Replication Package for "Cupid's Invisible Hand"

Alfred Galichon **Bernard Salanié**

November 19, 2021

This package includes the code and data needed to replicate our paper. It also contains information about how to obtain the raw data.

This README file is divided in five sections. The first documents the data sources used in the paper. The second describes and provides instructions for the programs in the `Code/DataTransformation` folder, which we used to prepare the data. The third section describes and provides instructions for the programs in the `Code/Analysis` folder, which estimates the model and produces the Tables and Figures in the paper.

1 Data Availability Statements

We use the 1970 US Census data from the Integrated Public-Use Microdata Sample (https://usa.ipums.org/usa/) to reconstruct the "availables". We downloaded the version we use in the published paper in May 2013. This data is publicly available. To cite from the IPUMS USA web site:

> IPUMS provides census and survey data from around the world integrated across time and space. IPUMS integration and documentation makes it easy to study change, conduct comparative research, merge information across data types, and analyze individuals within family and community context. Data and services available free of charge.

For the marriages data, we used the Vital Statistics data reported by states to the National Center for Health Statistics (https://www.cdc.gov/nchs/nvss/index.htm); we obtained it in May and September 2003. Unfortunately, NCHS has changed its policy due to "budget limitations" and it

does not make the individual data publicly available any more. We provide the 1971 and 1972 raw marriages data that we used as files `marr71.csv` and `marr72.csv` in the `Data/Raw` folder, along with files of "foreign" partners (see below).

References:

Steven Ruggles, Sarah Flood, Sophia Foster, Ronald Goeken, Jose Pacas, Megan Schouweiler and Matthew Sobek. *IPUMS USA: Version 11.0 [dataset]*. Minneapolis, MN: IPUMS, 2021. https://doi.org/10.18128/D010.V11.0

National Center for Health Statistics. *National Vital Statistics System [dataset]*. Downloaded from https://www.cdc.gov/nchs/nvss/index.htm in September 2003.

2 Data Transformation

We use the 1970 US Census data from Ipums-USA to reconstruct the "availables". We use the `marst` variable in the census to identify a person as either never married, currently married, or previously married (divorced or widowed). To calculate the number of available individuals, we simply add the never marrieds and previously married.

At this stage, we kept all individuals aged 16 to 75. We also kept race, education, state[1], and sampling weight.

Marriage records from the 1971/72 Vital Statistics of NCHS were used to construct the bivariate distributions of marriages. We store them in the `marrX.csv` files. Some marriages have one (and only one) "foreign" partner—someone who resides in a state that is not in our sample. We store these marriages in the `XforeignY.csv` files, for later use and correction as explained below.

2.1 Transformation Step 1

Step 1 converts the CSV files in the folder `Data/Input` into two files:

- `ChooSiowAvailables.csv`
- `ChooSiowMarriages.csv`

which it stores in the folder `Data/Intermediate`.

[1]The "state" of an individual is defined as his/her place of residence. The main text explains which states we retained for our analysis.

These two files only have the minimum set of variables required for our analysis, plus the state of residence, which is used in Step 2.

This is done using the `R` program `DataStep1.R` in the `DataTransformation` folder. The program

- reads in the CSV files for availables and for marriages.
- discards the variables we don't use in our analysis.
- accounts for the marriages that have one (and only one) "foreign" partner. These partners are subtracted from the number of availables.
- recodes the state numbers from the marriages data to Census codes; we use the variable `listStates` for this purpose.
- discards marriages in New York City (we don't have data for New York State).
- writes the transformed data in CSV format to `ChooSiowAvailables.csv` and `ChooSiowMarriages.csv` in the `Intermediate` data folder.

2.2 Transformation Step 2

The Python program `make_files_for_estimation.py` generates the files we use in the estimation program:

- `nx70n.txt` and `my70n.txt`: the margins, n_x and m_y in the paper
- `muxy70nN.txt`: the matching patterns μ_{xy}
- `varmus70nN.txt`: the variance-covariance matrix of the stacked vector $(\mu_{xy}, \mu_{x0}, \mu_{0y})$ where the ordering of the μ_{xy} is row-major.

All of these files are stored in `Data/Output`.

First we compute the numbers of marriages μ_{xy} by ages and reforms of the two partners, sample-weighted. We subtract one year to the ages for 1971, and two years for 1972.

We delete marriages whose partners are across the reform/non-reform boundary (`nR` and `rN` in the code). We do keep track of them by ages, so as to adjust the numbers of availables.

We compute variances and covariances assuming household-level sampling with the sampling weights in the raw data.

The code contains further documentation.

3 Code for Estimation and Simulation

All code is in Python, using `numpy`, `scipy`, `matplotlib`, and `pandas` in addition to the standard library packages. It relies on Knitro 12 (which requires a license) for optimization; and it includes the Knitro Python interface files.

The Python code has docstrings that describe what each function does.

The main program is `cupid_main.py`. The boolean variables `do_XXXX` define the models that it estimates and analyzes.

`cupid_main.py` first reads the formatted data from the `Data/Output` folder: the observed margins and matching patterns, their estimated sampling variance, and the values of the bases functions we use. Then it calls `maximize_loglik` in `estimate_model.py` to maximize the log-likelihood, which is defined, along with its gradient, as a Knitro callback function in `log_likelihood.py`. These functions take the stable matching as an argument; the stable matching itself is computed in `solve_for_mus.py`. For the variants of Choo and Siow, we use the IPFP algorithm introduced in the paper. For the FC-MNL model, we minimize the function $G(U)+H(\Phi-U)$, which gives $\mu = \nabla G(U)$ at the optimum. We set it up as a constrained optimization problem, where the constraint is $G_U(U) - H_V(\Phi - U) = 0$. The corresponding Knitro callbacks are `log_likelihood_MPEC.py`.

THe file `fcmnl.py` defines the functions G and H and their derivatives for a generic FC-MNL specification. A number of specifications are defined in `fcmn_specifs.py`.

To pass arguments between modules, we use the dataclasses defined in `cupid_classes.py`.

3.1 Additional Code

The program `check_fits.py` compares the values of the log-likelihood and the AIC and BIC for the specifications we use. The two figures in the main text of the paper were generated by the programs `plot_menshares.py` (Figure 1) and `plot_fcmnl.py` (Figure 2).

3.2 Execution Time

We used Python 3.9 with Knitro 12 on an iMac and/or on a Linux cluster. Estimating the variants of the Choo and Siow model only takes a minute or less, even on only one core. The estimation of the FC-MNL model is more computer-intensive, as it involves constrained optimization. Still, it only takes ten minutes or so.