Table 1: Step-by-Step Method description

Step	Description
Step 1: Corpus preparation	Create a folder /DIALOGUE containing the transcripts. Each line must have the dialogue participant identifier (ID) at the start of the turn. For each dialogue or dialogue section, create two version, one with a tabulation between the participant identifier and the dialogue turn, and one version with only a space between the two just mentioned. Remove punctuation. Refer to file: USAGE-CORPUS-eg
Step 2: Creation of subdirectories	Create a folder that will contain the experiment itself, each dialogue or dialogue section containing five subdirectories: mkdir -p S01/AnalysisLemma S01/AnalysisLemma+POS S01/AnalysisPOS S01/AnalysisToken S01/AnalysisToken+POS
Step 3: Corpus Labelling	Scripts using the TreeTagger: make sure have the correct language. file-path for each dialogue, directing to the corpus files without tabulation. Place yourself in the root directory and execute: perl ./preprocesstreetag-lemma-CORPUS.pl perl ./preprocesstreetag-lemma-POS-CORPUS.pl perl ./preprocesstreetag-POS-CORPUS.pl perl ./preprocesstreetag-Token-POS-CORPUS.pl This step will need to check the correctness of language used as well as filepath at each new usage. Remove first line of all created files with a pipeline(USAGE-CORPUS-lemma-eg)
Step 4: Normalization of pronouns	Execute iyp-treat.pl script (files with tabulation for L1 token level, and output files from labelled corpus for other levels) perl ./iyp-treat.pl -i S01Full.txt Move files to correct subdirectory: mv S01Full.txt.iy.p0.c1 S01/AnalysisToken
Step 5: Assign a time stamp to each turn	The time stamp assigned to each turn is random in its length but follow a chronological order: dialog-treat-time.pl perl ./dialog-treat-time.pl -i S01/AnalysisToken/S01Full.txt.iy.p0.c1
Step 6: Turn 10 times randomization	Execute randomization script to correct output file: perl ./transcriptrandomizer.5.pl -i S01/AnalysisLemma/S01FullLemma.iy.p0.c1.fmtd This operation does not support the use of pipeline, therefore, use shell script: sh ./randomizeToken.sh containing the execution for each file.
Step 7: Concatenation	Concatenation of actual and randomization files: cat S01/S01Full/AnalysisLemma/*.parsed >S01Full/mergedS01FullLemma.data
Step 8: Post Treatment	Addition of a column containing the levels identifiers: perl ./CORPUSPostProcessingLevel.pl (Addition of the level) Then create first main dataframe by concatenation: cat Dialogues/*Level.data >mergedCORPUS.data This dataframe contain each speaker turn with his count of Othershared and Selfshared repetitions, number of token, reality (Actual dialogue (0) or randomization), Ngram length, and level.
Step 9: In R, preparation of dataframe	Removing unused columns and factorization of speakers ID (for following steps): >createCORPUSDial.R
Step 10: Statistical Model	Tukey Test for Othershared and Selfshared repetitions: >Pvalue.FullCorpus.R The output files gives for each dialogue, n-gram length, level and speaker, the result P-Values, Odds Ratio and Confidence Intervals of the tests.
Step 11: Extraction of P-Values	Use the shell script: extractionPValues.sh that will execute all the perl files extracting the p-values from the previous step output files. Then merge them: cat *OS.txt >mergedPvalueOS.txt cat *SS.txt >mergedPvalueSS.txt
Step 12: Creation of final dataframe	Create file CORPUSDialData in excel file by merging the previous step files with their corresponding, dialogue ID, n-gram length, level and speaker ID. CORPUSDialData.csv