Item analysis

1. Data cleaning: by facet. No reverse coding or anything when nothing is sure.
2. Obtain in SPSS response frequencies and the histograms of the frequencies in order to detect any abnormal distribution. Normal distribution should have low frequencies for extreme response categories, but higher frequencies for more neutral ones.
3. After obtaining the frequencies distributions, put all items in GGUM and obtain the item parameters (using GGUM instead of MULTILOG because in GGUM, we don’t need to recode any responses, but in to use MULTILOG, the dominance model in other words, you need to first reversely code the items that require reverse coding).
4. After obtaining the item parameters, put them as well as item response into the excel MACRO called MODFIT, and with the fit plots and stats obtained, try to detect any abnormal items.
5. Mean chi^2/df – rule of thumb: 3
6. Plot: if discriminating, then item is fine; otherwise, not fine.
7. In DIF study, delete items based on plot that are problematic. If one item is not good in one group, delete it in both groups even though it works fine in the other group.
8. ~~Use the remaining items that exist in both groups to run a DIF analysis under GGUM, using Wei’s R code.~~

============================================================================

1. Before running DIF analysis for dominance model, items need to be all reversely coded.
2. To decide which items to be reversely coded, refer to GGUM model fit plots, FLs came along with the scale, and single item fit statistic.
3. If item performs well in one sample but not the other, then almost definitely this items has DIF, so don’t delete the item, but instead reverse (or not) the item based on the sample where it works fine.
4. For intermediate items, plots may indicate that the item does not perform poorly, but is not clear if the item should be reversed. In this situation, just keep the item without reversing it.

=============================================================================

1. Use all common items (reverse code all that need to be reversely coded) of both samples surviving the MODFIT plot check, and throw the item responses in MULTILOG, and obtain items pars
2. In MULTILOG, the a parameter is 1.7\*a, so after we obtain item pars from 13, select items again. This time, delete items that have a parameters smaller than 0.51 in both groups 🡪this how we got the final items used for dominance model DIF analysis (log likelihood ratio and effect sizes before summer break) 🡪 see folder 0420\_free&constrain\_2 under dropbox for pars
3. After obtaining the item parameters, do constrained baseline model first using R and data (item responses; reverse coding done) 🡪 constrained baseline model has inflated Type I error rate, meaning if an item is indicated as having no DIF using a constrained baseline model, then it really DOES NOT HAVE DIF 🡪 more conservative way to find non DIF items for linking purpose
4. After obtaining chi-square statistics for each item using the constrained baseline model, figure out item with no DIF 🡪 in our case, all items have DIF 🡪use items having chi-square statistics that are the closest to chi-square critical value as non DIF items and to serve as linking items for free baseline model (this conduct is valid because we have a large sample, which mean our tests have large power, and therefore, any diff across samples may be shown as significant)
5. With linking items identified for each facet respectively, we use the free baseline model for a more accurate DIF analysis 🡪chi-square values are negative, indicating null hypothesis fit better than H1 🡪 again, all items are shown to have DIF 🡪 large sample, high power, sigh….
6. Therefore, we switched to Nye’s effect size indices (MATLAB code can be found in dropbox, so can the revised version in R) 🡪rule of thumb: effect size <0.2 🡪 no DIF 🡪 we actually have a lot more items that have not DIF 🡪 yay!
7. Go back to looking at each items that have been shown to have DIF in 18, and figure out conceptually why they have DIF 🡪 based on the content and wording of the items
8. Run DIF analysis using GGUM, and compare differences between GGUM and dominance model

=============================================================================

1020\_2015

1. GGUM gave unsatisfying item parameters; to be more specific, some estimates are extremely large, as what we encountered with the dominance model
2. Fritz suggested that we use MCMC GGUM to compute item parameters, because MCMC GGUM uses Bayesian estimation, which hopefully will return less crazy estimates of parameters.
3. Tried MCMC with default priors coming along with the example syntax, but didn’t work 🡪 not that the est. are too wild, but they are just too good to be true 🡪 everything is similar (small s.d.) and similarly good 🡪 might be because the priors are so strong that they overwhelmed the estimation.
4. Therefore, attempts were made to loosen (weaken) the priors a little bit 🡪 changing them to non-informative priors (AKA most of the time it’s the uniform distribution that is flat) 🡪 first only the prior for alpha was altered, but still results are similarly good; 🡪then both priors of alpha and delta were changed 🡪 still not much improvement 🡪then priors for all 5 pars (alpha, delta, tau 1, tau 2, and tau 3) were altered to uniform distribution (with the default support area) 🡪again, no improvement. Not sure what to do next
5. Thinking of using GGUM2004 to run the constrained and free baseline model as under the dominance model using MULTILOG, and then run effect size analysis?

=================================================================================

1201\_2015

1. I was trying to call GGUM2004 from R using shell (), but failed and no error message was printed. R just keeps running and a cmd.exe window opens, and nothing happens other than this.
2. Reported this to Fritz, and he suggested that I email Wei and ask for help from ATLAS located in Lincoln Hall. However, Wei didn’t reply and ATLAS couldn’t help because it looked like GGUM2004 was what has gone wrong, which they know little about.
3. Finally I figured it out – first you need to change the working directory in R to C:/SGR\_GGUM, and then only write new syntax file by overwriting the CDMDC8 syntax file in that folder – This is the only way that the program can be called properly from R. Last but definitely not the least, you have to run it on a computer where the GGUM2004.3au file can be recognized (i.e. Jing’s computer in Brent’s lab).

=====================================================================

0124\_2016

1. When running the R program, encountered error messages like “Singularity when inverting DELTA/ALPHA for item #20 in Model 8” (have a picture of it in the cell phone). Also had a couple of items where no one chose option “1”.
2. Had to give up the problematic items (what these items are can be found in the word document regarding decisions on items)
3. Sys.sleep() is a command that asks R to wait until the GGUM2004 finishes running before it proceeds to writing the results to excel files. The number of seconds to put in “( )” really depends on how long it takes GGUM2004 to run on an item, which means it may vary drastically, ranging from 6 seconds to as long as 160 seconds.
4. Had a spreadsheet with the diagnoses of each of useable items using the dominance model, effect sizes, and the GGUM.
5. For the O facet, after using the linking items picked by the constrained baseline model to run the free baseline model, 3 out of the 6 non-linking (i.e. flagged as DIF by the constrained baseline model) items turn out to be of no DIF. I guess this is consistent with what they found in simulation that the constrained baseline model does have a higher type I error rate (i.e., giving that the item has no DIF, the probability of getting a chi-squared value that exceeds the critical value, indicating that the item has DIF). All the other 4 facets had no such issue.
6. In MLG, the LRs that are reported are the -2\*LR, while in GGUM2004, the LRs that are reported are just LR. In my R code for MLG, LR.dif = LR.base - LR.out, while in my R code for GGUM2004, LR.dif = -2\*(LR.out – LR.base). As for the original Log-likelihood ratio test, the default form is -2\*(LR of the base model – LR of the alternative model). The difference in the statistics reported in the two software has led to different decision criteria (for MLG, if the difference is positive, it means that the alternative model is better, while for GGUM2004, a negative difference indicates that the alternative model is better).

Questions: How should I justify comparing the 3 methods, given that their results are not based on completely the same bunch of items?

=========================================

Updated Feb 18, 2016

1. I showed Fritz the spreadsheet with the results of DIF analyses using the dominance model, effect sizes, and GGUM. He says the results look great, and next I need to do model fit. He doesn’t want me to do a usual fit with MODFIT, but with Alberto Maydeu-Olivares’s bivariate GOF method.
2. Liwen sent me the R package written by Alberto’s coauthor, and it’s for the graded response model. It uses the parameter estimates and acov matrix computed by Mplus to compute the GOF indices. I had a problem, and it’s when I run the package, error messages were returned saying the matrix is singular. Liwen asked me to run the code line by line, which also helps to understand the code.

===========================================

Updated Mar 20, 2016

1. Successfully ran the R package Liwen sent me (responses should be coded as 0123; no missing allowed). The parameter “eta” may be a little confusing, and what it serves is just as an “x” in a normal function like x+5, and the function integration (eta, lower = -6, upper = 6) defines eta (from -6 to +6). The “eta” doesn’t really have a fixed value. It’s the variable that we integrate on.
2. The indices are too sensitive, and it turned out that most of the items on the scales fit terribly, which may first, due to bad items, or second, local dependence (two items that are too similar in content will have terrible pair fit indices).
3. Fritz therefore suggested that I switched back to MODFIT, which most people use to do model fit.
4. MLG ran normally in MODFIT, and everything looked cool. However, GGUM refused to run with items containing options that nobody endorsed (A and O), and I had to therefore dichotomize these two facets, and rerun the baseline models DIF tests before running MODFIT. The error on singularity occurred again while I was running the baseline models in R. This time I made sure the data was read correctly. Liwen suggested that I get rid of these error items one at a time to see if the models run normally with a specific item. Fritz, on the other hand, suspected that these “error items” may have too small alphas. He suggested that I take a look at the GGUM item parameters, or plot empirical PDFs. Empirical PDFs are where I plot the top, medium, and bottom 10% percent of the sample against the proportion of them endorsing the same item. If the plot is a rather flat line, then it indicates poor discrimination.

Updated 08032016:

1. Because of all the errors, Fritz has asked me to switch the topic from DIF in real data to a simulation investigating what crashes GGUM2004 and what the potential solutions are.
2. Simulation: 1st round: the 0 frequency problem is serious, so Fritz asks for a 2nd round of simulation, which focuses on solving the 0 frequency problem, because 0 frequency items are simply “wasting” the items. We hope that by reducing 0 frequency items, there will be an increase in the proportion of Y cases (GGUM2004 runs normally).
   1. The two solutions for 0 frequency problem are: (1) collapsing CAT = 4 to CAT = 3 for the items with a 0-frquency option; (2) simulate more data for the focal group, but the theta distribution will be N (0, 1), instead of the deviant ones.
   2. Problem with simulating more data for the focal group: when the original N = 1000 for both the reference and the focal groups, and we simulate 1000 more people for the focal group, we have 3000 people in total, which exceeds the limit of the sample size that GGUM2004 can handle (2000) – Fritz emailed Jim Roberts during our meeting asking if there’s a hidden version of GGUM2004 that can handle more than 2000 people and Jim said no. So I suggested that we randomly sample 500 from the original focal group and 500 from the N (0, 1) sample, so that we’ll still have 1000 people for the focal group, and Fritz said OK.
   3. Now I need to add another computer for the simulation to cut down the total time the simulation takes, and I have my eyes on the computer at Fritz’s lab, but I need the administrator’s passcode to install R studio.

Updated 08142016:

1. For the past 10 days (from my last update), I’ve been running the collapsing and “collecting more data” conditions.
2. All collapsing analysis were run on my IBM computer, while only C19-21 for the “more data” analysis was. The other 21 conditions for the “more data” solution were run on the two computers in Brent’s lab, which have the problem of GGUM2004 pausing out of no obvious reason, and the case where the program pauses will have the exactly same output and parameter files as the case before it, because the analysis fail to run and everything just gets copied from the TEMPFILE folder.
3. To figure out which cases were the “paused” cases, I wrote a new code (“16.08.13\_Any\_pause\_when\_running. R”) which basically does 3 things: (1) which cases (both “Y” and “no-Y” cases are inspected) have the exactly same output file as the one right before it (the major thing to compare would be the start time in Line2 of the output file, because for the “no-Y” cases, there’s no information on fit or how long it takes to finish the analysis); (2) if the first case (“rep1\_ref”) of Condition C+1 paused and thus copied all files of the last case (“rep50\_i20”) from Condition C; (3) rerun all the paused cases and save the output and parameter files to where they belong (e.g., “C1\_results3”).

Updated 09.03.2016:

1. I’ve finished running simulation for collapsed and collecting more data, and showed them to Fritz. Fritz was a little disappointed to know that collapsing and combining (collecting more data) were not effective to reducing significantly % of screwed items. I, on the other hand, think the results have showed that combining is a better approach than collapsing, but Fritz said in reality it would be harder to collect more data than collapsing. So he still wants me to try to find a way show that collapsing is a great way to reduce screwed items.
2. I brought up again the discovery that an item that crashes GGUM2004 even when it’s not being studied usually has evenly frequency distribution
   1. Fritz is suspicious that although GGUM2004 returns an error message pointing to this “evenly distributed” item, it may actually be pointing to the item near it (happened to some early computer code/program that the true spot of the error is a few spaces before or after the spot flagged/reported by the program)
   2. I believe that the flag was in the place where it should be, because across different replications and conditions, the order of items (in terms of true parameters) changes, but GGUM2004 has successfully picked them out – therefore it may not be a problem of flagging the wrong item.
   3. Evenly distributed frequencies indicate low alpha parameters, so I’m now rerunning all truly screwed items and some suspicious Y items, manually, and put down which item screwed up based on the error message returned by GGUM2004. I’ll do this to all 3 simulations: the initial one, the collapsed one, and the combining one. Then I’ll refer to their frequencies and true parameters and see if I can find a pattern.
   4. Our ultimate goal is to come up with a flowchart that helps people diagnose what went wrong when they get the singularity error.
   5. About the potential suspicious Y items: most of the time, if an item is a screwed one, the the program crashes and the error message pops up in about 3 seconds after it starts running. However, when the screwed item is not the item being studied, sometimes (less than 5%) the program crashes after a longer time (6s to 60s have been observed), which means it’s possible that you’ll take it as an Y item until you stay for a little longer or much longer – which is totally unrealistic during the first round of simulation rerun, because I had more than 200 cases on average to rerun, and sometimes even more than 500.