Replication instructions for "Tug of War: The Heterogeneous Effects of Outbidding between Terrorist Groups"

Casey Crisman-Cox and Michael Gibilisco

Nov 2024

A note for replicators

Conducting constrained maximum likelihood estimation (CMLE) requires specialized (open source) software that we are only able to run using the Ubuntu Linux operating system. We provide detailed setup instructions below. All results were produced on a computer using Ubuntu 20.04.6 (Focal Fossa) using R 4.4.1 ("Race for Your Life") and Python 3.8.10. The entire archive should require no more than 2 GB of available RAM, but we recommend a machine with at least 4GB to be safe.

Replication package contents

Files marked with (U) require Ubuntu, with the setup as described below.

- Basic Information
  - `readme.md` plain text readme (this file)
  - `readme.pdf` This document in pdf format
  - `replicateMainText.sh` (U) Reproduces all results in the main text using the code files below in order. Tables and Figures are produced and placed in the Output folder.
  - `replicateAppendix.sh` (U) Reproduces all results in the Appendix using the code files below in order. Tables and Figures are produced and placed in the Output folder
  - `replicateAll.sh` (U) Runs both of the above replication scripts in order.
- Installation
  - `pyopt_setup_python3.sh` (U) A bash script designed to be run on a fresh installation of Ubuntu 20.04.6. It should also work on the live version or on existing Ubuntu 20.04.6 installations, but this is not tested. This file will install all the necessary outside software to replicate the results (Internet connection is required).
  - `Rpackages.r` An R script that installs all the `R` packages used here with the versions used here.
  - `sources.list` A sources file needed to replace the existing sources file if using a "live" version of Ubuntu 20.04.6 (see below)
- Data: These are both the original data used in the analysis as well as the merged and complete versions used in the analysis.

  - `acosta1993.csv` Acosta and Ramos' (2017) data to supplement missing data from the Global Terrorism Database (GTD).
  - `actionsSetup.Rdata` Produced by `buildDataSets.r`, below, aggregates the GTD data to monthly level. This produces the main measurement of the actions
  - `actionsSetup_byAttackType.Rdata` Contains terrorism data for each group, disaggregated by attack targets. Produced by `buildDataSets_byAttackType.r`, below.
  - `corruption_WBG.csv` World Bank attitudes towards corruption data for the Palestinian Territories.
  - `cpsr.csv` Survey data from CPSR/PCPSR.
  - `cpsr_GAZA.csv` Survey data from CPSR/PCPSR, disaggregated to just the Gaza Strip
  - `cpsr_WB.csv` Survey data from CPSR/PCPSR, disaggregated to just the West Bank
  - `ExtraFactors.rdata` Output from the latent measurement model for unemployment status and attitudes towards violence. Produced by `appendixD_latentMeasures.r`, below.
  - `gadm41_PSE.gpkg` Spatial administrative lines file used to assess the location of the West Bank and Gaza Strip for rainfall data.
  - `gtd.rdata` Terrorist attacks data from the GTD.
  - `jmcc.csv` Survey data from JMCC
  - `jmcc_2023.csv` Survey data from JMCC, with additional details collected later in the analysis
  - `jmcc_GAZA.csv` Survey data from JMCC, disaggregated to just the Gaza Strip
  - `jmcc_WB.csv` Survey data from JMCC, disaggregated to just the West Bank
  - `measurement.rdata` The results of the measurement model that produces the latent state variable $\tilde{s}^t$. Produced by `measurementModel.r`, below
  - `mortality_WB.csv` Infant mortality data for the Palestinian Territories from the World Bank
  - `otherattacks.rdata` Palestinian Islamic Jihad (PIJ) attacks data from the GTD, produced by `appendixD_PIJatacks.r`, below
  - `palestinian_deaths_2000_2008.csv` Palestinian fatalities data from B'Tselem (2000-2008)
  - `palestinian_deaths_2008_2020.csv` Palestinian fatalities data from B'Tselem (2008-2020)
  - `PalestinianDeaths.rdata` Aggregated data on Palestinian fatalities from B'Tselem. Created by `appendixD_aggregateDeaths.r`, below
  - `rainData.rdata` Aggregated data on extreme rainfall in the Palestinian territories from the Global Precipitation Climatology Centre (GPCC) and provided by NOAA. Created by `appendixD_buildraindata.r`, below
- Code

- Python3 (U)
  - `attackProbs.py` Generate attack probabilities from values
  - `estFunctions_NoComp.py` Estimation functions for the no competition model
  - `estFunctions.py` Estimation functions for the main model
  - `estFunctions_t4t.py` Estimation functions for the tit-for-tat model
  - `fitChangingDeltas.py` Fit the model with different discount factors
  - `fitMainModel.py` Fit the main model
  - `fitMainModel_t4t.py`Fit the tit-for-tat model
  - `fitNoCompetition.py` Fit the no competition model
  - `fitSensitivity.py` Fit the main model, but designed for parallel use
  - `genGiven.py` Helper function for the estimation functions
  - `usaParam.py` Generate utilities from parameter and data
  - `usaParam_t4t.py` Generate utilities from parameter and data for the tit-for-tat model
- R4.4.1
  - Results: A folder of results that are saved and used along the way, but are not directly included in the paper or appendix.
  - `appendixB.R` The numerical examples in Appendix B. Produces Figures B.1–4
  - `appendixB_equilibiraSearch.R` Searches for different solutions to the numerical example. We do not recommend actually running this.
  - `appendixC_alternatives.r` Consider alternatives to the main measurement model. Produces Tables C.3–4
  - `appendixC_geographic.R` Considers the geographical differences in the survey responses. Produces Figure C.2
  - `appendixD_aggregateDeaths.r` Aggregates Palestinian fatalities from B'Tselem. Creates `PalestinianDeaths.rdata`
  - `appendixD_buildraindata.r` Downloads rainfall data (1.2GB) from GPCC and produces the measures of extreme rainfall in the Palestinian Territories. Creates `rainData.rdata`
  - `appendixD_latentMeasures.r` Fits the latent measurement model for unemployment status and attitudes towards violence. Creates`ExtraFactors.rdata`.
  - `appendixD_PIJattacks.r` Aggregates PIJ attacks. Produces `otherattacks.rdata`
  - `appendixD_robustness.r` Fits the robustness checks in Appendix D along with the sensitivity analysis. Produces Tables D.1–7 and Figure D.1.
  - `appendixF.r` (U) Fits the simulations in Appendix F. Produces Figure F.1
  - `appendixG_VAR_comparison.r` Fits the Vector Autoregression

models in Appendix G. Produces Table G.4 and Figure G.1.

* `appendixH.r` (U) Fits the model at different temporal subsets. Produces Table H.1.
* `appendixI.r` (U) Fits the model with different discount factors. Produces Table I.1 and Figure I.1.
* `appendixJ1.r`(U) Fits the model with different discretization parameters. Produces Table J.1
* `appendixJ2.r` Fits the model with very coarse discretization parameters. Produces Table J.2 and Figures J.1–2.
* `appendixK.R` Analyses and interprets β. Produces Figure K.1
* `buildDataSets.r` Merges and aggregates the polling and terrorist attack data. Produces `actionsSetup.Rdata` and Figure A.1
* `buildDataSets_byAttackType.r` Merges and aggregates the polling and terrorist attack data but breaks it down by target type. Produces `actionsSetup_byAttackType.Rdata`
* `counterfactual_beta.R` Conducts counterfactual analysis on changes in the value of popularity (β). Produces Figure A.3.
* `counterfactual_gamma.R` Conducts counterfactual analysis on changes the ability to affect popularity (γ). Produces Figure 6.
* `counterfactual_kappa.R` Conducts counterfactual analysis on changes in the costs of terrorism (κ). Produces Figure A.4
* `counterfactual_kappa_discussion.R` Conducts counterfactual analysis on changes in the costs of terrorism (κ). Produces Table 5.
* `counterfactual_single_agent.R` Conducts the counterfactual comparisons with the single agent models. Produces Figure 5 and Table 4.
* `firststageboot.r` Function for a parametric bootstrap on the first stage
* `firstStageEstimation.r` Fit the first stage model. Produces Table 1
* `fitNoCompetition.r` (U) Fit the no competition model
* `gamma2trans.R` Function to produce the Markov transition matrix from the first-stage results
* `graphAttackProbs.R` Produces Figures 3 and 4
* `helperFunctions.r` Various helper functions
* `helper_functions_t4t.R` Various helper functions for the tit-for-tat model
* `liml.r` Functions for limited information maximum likelihood estimators (LIML) for IV regression
* `measurementModel.r` Uses the polling data to produce the continuous version of the state space. Produces `measurement.rdata`, Figures 1–2, and Tables C.2-3.
* `secondStageEstimation.r` (U) Fits the second stage model. Produces Table 2.
* `secondStageEstimation_t4t.r`(U) Fits the tit-for-tat model.

- Output
  - Figures. A folder containing all the produced figures
  - Tables. A folder containing all the produced tables in text format

Ubuntu 20.04.6 setup

The software setup currently uses Ubuntu 20.04.6. To install operating system

1. Download iso file and use it to create a "bootable" flash drive
   - The iso image can be obtained at this link.
   - Instructions for creating a bootable flash drive can be found here (Mac) or here (Windows).
2. Insert the flash drive and power on the computer
   - Enter the machine's boot menu by pressing the appropriate key repeatedly (the button to activate the boot menu varies with computer make and models – common keys include F12 (DELL, Most ACERs), F8 (Most ASUS), ESC (HP and some ASUS).
3. Select the option to boot from the flash drive USB
4. Select "install." WARNING: this will require you to format all or part of your hard drive so don't do it if you don't want that. You can also do a dual boot or live, but we only vouch for and fully support for installation.
5. Download the replication package (scroll your cursor to the upper left corner where it says "Activities" and then select the Firefox icon to access the internet)

Using a live version

If trying to use a live version Ubuntu, you should make the following adjustments.

1. Replace the live versions sources.list file with the one provided here. This can be accomplished using running the following shell command from the main replication folder

```
sudo mv sources.list /etc/apt/sources.list
```

This should be down between steps 3 and 4 in the replication instructions, below.

WARNINGS

1. We did not test the live version and do not support it beyond what is listed here.
2. Live versions store the OS in active memory, reducing what RAM is available for actual use. This means that the available memory for replication will be noticeably less than the computer's listed specs.

Other alternatives

Another alternative that should work is the Windows sub-system for Linux (WSL). This is probably the easiest approach for Windows users and should

work. However, we do not vouch for it and haven't tested it.

Replication

From a fresh installation of Ubuntu 20.04.6, you will need to use the following steps to prepare the replication environment.

1. Download the replication package
2. Extract the replication package to the desired location (`${REPDIR}`)
3. Open a terminal and navigate to Installation directory (`${REPDIR}/Installation`)
4. Run the file `pyopt_setup_python3.sh` using the command

```
bash pyopt_setup_python3.sh
```

This step may take up an hour depending on network speed and you may be prompted to press "Enter" at one or more points in the process. As software is downloaded, updated, or installed you may notice various background notifications appearing. These are normal and can be ignored.

A known bug sometimes appears where the installation hangs on "Pregenerating ConTeXt MarkIV format. This may take some time..." If you find yourself here for more than five minutes, press "Enter" 4–5 times and wait about another minute. This will often work to "unstick" it (see this link).

5. Run the file `Rpackages.R`

```
Rscript Rpackages.R
```

6. We are now ready to produce the results

Order of replication

Any file can be run and will produce the desired output as listed in its description. Two additional script files are provided to replicate the main paper and the appendix, respectively. These call the scripts, in order, to produce the files in the Output folder. Once the above installation is complete, these can be run by opening the terminal in this folder and running

```
nohup bash replicateMainText.sh &
```

or

```
nohup bash replicateAppendix.sh &
```

respectively. Note the use of `nohup` allows you to let these run in the background for as long as needed. This can be useful if you're replicating this on a remote device. Tables and Figures will be exported to the correct folders and any on-screen text will be collected within the file `nohup.out`. Alternatively, the command

```
nohup bash replicateAll.sh &
```

will run both of the above order. Finally, note that some of code files in the main text replication file also produce results for the appendix (e.g., some of Appendix G is actually produced by `modelFit.r`).

Replicating just the main text results takes roughly 40 minutes on an ASUS laptop with 4GB RAM with an Intel i3-5020U (4 threads). Replicating the appendix without Appendix F takes roughly one day on the same machine. Replicating Appendix F takes roughly a week on this machine, but the code in `appendixF.r` is parallelized and timing improves in the number of available cores.

Final note on RAM concerns

If the output in `nohup.out` text contains the phrase "killed" or outputs are missing in the output folder, then there is likely a memory problem. This can be due to either using a live version that eats up RAM or otherwise having insufficient available memory. Close all other applications (i.e., everything but the terminal running the code) and try again. If there are still issues, try upgrading to a higher memory machine or switching to a real installation rather than a live version. This is most likely to occur when building the two-step corrected standard errors in `fitMainModel.py` and creating Table 2.