Revised Regression

Evan Dragich

Last compiled on 2021-12-26

Data Cleaning

Check out the .Rmd file for full code (omitted here for brevity).

Motivation for a new statistical method

My personal biggest concern with our methods, even before receiving reviewer comments, was the blatant violation of the independence condition for standard linear regression. I **knew** there was an alternative– something that could account for the repeated measures, or “nesting” of `blooms_level * task` within each student.

During my search, I finally found it– a two-way repeated measures ANOVA. This page has good vignettes for sample scenarios, which you can draw analogies to our data structure.

I tried to find out how to implement it, and ran into recent literature eschewing ANOVA in favor of regression. This page also pointed me to the family of linear mixed models, which extend simple linear regression by allowing for random effects, in addtion to fixed.

Random Effects

Random effects control for the repeated measures factor, by essentially allowing factors to vary along `student_id` without using up degrees of freedom to try to make sense of the results. In other words, `student_id` is a random effect in our case because we have to account for each student having a different, random intercept, before examining the global, fixed effects of `task` and `blooms_level`.

The `(1 | year/student_id)` term in these models indicates that we have to account for the random, nuisance variance caused by each student and each year/cohort before examining the effects of `blooms_level`, `task`, and their interaction. Specifically, the `/` nesting operator means that we have a first random intercept to account for variance across years, and after accounting for this variance we introduce a random intercept to further account for that particular student’s variance.

Mass Analysis

Omnibus ANOVA

```
mass <- mass %>%
  mutate(rating = factor(rating, ordered = TRUE))

mass_clmm <- clmm(rating ~ blooms_level * task + year + (1 | student_id), data = mass)

summary(mass_clmm)
```

```
## Cumulative Link Mixed Model fitted with the Laplace approximation
##
## formula: rating ~ blooms_level * task + year + (1 | student_id)
## data:    mass
##
##  link  threshold nobs logLik   AIC     niter       max.grad cond.H
##  logit flexible  2668 -3204.95 6467.90 4687(18753) 1.26e-03 1.2e+03
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  student_id (Intercept) 1.566    1.252
## Number of groups:  student_id 90
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## blooms_levelunderstand          0.55051    0.28832   1.909  0.05622 .
## blooms_levelanalyze            -1.31682    0.28375  -4.641 3.47e-06 ***
## blooms_levelapply              -1.22389    0.28091  -4.357 1.32e-05 ***
## blooms_levelevaluate           -1.87272    0.28485  -6.574 4.88e-11 ***
## blooms_levelcreate             -2.77720    0.29524  -9.406  < 2e-16 ***
## taskicp                        -0.12316    0.28721  -0.429  0.66805
## taskhw                          0.48298    0.29046   1.663  0.09634 .
## taskpbl                        -1.37448    0.24985  -5.501 3.77e-08 ***
## year2020                       -0.51974    0.27589  -1.884  0.05958 .
## blooms_levelunderstand:taskicp -0.11151    0.40643  -0.274  0.78381
## blooms_levelanalyze:taskicp     1.09333    0.39622   2.759  0.00579 **
## blooms_levelapply:taskicp       1.71607    0.39980   4.292 1.77e-05 ***
## blooms_levelevaluate:taskicp    0.99320    0.39883   2.490  0.01276 *
## blooms_levelcreate:taskicp      0.62510    0.40634   1.538  0.12396
## blooms_levelunderstand:taskhw  -0.35728    0.41285  -0.865  0.38682
## blooms_levelanalyze:taskhw      1.05436    0.40420   2.609  0.00909 **
## blooms_levelapply:taskhw        2.00345    0.41634   4.812 1.49e-06 ***
## blooms_levelevaluate:taskhw     0.92013    0.40537   2.270  0.02322 *
## blooms_levelcreate:taskhw       0.70755    0.41599   1.701  0.08897 .
## blooms_levelunderstand:taskpbl  0.01264    0.34873   0.036  0.97108
## blooms_levelanalyze:taskpbl     2.76783    0.35074   7.892 2.99e-15 ***
## blooms_levelapply:taskpbl       2.88040    0.35009   8.228  < 2e-16 ***
## blooms_levelevaluate:taskpbl    3.49730    0.35479   9.858  < 2e-16 ***
## blooms_levelcreate:taskpbl      4.99270    0.37088  13.462  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 1|2  -5.2848     0.3173 -16.657
## 2|3  -3.1013     0.2933 -10.572
## 3|4  -1.3877     0.2870  -4.835
## 4|5   0.3534     0.2856   1.237
## (32 observations deleted due to missingness)
```

```
Anova.clmm(mass_clmm)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: rating
##                    LR Chisq Df Pr(>Chisq)    
## blooms_level         91.50  5    < 2e-16 ***
## task                110.77  3    < 2e-16 ***
## year                  3.46  1    0.06282 .  
## blooms_level:task   428.58 15    < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# null model to test significance of fixed effects
mass_null <- clm(rating ~ 1, data = mass)
anova(mass_clmm, mass_null)
```

```
## Likelihood ratio tests of cumulative link models:
##  
##           formula:                                               link:
## mass_null rating ~ 1                                             logit
## mass_clmm rating ~ blooms_level * task + year + (1 | student_id) logit
##           threshold:
## mass_null flexible  
## mass_clmm flexible  
## 
##           no.par    AIC  logLik LR.stat df Pr(>Chisq)    
## mass_null      4 7562.9 -3777.4                          
## mass_clmm     29 6467.9 -3205.0    1145 25  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# null model to test significance of random effects
mass_null2 <- clm(rating ~ blooms_level * task, data = mass)
anova(mass_clmm, mass_null2)
```

```
## Likelihood ratio tests of cumulative link models:
##  
##            formula:                                               link:
## mass_null2 rating ~ blooms_level * task                           logit
## mass_clmm  rating ~ blooms_level * task + year + (1 | student_id) logit
##            threshold:
## mass_null2 flexible  
## mass_clmm  flexible  
## 
##            no.par    AIC  logLik LR.stat df Pr(>Chisq)    
## mass_null2     27 7142.2 -3544.1                          
## mass_clmm      29 6467.9 -3205.0  678.29  2  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Great! Significant main effects for `task` and `blooms_level`, as well as their interaction. The next step is to break this down into pairwise comparisons. I chose to analyze pairs of `blooms_level` within each `task`; this is most analogous to our former method.

Marginal Means and Contrasts

```r
# split by task, then compares pairs of blooms levels
mass_emm_t <- emmeans(mass_clmm, specs = pairwise ~ blooms_level | task, mode = "mean.class")

mass_task_means <- mass_emm_t$emmeans %>%
  summary(infer = TRUE, null = mean(as.numeric(mass$rating), na.rm = TRUE), level = 0.99)
mass_task_contrasts <-mass_emm_t$contrasts %>%
  summary(infer = TRUE, level = 0.99)

# split by blooms level, then compares pairs of tasks
mass_emm_bl <- emmeans(mass_clmm, specs = pairwise ~ task | blooms_level, mode = "mean.class")

mass_bl_means <- mass_emm_bl$emmeans %>%
  summary(infer = TRUE, null = mean(as.numeric(mass$rating), na.rm = TRUE), level = 0.99)
mass_bl_contrasts <- mass_emm_bl$contrasts %>%
  summary(infer = TRUE, level = 0.99)
```

(Surprisingly, there don't exist any simple packages to transform this output into a neater table–I think we'd have to do it ourselves.)

The first table is the estimated marginal means (EMM), which enhances bare-bones descriptive statistics by accounting for imbalances in data. This is huge, because our mass data has double the `pbl` observations of the other categories. This method also helps with the imbalance from missing data, but that is a trivial concern compared to the double-PBL issue.

The second table output contains the pairwise contrasts between each level for a particular task, with the α = 0.05 p-value, associated 95% confidence interval, and Tukey family-wise adjustment.

My Thoughts

This is messier than our previous output, but displays similar effects which lead to similar interpretations. It isn't nearly as parsimonious as "these 3 coefficients are negative, but PBL is the only positive one!" However, examine the `lec` contrasts and you can see ample comparisons which estimate `Remember` and `Understand` well below the higher levels, with significance. Conversely, these same comparisons have negative estimated effects for `pbl`, indicating a significant difference in the opposite direction. Middle Bloom's Levels are hazier to tease apart, as they were before, but you can see the clear stratification between `lec -> hw/icp -> pbl`, in my opinion.

Kinetics Analysis

Omnibus ANOVA

Here is the same procedure for kinetics. Starting with model fitting and the omnibus ANOVA:

```
kinetics <- kinetics %>%
  mutate(rating = factor(rating, ordered = TRUE))

kinetics_clmm <- clmm(rating ~ blooms_level * task + year + (1 | student_id), data = kinetics)

summary(kinetics_clmm)
```

```
## Cumulative Link Mixed Model fitted with the Laplace approximation
##
## formula: rating ~ blooms_level * task + year + (1 | student_id)
## data:    kinetics
##
##  link  threshold nobs logLik   AIC     niter       max.grad cond.H
##  logit flexible  2141 -2586.52 5231.04 4475(22349) 1.84e-03 8.4e+02
##
## Random effects:
```

```
##  Groups     Name        Variance Std.Dev.
##  student_id (Intercept) 2.409    1.552
## Number of groups:  student_id 90
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## blooms_levelunderstand          0.0259     0.2866   0.090 0.927999
## blooms_levelanalyze            -1.0350     0.2799  -3.698 0.000217 ***
## blooms_levelapply              -1.0260     0.2782  -3.688 0.000226 ***
## blooms_levelevaluate           -1.6599     0.2812  -5.902 3.59e-09 ***
## blooms_levelcreate             -2.3911     0.2881  -8.300  < 2e-16 ***
## taskicp                         0.1406     0.2818   0.499 0.617743
## taskhw                          0.5414     0.2938   1.843 0.065355 .
## taskpbl                        -0.5729     0.2943  -1.946 0.051609 .
## year2020                       -0.7565     0.3397  -2.227 0.025940 *
## blooms_levelunderstand:taskicp  0.2168     0.4000   0.542 0.587930
## blooms_levelanalyze:taskicp     0.5714     0.3913   1.460 0.144277
## blooms_levelapply:taskicp       1.0087     0.3931   2.566 0.010281 *
## blooms_levelevaluate:taskicp    0.4311     0.3907   1.103 0.269816
## blooms_levelcreate:taskicp      0.2486     0.3970   0.626 0.531183
## blooms_levelunderstand:taskhw  -0.0702     0.4124  -0.170 0.864829
## blooms_levelanalyze:taskhw      0.7228     0.4046   1.787 0.073981 .
## blooms_levelapply:taskhw        1.4440     0.4086   3.534 0.000410 ***
## blooms_levelevaluate:taskhw     0.7946     0.4048   1.963 0.049668 *
## blooms_levelcreate:taskhw       0.4822     0.4093   1.178 0.238674
## blooms_levelunderstand:taskpbl  0.0342     0.4137   0.083 0.934122
## blooms_levelanalyze:taskpbl     2.0617     0.4181   4.931 8.19e-07 ***
## blooms_levelapply:taskpbl       2.1544     0.4189   5.143 2.70e-07 ***
## blooms_levelevaluate:taskpbl    2.4275     0.4186   5.799 6.69e-09 ***
## blooms_levelcreate:taskpbl      3.4622     0.4291   8.068 7.15e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 1|2  -5.0780     0.3468 -14.644
## 2|3  -3.1506     0.3276  -9.619
## 3|4  -1.3550     0.3210  -4.222
## 4|5   0.5846     0.3199   1.827
## (19 observations deleted due to missingness)
```

```
Anova.clmm(kinetics_clmm)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: rating
##                   LR Chisq Df Pr(>Chisq)
## blooms_level       164.799  5    < 2e-16 ***
## task               113.278  3    < 2e-16 ***
## year                 4.824  1    0.02806 *
## blooms_level:task  141.597 15    < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# null model to test significance of fixed effects
kinetics_null <- clm(rating ~ 1, data = kinetics)
anova(kinetics_clmm, kinetics_null)
```

```
## Likelihood ratio tests of cumulative link models:
##
##               formula:                                               link:
## kinetics_null rating ~ 1                                             logit
## kinetics_clmm rating ~ blooms_level * task + year + (1 | student_id) logit
##               threshold:
## kinetics_null flexible
## kinetics_clmm flexible
##
##               no.par    AIC  logLik LR.stat df Pr(>Chisq)
## kinetics_null      4 6261.2 -3126.6
## kinetics_clmm     29 5231.0 -2586.5  1080.2 25  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# null model to test significance of random effects
kinetics_null2 <- clm(rating ~ blooms_level * task + year, data = kinetics)
anova(kinetics_clmm, kinetics_null2)
```

```
## Likelihood ratio tests of cumulative link models:
##
##                formula:                                               link:
## kinetics_null2 rating ~ blooms_level * task + year                    logit
## kinetics_clmm  rating ~ blooms_level * task + year + (1 | student_id) logit
##                threshold:
## kinetics_null2 flexible
## kinetics_clmm  flexible
##
##                no.par    AIC  logLik LR.stat df Pr(>Chisq)
## kinetics_null2     28 5980.2 -2962.1
## kinetics_clmm      29 5231.0 -2586.5  751.18  1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Estimated Marginal Means and Contrasts

```
kinetics_emm_t <- emmeans(kinetics_clmm, specs = pairwise ~ blooms_level | task, mode = "mean.class")

kinetics_task_means <- kinetics_emm_t$emmeans %>%
  summary(infer = TRUE, null = mean(as.numeric(kinetics$rating), na.rm = TRUE), level = 0.99)
kinetics_task_contrasts <-kinetics_emm_t$contrasts %>%
  summary(infer = TRUE, level = 0.99)


kinetics_emm_bl <- emmeans(kinetics_clmm, specs = pairwise ~ task | blooms_level, mode = "mean.class")

kinetics_bl_means <- kinetics_emm_bl$emmeans %>%
  summary(infer = TRUE, null = mean(as.numeric(kinetics$rating), na.rm = TRUE), level = 0.99)
kinetics_bl_contrasts <- kinetics_emm_bl$contrasts %>%
  summary(infer = TRUE, level = 0.99)
```

My thoughts

Very analogous results again. `pbl` in particular **definitely** doesn't look as pretty as our original output.

But, I think we can interpret each of these significant pairwise comparisons in a much more statistically sound way, which will please reviewers.

Write results to xlsx

```
write_xlsx(list("Mass by Bloom's -  Mean Classes" = mass_bl_means,
                "Mass by Bloom's -  Contrasts" = mass_bl_contrasts,
                "Mass by Task -  Mean Classes" = mass_task_means,
                "Mass by Task -  Contrasts" = mass_task_contrasts,
                "Kinetics by Bloom's -  Mean Classes" = kinetics_bl_means,
                "Kinetics by Bloom's -  Contrasts" = kinetics_bl_contrasts,
                "Kinetics by Task -  Mean Classes" = kinetics_task_means,
                "Kinetics by Task -  Contrasts" = kinetics_task_contrasts),
            "clmm_anova_output.xlsx")
```

```
# information about this R session:
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] emmeans_1.7.1-1      RVAideMemoire_0.9-81 ordinal_2019.12-10
##  [4] forcats_0.5.1        stringr_1.4.0        dplyr_1.0.7
##  [7] purrr_0.3.4          readr_2.1.1          tidyr_1.1.4
## [10] tibble_3.1.6         ggplot2_3.3.5        tidyverse_1.3.1
## [13] writexl_1.4.0        readxl_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] httr_1.4.2          jsonlite_1.7.2      splines_4.1.2
##  [4] here_1.0.1          modelr_0.1.8        ucminf_1.1-4
##  [7] assertthat_0.2.1    cellranger_1.1.0    yaml_2.2.1
## [10] numDeriv_2016.8-1.1 pillar_1.6.4        backports_1.4.0
## [13] lattice_0.20-45     glue_1.5.1          digest_0.6.29
## [16] rvest_1.0.2         colorspace_2.0-2    sandwich_3.0-1
## [19] htmltools_0.5.2     Matrix_1.3-4        pkgconfig_2.0.3
## [22] broom_0.7.10        haven_2.4.3         xtable_1.8-4
## [25] mvtnorm_1.1-3       scales_1.1.1        tzdb_0.2.0
## [28] generics_0.1.1      ellipsis_0.3.2      TH.data_1.1-0
## [31] withr_2.4.3         cli_3.1.0           survival_3.2-13
## [34] magrittr_2.0.1      crayon_1.4.2        estimability_1.3
## [37] evaluate_0.14       fs_1.5.2            fansi_0.5.0
## [40] MASS_7.3-54         xml2_1.3.3          tools_4.1.2
## [43] hms_1.1.1           lifecycle_1.0.1     multcomp_1.4-17
## [46] munsell_0.5.0       reprex_2.0.1        compiler_4.1.2
## [49] rlang_0.4.12        grid_4.1.2          rstudioapi_0.13
## [52] rmarkdown_2.11      gtable_0.3.0        codetools_0.2-18
## [55] DBI_1.1.1           R6_2.5.1            zoo_1.8-9
## [58] lubridate_1.8.0     knitr_1.36          fastmap_1.1.0
## [61] utf8_1.2.2          rprojroot_2.0.2     stringi_1.7.6
## [64] Rcpp_1.0.7          vctrs_0.3.8         dbplyr_2.1.1
## [67] tidyselect_1.1.1    xfun_0.28
```