1. Data file needs to be sorted by skill, student, and time (in that order).
   1. student : unique student id
   2. skill : unique skill name for each skill
   3. time: when the question was attempted
   4. right : 1 (correct) and 0 (incorrect)
2. We’ll be using a shared c library (find\_par.c file) to speed up the execution of nested for loops required for finding BKT parameters using grid search. First, create a shared library file (.so extension) using the C compiler for the “find\_par.c” file.
   1. In terminal, navigate to the directory where find\_par.c file is located.
   2. Create the shared library using the following command

gcc -fPIC -shared -o find\_par.so find\_par.c

* 1. This should generate a “find\_par.so” file in the same directory.

1. The find\_par.c file implements a grid search for BKT parameters where L0 and T are bounded to 0.99. S and G are bounded to 0.3. These can be easily changed in the find\_par.c file. Just make sure to repeat step 2 (generate .so file) if any changes are made to the find\_par.c file.
2. A brief description of the functions in the “bkt\_parameters.ipynb” file.
   1. get\_parmtrs\_all\_skills(): Takes a DataFrame (discussed in step 1) as input. Returns L0, T, S, G for each of the skills present in the DataFrame. (See step 1). The function will return BKT parameters only for skills for which there are at least 25 students who attempted at least 3 questions. This check for threshold can be changed or removed if needed.
   2. get\_predictions(): Takes DataFrame1 (discussed in step 1) and DataFrame 2 (BKT parameters obtained from get\_parmtrs\_all\_skill() function) as input. Returns predicted P(learned) and P(correct) with the combined DataFrame 1 and 2. Predictions are estimated only for the skills for which we BKT parameters are provided in the DataFrame2.
   3. collapsed\_metrics(): Takes DataFrame obtained form get\_predictions() function and returns collapsed rmse, collapsed auc\_roc, collapsed accuracy
   4. non-collapsed\_metrics(): Takes DataFrame obtained form get\_predictions() function and returns non-collapsed rmse, non-collapsed auc\_roc, non-collapsed accuracy
   5. ss\_nfold\_crossV(): Takes a DataFrame (discussed in step 1) and n (number of folds) as inputs. Implements n-fold student stratified cross validation. Returns collapsed[rmse, auc\_roc, auccary], non-collapsed[rmse, auc\_roc, auccary], collapsed[rmse, auc\_roc, auccary of each fold], non-collapsed[rmse, auc\_roc, auccary of each fold].