Revised Regression

Evan Dragich

Last compiled on 2021-12-18

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)
```

```r
# looks like year wasn't signfiicant
mass_clmm2 <- clmm(rating ~ blooms_level * task + (1 | student_id), data = mass)

summary(mass_clmm2)
```

```
## Cumulative Link Mixed Model fitted with the Laplace approximation
##
## formula: rating ~ blooms_level * task + (1 | student_id)
## data:    mass
##
##  link  threshold nobs logLik   AIC     niter       max.grad cond.H
##  logit flexible  2668 -3206.68 6469.36 4735(23323) 4.46e-03 1.2e+03
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  student_id (Intercept) 1.648    1.284
## Number of groups:  student_id 90
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## blooms_levelunderstand           0.55000    0.28859   1.906  0.05667 .
## blooms_levelanalyze             -1.31694    0.28404  -4.637 3.54e-06 ***
## blooms_levelapply               -1.22481    0.28121  -4.356 1.33e-05 ***
## blooms_levelevaluate            -1.87395    0.28511  -6.573 4.94e-11 ***
## blooms_levelcreate              -2.77879    0.29549  -9.404  < 2e-16 ***
## taskicp                         -0.12354    0.28742  -0.430  0.66732
## taskhw                           0.48263    0.29068   1.660  0.09685 .
## taskpbl                         -1.37517    0.25014  -5.498 3.85e-08 ***
## blooms_levelunderstand:taskicp -0.11163    0.40668  -0.274  0.78370
## blooms_levelanalyze:taskicp     1.09323    0.39652   2.757  0.00583 **
## blooms_levelapply:taskicp       1.71726    0.40004   4.293 1.77e-05 ***
## blooms_levelevaluate:taskicp    0.99309    0.39908   2.488  0.01283 *
## blooms_levelcreate:taskicp      0.62585    0.40651   1.540  0.12366
## blooms_levelunderstand:taskhw  -0.35689    0.41303  -0.864  0.38754
## blooms_levelanalyze:taskhw      1.05441    0.40442   2.607  0.00913 **
## blooms_levelapply:taskhw        2.00459    0.41653   4.813 1.49e-06 ***
## blooms_levelevaluate:taskhw     0.92089    0.40559   2.270  0.02318 *
## blooms_levelcreate:taskhw       0.70857    0.41621   1.702  0.08868 .
## blooms_levelunderstand:taskpbl  0.01386    0.34910   0.040  0.96833
## blooms_levelanalyze:taskpbl     2.76817    0.35108   7.885 3.15e-15 ***
## blooms_levelapply:taskpbl       2.88174    0.35044   8.223  < 2e-16 ***
## blooms_levelevaluate:taskpbl    3.49910    0.35513   9.853  < 2e-16 ***
## blooms_levelcreate:taskpbl      4.99451    0.37120  13.455  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 1|2  -5.0098     0.2816 -17.789
## 2|3  -2.8261     0.2550 -11.083
## 3|4  -1.1127     0.2484  -4.479
## 4|5   0.6281     0.2476   2.537
## (32 observations deleted due to missingness)
```

```
Anova.clmm(mass_clmm2)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: rating
##                   LR Chisq Df Pr(>Chisq)
```

```
## blooms_level         91.83  5  < 2.2e-16 ***
## task                 110.84  3  < 2.2e-16 ***
## blooms_level:task    428.54 15  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# null model to test significance of fixed effects
mass_null <- clm(rating ~ 1, data = mass)
anova(mass_clmm2, mass_null)
```

```
## Likelihood ratio tests of cumulative link models:
##
##            formula:                                        link: threshold:
## mass_null  rating ~ 1                                      logit flexible
## mass_clmm2 rating ~ blooms_level * task + (1 | student_id) logit flexible
##
##            no.par    AIC  logLik LR.stat df Pr(>Chisq)
## mass_null       4 7562.9 -3777.4
## mass_clmm2     28 6469.4 -3206.7  1141.5 24  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# null model to test significance of random effects
mass_null2 <- clm(rating ~ blooms_level * task, data = mass)
anova(mass_clmm2, mass_null2)
```

```
## Likelihood ratio tests of cumulative link models:
##
##            formula:                                        link: threshold:
## mass_null2 rating ~ blooms_level * task                    logit flexible
## mass_clmm2 rating ~ blooms_level * task + (1 | student_id) logit flexible
##
##            no.par    AIC  logLik LR.stat df Pr(>Chisq)
## mass_null2     27 7142.2 -3544.1
## mass_clmm2     28 6469.4 -3206.7  674.83  1  < 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

```
mass_emm <- emmeans(mass_clmm2, specs = pairwise ~ blooms_level | task, mode = "mean.class")

mass_emm$emmeans %>%
  summary(infer = TRUE, null = mean(as.numeric(mass$rating), na.rm = TRUE), level = 0.99)
```

```
## task = lec:
##  blooms_level mean.class     SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember           4.04 0.1164 Inf      3.74      4.34 3.79   2.169  0.0301
##  understand         4.28 0.1025 Inf      4.02      4.55 3.79   4.865  <.0001
##  analyze            3.37 0.1255 Inf      3.05      3.69 3.79  -3.321  0.0009
##  apply              3.42 0.1234 Inf      3.10      3.74 3.79  -2.982  0.0029
##  evaluate           3.07 0.1252 Inf      2.75      3.40 3.79  -5.683  <.0001
```

```
##  create              2.61 0.1236 Inf      2.29      2.92 3.79  -9.546  <.0001
##
## task = icp:
##  blooms_level mean.class     SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember            3.98 0.1170 Inf      3.68      4.28 3.79   1.654  0.0981
##  understand          4.18 0.1092 Inf      3.90      4.46 3.79   3.635  0.0003
##  analyze             3.87 0.1162 Inf      3.57      4.17 3.79   0.723  0.4694
##  apply               4.21 0.1063 Inf      3.93      4.48 3.79   3.959  0.0001
##  evaluate            3.53 0.1248 Inf      3.21      3.85 3.79  -2.021  0.0432
##  create              2.86 0.1263 Inf      2.54      3.19 3.79  -7.303  <.0001
##
## task = hw:
##  blooms_level mean.class     SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember            4.26 0.1053 Inf      3.98      4.53 3.79   4.463  <.0001
##  understand          4.34 0.1020 Inf      4.07      4.60 3.79   5.394  <.0001
##  analyze             4.14 0.1106 Inf      3.86      4.42 3.79   3.203  0.0014
##  apply               4.55 0.0854 Inf      4.33      4.77 3.79   8.955  <.0001
##  evaluate            3.81 0.1236 Inf      3.49      4.13 3.79   0.182  0.8556
##  create              3.23 0.1350 Inf      2.88      3.57 3.79  -4.148  <.0001
##
## task = pbl:
##  blooms_level mean.class     SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember            3.34 0.1034 Inf      3.07      3.60 3.79  -4.327  <.0001
##  understand          3.63 0.1002 Inf      3.38      3.89 3.79  -1.514  0.1301
##  analyze             4.07 0.0927 Inf      3.83      4.31 3.79   3.107  0.0019
##  apply               4.17 0.0901 Inf      3.94      4.40 3.79   4.243  <.0001
##  evaluate            4.15 0.0911 Inf      3.92      4.39 3.79   4.039  0.0001
##  create              4.40 0.0806 Inf      4.19      4.61 3.79   7.633  <.0001
##
## Confidence level used: 0.99
```

```r
mass_emm$contrasts %>%
  summary(infer = TRUE, level = 0.99)
```

```
## task = lec:
##  contrast                estimate     SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand  -0.2462 0.1292 Inf  -0.68062    0.1883  -1.906  0.3983
##  remember - analyze      0.6692 0.1417 Inf   0.19259    1.1457   4.723  <.0001
##  remember - apply        0.6205 0.1400 Inf   0.14965    1.0913   4.433  0.0001
##  remember - evaluate     0.9640 0.1415 Inf   0.48790    1.4402   6.811  <.0001
##  remember - create       1.4322 0.1418 Inf   0.95531    1.9090  10.102  <.0001
##  understand - analyze    0.9153 0.1345 Inf   0.46303    1.3677   6.807  <.0001
##  understand - apply      0.8667 0.1326 Inf   0.42067    1.3127   6.536  <.0001
##  understand - evaluate   1.2102 0.1343 Inf   0.75832    1.6621   9.008  <.0001
##  understand - create     1.6783 0.1344 Inf   1.22640    2.1303  12.491  <.0001
##  analyze - apply        -0.0487 0.1438 Inf  -0.53252    0.4352  -0.338  0.9994
##  analyze - evaluate      0.2949 0.1453 Inf  -0.19372    0.7835   2.030  0.3251
##  analyze - create        0.7630 0.1457 Inf   0.27296    1.2531   5.237  <.0001
##  apply - evaluate        0.3435 0.1437 Inf  -0.13980    0.8269   2.391  0.1593
##  apply - create          0.8117 0.1441 Inf   0.32687    1.2965   5.632  <.0001
##  evaluate - create       0.4681 0.1454 Inf  -0.02091    0.9572   3.220  0.0162
##
## task = icp:
##  contrast                estimate     SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand  -0.2035 0.1329 Inf  -0.65037    0.2434  -1.531  0.6439
```

```
##  remember - analyze      0.1094 0.1354 Inf  -0.34613    0.5650   0.808  0.9662
##  remember - apply       -0.2274 0.1309 Inf  -0.66770    0.2130  -1.737  0.5072
##  remember - evaluate     0.4457 0.1413 Inf  -0.02966    0.9210   3.154  0.0200
##  remember - create       1.1162 0.1429 Inf   0.63537    1.5969   7.809  <.0001
##  understand - analyze    0.3129 0.1313 Inf  -0.12889    0.7547   2.382  0.1623
##  understand - apply     -0.0239 0.1263 Inf  -0.44877    0.4010  -0.189  1.0000
##  understand - evaluate   0.6491 0.1375 Inf   0.18667    1.1116   4.722  <.0001
##  understand - create     1.3196 0.1392 Inf   0.85134    1.7879   9.479  <.0001
##  analyze - apply        -0.3368 0.1293 Inf  -0.77172    0.0981  -2.605  0.0960
##  analyze - evaluate      0.3362 0.1395 Inf  -0.13310    0.8056   2.410  0.1527
##  analyze - create        1.0067 0.1413 Inf   0.53141    1.4820   7.124  <.0001
##  apply - evaluate        0.6730 0.1356 Inf   0.21697    1.1291   4.964  <.0001
##  apply - create          1.3435 0.1373 Inf   0.88152    1.8055   9.782  <.0001
##  evaluate - create       0.6705 0.1466 Inf   0.17739    1.1636   4.574  0.0001
##
## task = hw:
##  contrast              estimate     SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand  -0.0803 0.1228 Inf  -0.49343    0.3328  -0.654  0.9868
##  remember - analyze      0.1157 0.1270 Inf  -0.31152    0.5429   0.911  0.9439
##  remember - apply       -0.2943 0.1151 Inf  -0.68160    0.0930  -2.556  0.1083
##  remember - evaluate     0.4476 0.1357 Inf  -0.00892    0.9041   3.298  0.0125
##  remember - create       1.0302 0.1448 Inf   0.54297    1.5174   7.113  <.0001
##  understand - analyze    0.1960 0.1258 Inf  -0.22729    0.6193   1.558  0.6267
##  understand - apply     -0.2140 0.1133 Inf  -0.59505    0.1670  -1.889  0.4088
##  understand - evaluate   0.5279 0.1348 Inf   0.07455    0.9813   3.917  0.0013
##  understand - create     1.1105 0.1440 Inf   0.62598    1.5950   7.710  <.0001
##  analyze - apply        -0.4100 0.1187 Inf  -0.80918   -0.0108  -3.455  0.0073
##  analyze - evaluate      0.3319 0.1381 Inf  -0.13259    0.7964   2.404  0.1549
##  analyze - create        0.9145 0.1470 Inf   0.42007    1.4089   6.222  <.0001
##  apply - evaluate        0.7419 0.1287 Inf   0.30893    1.1749   5.764  <.0001
##  apply - create          1.3245 0.1386 Inf   0.85813    1.7909   9.553  <.0001
##  evaluate - create       0.5826 0.1538 Inf   0.06520    1.1000   3.788  0.0021
##
## task = pbl:
##  contrast              estimate     SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand  -0.2959 0.1029 Inf  -0.64216    0.0503  -2.875  0.0466
##  remember - analyze     -0.7357 0.1009 Inf  -1.07496   -0.3964  -7.294  <.0001
##  remember - apply       -0.8297 0.1002 Inf  -1.16681   -0.4927  -8.280  <.0001
##  remember - evaluate    -0.8155 0.1009 Inf  -1.15474   -0.4762  -8.085  <.0001
##  remember - create      -1.0629 0.0985 Inf  -1.39413   -0.7316 -10.793  <.0001
##  understand - analyze   -0.4398 0.0985 Inf  -0.77106   -0.1085  -4.465  0.0001
##  understand - apply     -0.5338 0.0978 Inf  -0.86280   -0.2048  -5.458  <.0001
##  understand - evaluate  -0.5195 0.0984 Inf  -0.85056   -0.1885  -5.279  <.0001
##  understand - create    -0.7670 0.0959 Inf  -1.08951   -0.4444  -7.998  <.0001
##  analyze - apply        -0.0941 0.0946 Inf  -0.41219    0.2241  -0.995  0.9199
##  analyze - evaluate     -0.0798 0.0952 Inf  -0.40005    0.2405  -0.838  0.9605
##  analyze - create       -0.3272 0.0920 Inf  -0.63655   -0.0178  -3.558  0.0050
##  apply - evaluate        0.0143 0.0943 Inf  -0.30303    0.3316   0.151  1.0000
##  apply - create         -0.2331 0.0907 Inf  -0.53837    0.0721  -2.569  0.1049
##  evaluate - create      -0.2474 0.0915 Inf  -0.55506    0.0602  -2.705  0.0741
##
## Confidence level used: 0.99
## Conf-level adjustment: tukey method for comparing a family of 6 estimates
## P value adjustment: tukey method for comparing a family of 6 estimates
```

(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 $\alpha = 0.05$ 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
```

```r
# 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

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

kinetics_emm$emmeans %>%
  summary(infer = TRUE, null = mean(as.numeric(kinetics$rating), na.rm = TRUE), level = 0.99)
```

```
## task = lec:
##  blooms_level mean.class    SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember           3.93 0.120 Inf      3.62      4.24 3.68   2.111  0.0348
##  understand         3.94 0.120 Inf      3.63      4.25 3.68   2.221  0.0264
##  analyze            3.43 0.127 Inf      3.10      3.76 3.68  -1.953  0.0509
##  apply              3.43 0.126 Inf      3.11      3.76 3.68  -1.932  0.0533
##  evaluate           3.11 0.130 Inf      2.78      3.44 3.68  -4.373  <.0001
##  create             2.73 0.132 Inf      2.40      3.07 3.68  -7.162  <.0001
##
## task = icp:
##  blooms_level mean.class    SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember           3.99 0.115 Inf      3.70      4.29 3.68   2.756  0.0058
##  understand         4.10 0.112 Inf      3.81      4.39 3.68   3.793  0.0001
##  analyze            3.78 0.121 Inf      3.47      4.09 3.68   0.846  0.3978
##  apply              3.99 0.116 Inf      3.69      4.29 3.68   2.667  0.0077
##  evaluate           3.40 0.127 Inf      3.07      3.73 3.68  -2.160  0.0307
##  create             2.93 0.132 Inf      2.59      3.27 3.68  -5.614  <.0001
##
## task = hw:
##  blooms_level mean.class    SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember           4.17 0.113 Inf      3.88      4.46 3.68   4.358  <.0001
##  understand         4.15 0.113 Inf      3.86      4.44 3.68   4.212  <.0001
##  analyze            4.03 0.116 Inf      3.74      4.33 3.68   3.084  0.0020
##  apply              4.34 0.102 Inf      4.08      4.60 3.68   6.483  <.0001
##  evaluate           3.78 0.124 Inf      3.46      4.10 3.68   0.818  0.4131
```

```
##  create             3.26 0.134 Inf      2.91      3.60 3.68  -3.113  0.0019
##
## task = pbl:
##  blooms_level mean.class    SE  df asymp.LCL asymp.UCL null z.ratio p.value
##  remember           3.66 0.131 Inf      3.32      4.00 3.68  -0.141  0.8877
##  understand         3.69 0.130 Inf      3.35      4.02 3.68   0.083  0.9340
##  analyze            4.13 0.119 Inf      3.83      4.44 3.68   3.823  0.0001
##  apply              4.18 0.118 Inf      3.87      4.48 3.68   4.232  <.0001
##  evaluate           4.02 0.124 Inf      3.70      4.34 3.68   2.754  0.0059
##  create             4.15 0.121 Inf      3.84      4.46 3.68   3.930  0.0001
##
## Results are averaged over the levels of: year
## Confidence level used: 0.99
```

```
kinetics_emm$contrasts %>%
  summary(infer = TRUE, level = 0.99)
```

```
## task = lec:
##  contrast              estimate    SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand -0.01191 0.132 Inf  -0.45522    0.4314  -0.090  1.0000
##  remember - analyze     0.50229 0.135 Inf   0.04946    0.9551   3.731  0.0026
##  remember - apply       0.49774 0.134 Inf   0.04792    0.9476   3.722  0.0027
##  remember - evaluate    0.82099 0.136 Inf   0.36284    1.2791   6.028  <.0001
##  remember - create      1.19667 0.138 Inf   0.73167    1.6617   8.657  <.0001
##  understand - analyze   0.51420 0.134 Inf   0.06230    0.9661   3.828  0.0018
##  understand - apply     0.50965 0.133 Inf   0.06093    0.9584   3.821  0.0019
##  understand - evaluate  0.83290 0.136 Inf   0.37587    1.2899   6.130  <.0001
##  understand - create    1.20858 0.138 Inf   0.74427    1.6729   8.756  <.0001
##  analyze - apply       -0.00455 0.136 Inf  -0.46040    0.4513  -0.034  1.0000
##  analyze - evaluate     0.31870 0.138 Inf  -0.14496    0.7824   2.312  0.1890
##  analyze - create       0.69438 0.140 Inf   0.22369    1.1651   4.962  <.0001
##  apply - evaluate       0.32325 0.137 Inf  -0.13758    0.7841   2.359  0.1707
##  apply - create         0.69893 0.139 Inf   0.23094    1.1669   5.024  <.0001
##  evaluate - create      0.37568 0.141 Inf  -0.09906    0.8504   2.662  0.0830
##
## task = icp:
##  contrast              estimate    SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand -0.10759 0.124 Inf  -0.52355    0.3084  -0.870  0.9537
##  remember - analyze     0.21585 0.128 Inf  -0.21331    0.6450   1.692  0.5372
##  remember - apply       0.00785 0.126 Inf  -0.41479    0.4305   0.062  1.0000
##  remember - evaluate    0.59326 0.131 Inf   0.15158    1.0349   4.518  0.0001
##  remember - create      1.06107 0.135 Inf   0.60609    1.5161   7.845  <.0001
##  understand - analyze   0.32344 0.126 Inf  -0.10187    0.7487   2.558  0.1078
##  understand - apply     0.11544 0.124 Inf  -0.30299    0.5339   0.928  0.9394
##  understand - evaluate  0.70085 0.130 Inf   0.26238    1.1393   5.377  <.0001
##  understand - create    1.16866 0.134 Inf   0.71673    1.6206   8.698  <.0001
##  analyze - apply       -0.20800 0.128 Inf  -0.63915    0.2232  -1.623  0.5834
##  analyze - evaluate     0.37741 0.133 Inf  -0.07144    0.8263   2.828  0.0531
##  analyze - create       0.84523 0.137 Inf   0.38355    1.3069   6.158  <.0001
##  apply - evaluate       0.58541 0.132 Inf   0.14153    1.0293   4.436  0.0001
##  apply - create         1.05322 0.136 Inf   0.59611    1.5103   7.750  <.0001
##  evaluate - create      0.46781 0.140 Inf  -0.00388    0.9395   3.336  0.0110
##
## task = hw:
##  contrast              estimate    SE  df asymp.LCL asymp.UCL z.ratio p.value
```

```
##  remember - understand  0.01877 0.126 Inf  -0.40379    0.4413   0.149  1.0000
##  remember - analyze     0.13555 0.127 Inf  -0.29119    0.5623   1.068  0.8940
##  remember - apply      -0.16835 0.120 Inf  -0.57331    0.2366  -1.398  0.7281
##  remember - evaluate    0.39158 0.132 Inf  -0.05219    0.8353   2.968  0.0355
##  remember - create      0.91055 0.138 Inf   0.44483    1.3763   6.577  <.0001
##  understand - analyze   0.11678 0.126 Inf  -0.30699    0.5406   0.927  0.9397
##  understand - apply    -0.18712 0.119 Inf  -0.58903    0.2148  -1.566  0.6211
##  understand - evaluate  0.37281 0.131 Inf  -0.06819    0.8138   2.844  0.0509
##  understand - create    0.89178 0.138 Inf   0.42859    1.3550   6.476  <.0001
##  analyze - apply       -0.30390 0.121 Inf  -0.71074    0.1029  -2.513  0.1203
##  analyze - evaluate     0.25602 0.132 Inf  -0.18769    0.6997   1.941  0.3771
##  analyze - create       0.77500 0.138 Inf   0.30969    1.2403   5.602  <.0001
##  apply - evaluate       0.55992 0.127 Inf   0.13402    0.9858   4.422  0.0001
##  apply - create         1.07889 0.134 Inf   0.62924    1.5285   8.071  <.0001
##  evaluate - create      0.51897 0.142 Inf   0.04048    0.9975   3.648  0.0036
##
## task = pbl:
##  contrast              estimate    SE  df asymp.LCL asymp.UCL z.ratio p.value
##  remember - understand -0.02932 0.145 Inf  -0.51868    0.4601  -0.202  1.0000
##  remember - analyze    -0.47474 0.142 Inf  -0.95229    0.0028  -3.344  0.0107
##  remember - apply      -0.51796 0.142 Inf  -0.99432   -0.0416  -3.658  0.0035
##  remember - evaluate   -0.36102 0.144 Inf  -0.84679    0.1248  -2.500  0.1240
##  remember - create     -0.49371 0.144 Inf  -0.97667   -0.0107  -3.439  0.0077
##  understand - analyze  -0.44543 0.141 Inf  -0.91814    0.0273  -3.170  0.0191
##  understand - apply    -0.48865 0.140 Inf  -0.96015   -0.0171  -3.486  0.0065
##  understand - evaluate -0.33170 0.143 Inf  -0.81273    0.1493  -2.320  0.1861
##  understand - create   -0.46439 0.142 Inf  -0.94260    0.0138  -3.267  0.0139
##  analyze - apply       -0.04322 0.135 Inf  -0.49862    0.4122  -0.319  0.9996
##  analyze - evaluate     0.11372 0.139 Inf  -0.35255    0.5800   0.820  0.9639
##  analyze - create      -0.01897 0.138 Inf  -0.48161    0.4437  -0.138  1.0000
##  apply - evaluate       0.15695 0.138 Inf  -0.30833    0.6222   1.135  0.8670
##  apply - create         0.02425 0.137 Inf  -0.43695    0.4855   0.177  1.0000
##  evaluate - create     -0.13269 0.140 Inf  -0.60466    0.3393  -0.946  0.9346
##
## Results are averaged over the levels of: year
## Confidence level used: 0.99
## Conf-level adjustment: tukey method for comparing a family of 6 estimates
## P value adjustment: tukey method for comparing a family of 6 estimates
```

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.