part of each estimated intercept, with the confidence interval included for each of them.

```
dotplot(ranef(mod_lmm_1, condVar = TRUE))
```

```
## $reg_type
```



Alternatively, We can use the `coef()` function again to extract the estimated intercept of each regime type – this is a good way to double check your calculation (or skip the calculation).

```
coef(mod_lmm_1)
```

```
## $reg_type
##                 (Intercept)
## autocracy          1.866890
## closed anocracy    1.885017
## open anocracy      1.972188
## democracy          1.799083
## full democracy     1.892019
```

```
##
## attr(,"class")
## [1] "coef.mer"
```

Question: How do you define and derive the confidence interval of each estimated intercept?

You can repeat the same process but with different modeling approaches (see below). Again, we can use `names()` to open the black box of the regression output.

```
res_lmm_1 <- summary(mod_lmm_1)
names(res_lmm_1)
```

```
##  [1] "methTitle"    "objClass"     "devcomp"      "isLmer"       "useScale"
##  [6] "logLik"       "family"       "link"         "ngrps"        "coefficients"
## [11] "sigma"        "vcov"         "varcor"       "AICtab"       "call"
## [16] "residuals"    "fitMsgs"      "optinfo"
```

Question: See anything familiar?

5.2 With Predictor; Varying Intercept and Fixed Slope

Now let's include the predictor `edu` in our model:

$$\text{happy} = \alpha_i + \beta(\text{edu}) + \epsilon.$$

Again, our objective is to find the estimated intercept corresponding to each regime type.

```
options(scipen=999)
mod_lmm_int <- lmer(happy ~ edu + (1|reg_type), data=dta_new)

fixef(mod_lmm_int) # fixed part of each intercept
ranef(mod_lmm_int) # random part of each intercept
dotplot(ranef(mod_lmm_int, condVar = TRUE))
coef(mod_lmm_int) # sum of fixed and random
```

5.3 With Predictor; Fixed Intercept and Varying Slope

Say now we believe that the intercept for each regime type should be the same, but each regime type will have a different slope. This idea will turn our model into:

$$\text{happy} = \alpha + \beta_i(\text{edu}) + \epsilon.$$

Our objective is to find the estimated slope corresponding to each regime type.

```
options(scipen=999)
mod_lmm_beta <- lmer(happy ~ edu + (0+edu|reg_type), data=dta_new)

fixef(mod_lmm_beta) # fixed part of each slope
ranef(mod_lmm_beta) # random part of each slope
dotplot(ranef(mod_lmm_beta, condVar = TRUE))
coef(mod_lmm_beta) # sum of fixed and random
```

5.4 With Predictor; Varying Intercept and Slope

Finally, if we decide that each regime type should have its own unique intercept and slope, then the model becomes:

$$\text{happy} = \alpha_i + \beta_i(\text{edu}) + \epsilon.$$

Our objective is to find the estimated intercept and slope corresponding to each regime type.

```
options(scipen=999)
mod_lmm_int_beta <- lmer(happy ~ edu + (1+edu|reg_type), data=dta_new)

fixef(mod_lmm_int_beta) # fixed part of each intercept and slope
ranef(mod_lmm_int_beta) # random part of each intercept and slope
dotplot(ranef(mod_lmm_int_beta, condVar = TRUE))
coef(mod_lmm_int_beta) # sum of fixed and random
```

6 Concluding Remarks

Question: Which one do you prefer? And any ideas to justify your choice? (hint: residuals may come to rescue).