The Many Faces of Human Sociality: Uncovering the Distribution and Stability of Social Preferences

– Replication Guide –

Adrian Bruhin Ernst Fehr Daniel Schunk

February 08, 2018

Contents

1 Introduction

This guide describes how to replicate the main results of the paper:

Bruhin A., Fehr E., D. Schunk (2018): „The Many Faces of Human Sociality: Uncovering the Distribution and Stability of Social Preferences", *Journal of the European Economic Association*, forthcoming.

If you use any of the data or the code described in this paper in your own work, even in altered form, please do not forget to properly cite the above paper. Thank you.

Please note that the accuracy of your replication depends on the specific system and installation that you are using. Hence, small differences between the replicated results and the results reported in the paper may occur.

2 Requirements

To run the replication, you need

- An installation of the R Software Environment which can be obtained for free from the website www.r-project.org for various platforms (including Windows, Mac OS, and Linux).

  The replication was tested with version 3.3.2 of R on Mac OS Sierra (version 10.12.6). However, it should also work with any newer version of R and on a different platform.

- For running the out-of-sample regressions, it is convenient but not necessary to also have a installation of Stata (www.stata.com).
- A text editor if you want to view and edit the code.

3 Description of Files

The following data is included in comma separated value (csv) files:

- **choices_exp1.csv**, and **choices_exp2.csv**: The subjects' choices in the 39 dictator games and the 78 reciprocity games in Session 1 and 2, respectively. "sid" indicates the subject, "gid" the game. "dg" indicates a dictator game (=1) or a reciprocity game (=0). "order" is the (random) order in which the game appeared in the block of dictator or reciprocity games. "self_[A]" and "other_[A]" denote the payoff of player A and B, respectively, when A choses allocation [A]. "choice_x" indicates whether the subject chose allocation X. "q", "v", "s_[A]", and "r_[A]" indicate the situation as described in Section 2.1 of the paper.
- **choices_trustgames.csv**: The subjects' choices in the additional 10 trust games used for making out-of-sample predictions. "cost" is the cost of being trustworthy. "trustworthy" indicates whether the subject was trustworthy or not.
- **choices_rpgames.csv**: The subjects' choices in the additional 2 reward and punishment games used for making out-of-sample predictions. "ug_id" indicates the game. "payoff_p" is the proposer's payoff, i.e. the payoff of the other player B. "pyoff_r" is the subject's payoff. "c[1-6]" are the costs the subject incurs for implementing each of the possible reward- and punishment-levels. "r[1-6]" is the change in the proposer's payoff for each of the possible reward- and punishment-levels. "reward_punish" is the actual reward or punishment implemented by the subject. "pgoesleft_kind" indicates whether it was kind or unkind from the proposer (player B) to choose the actual allocation.
- **subjects.csv**: The subjects' individual characteristics. The labels should be self-explanatory.

The following estimation scripts are included. You can inspect them by opening them in your text editor:

- **1_replicate_individual_results.r**: An R script for replicating the individual-specific estimations.
- **2_replicate_aggregate_results.r**: An R script for replicating the aggreagate estimations.
- **3_replicate_finmix_results.r**: An R script for replicating the finite mixture estimations.
- **4_combine_OutOfSampleData.r**: An R script for combining the different data files and estimation results to create the out-of-sample predictions in the additional trust and reward & punishment games.
- **5_OutOfSampleRegressions.do**: An optional STATA do-file for assessing the power in making out-of-sample predictions.

  Note: If you do not have STATA, you can run the OLS regressions described in the do-file also in R. However, using obtaining cluster robust standard errors will be less convenient in this case.
- **99_combine_OutOfSampleData.r**: An R script containing various functions that are jointly used by "1_replicate_individual_results.r" and "2_replicate_aggregate_results.r".

4 How to Replicate the Main Results

To replicate the main results of the paper, please proceed as follows:

1. Copy all files into a folder that will be your working directory. For example:

   /Users/YourName/Documents/Replication/

2. Start the R environment and change the working directory to the above folder. You can do this by typing the following command in the R console:

   setwd("/Users/YourName/Documents/Replication/")

   It is important to set the working directory, as otherwise the R scripts will create at unwanted locations.

3. You can now replicate the individual-specific estimations by running the corresponding R script. You can run the script by typing the following command in the R-console:

   source("1_replicate_individual_results.r")

   The script will starts immediately with the individual-specific estimations for both Sessions. This takes a while, as for each subject, it cycles through a grid of 140 start value combinations.

   After the individual-specific estimations are finished, you see a summary of the results which should be very similar to the results reported in Table 5 of the paper.

   The script also identifies the 14 subjects that behaved so erratically that their individual-specific parameter estimates lay outside the range that is identifiable with the experimental design (see the 2nd paragraph of Section 4 for more details). The R script stores their ids in the newly created file "dropped_subjects_section4paragraph2.csv", so that the subsequent scripts can exclude these subjects from the estimations.

   The results of the individual estimations will also be stored in two newly-created files: "individual_est_exp1.csv" and "individual_est_exp2.csv".

4. Now, you are ready to replicate the aggregate results by typing the following command in the R-console:

   source("2_replicate_aggregate_results.r")

   This script should run much faster than the previous one and should display results that are very similar to the ones reported in Table 1 of the paper.

   The script will also save the point estimates of the aggregate estimations in two files, "svExp1_1cl.csv" and "svExp2_1cl.csv", so that they can be used to create good start values for the type-specific finite mixture estimation.

5. You can now run the type-specific estimations using the finite mixture model by typing the following command in the R-console:

   source("3_replicate_finmix_results.r")

   This script immediately starts with the finite mixture estimations with K=3 preference types. It will take a while, as the script uses a version of Dempster et al.'s (1977) Expectation

Maximization (EM) algorithm which better deals with the log likelihood's non-linearity at the cost of being relatively slow (for more details, see the book "Finite Mixture Models" by McLachlan (2000).

At the end, you should get results that are very similar to the ones reported in Table 2 of the paper. If you fail to replicate these results, this may be because the estimation converged towards a local maximum of the log likelihood function. In this case, try to set a different random seed on line 8 of the script.

The script will save the results in an R-workspace-file "finmix_est_3cl.RData", so that they are available in the next step or later on. You can, for example, load this R-workspace-file to extract the individual ex-post probabilities of type-membership (see Equation (7) in the paper) in Session 2 by typing: pe2$tau

If you also want to replicate the results with a different number of preference types, for example K=2, adjust the number of types on lines 542 and 545 to nc=2 and rund the script again. Note that the more types you estimate the more likely it becomes that the estimation will converge towards a local maximum. Hence with K>3, you may need to try various different random seeds.

6. Now you can create the out-of-sample predictions in the additional trust and reward & punishment games by typing the following command in the R-console:

   source("4_combine_OutOfSampleData.r")

   This script will run very fast and will not display any output in the console. First, it combines (i) the sbubject's individual characteristics, (ii) their behavior in the trust and reward & punishment games, (iii) their individual-specific parameter estimates, and (iv) their type-specific parameter estimates. Subsequently, it predicts the subjects' trustworthiness and their desired reward- & punishment-level based on the individual-specific estimates and the type-specific estimates (see Section 4.5 in the paper). The results are stored in two files: "OutOfSample_trustgames.csv" and "OutOfSample_rpgames.csv".

7. Finally, you can assess the power of the individual-specific and type-specific estimates in making out-of-sample predictions in the trust games and the reward & punishment games by running the below do-file in STATA. Note that you may first have to define the working directory using the "cd" command:

   5_OutOfSampleRegressions.do

   The results of the OLS-regressions and in particular their R2 should be very similar to the ones reported in Tables 6 and 7 in the paper.

   If you do not have STATA, you can run these OLS regressions also directly in R. However, obtaining cluster robust standard errors may be a bit less convenient in this case and may require you to load an additional R-package.