Readme for "On the Optimal Design of Transfers and Income-Tax Progressivity"
Axelle Ferriere, Philipp Grübener, Gaston Navarro, Oliko Vardishvili

1 Overview

The materials are organized in three folders. "Data" contains the Matlab/Stata codes for preparing and analyzing aggregate data from National Income and Product Accounts (NIPA) and the Financial Accounts (FA), the Current Population Survey (CPS), the Panel Study of Income Dynamics (PSID), and the Survey of Consumer Finances (SCF). "Analytical_Model" contains Matlab codes for the solution of the simple model in section 2 of the paper. "Quantitative_Model" contains Fortran codes for the solution of the model in sections 3 to 5, as well as some Matlab codes for further processing of the results.[1]

2 Data

National Income and Product Accounts and Financial Accounts. The relevant NIPA and FA tables, described in detail in appendices B.4 and B.5 are contained in the folder "Aggregate_Data." "NIPA_FA.m" loads the data and computes the required statistics.

Current Population Survey. To work with the CPS, we use data from three sources. The main data source is the CPS Annual Social and Economic Supplement (ASEC) for 2013, as provided by IPUMS (Flood et al., 2021). Our data extract consists of the following variables: year serial month cpsid asecflag asecwth hhincome rentsub heatsub heatval stampval hseq pernum cpsidp asecwt relate age sex race marst sploc aspouse pecohab empstat labforce classwkr educ lineno wkswork1 uhrsworkly inctot incwage incbus incfarm incss incwelfr incretir incssi incint incunemp incwkcom incvet incsurv incdisab incdivid incrent inceduc incchild incalim incasist incother srceduc srcunemp srcwelfr srcwkcom ctccrd actccrd adjginc eitcred fedretir fedtax fedtaxac fica filestat margtax statetax stataxac taxinc, with the sample being the 2013 ASEC. Our Stata codes assume that the CPS data extract is stored in the subfolder "Datasets\IPUMS." It can be generated from IPUMS for free after registering, at https://cps.ipums.org/cps/.

For transfers we rely on the imputation procedure of the Congressional Budget Office (CBO), as described in Habib (2018). The CBO codes can be freely downloaded at https://github.com/US-CBO/means_tested_transfer_imputations; our Stata codes assume that this repository is stored in "Datasets\CBO_Imputation." Note that the main CBO imputation file requires to set a path to the data. In order to merge the IPUMS CPS with the CBO imputations, we require original CPS identifiers (see Appendix B.1), which we obtain from https://www.census.gov/data/datasets/time-series/demo/cps/cps-asec.2013.html, and which have to be stored in "Datasets\Census."

"CPS_Main.do" is the file that calls all other do-files in order to combine the data and produce the results. The code as provided produces an Excel file "cps_results_1.xslx", which is stored in the "Outputs" subfolder. "CPS_Tax_Function_Figures.m" in that folder uses this output to produce the figures based on the CPS data in the paper. For the robustness checks in the Appendix of the paper, "choose_version.do" has to be adjusted to the respective case.

Panel Study of Income Dynamics. Earnings dynamics are computed with the PSID, as described in Appendix B.3. "PSID_Main.do" uses the PSID packaged data, freely available at https://simba.isr.umich.edu/data/PackagedData.aspx after registration. It is assumed that the data files are in "Datasets\Raw_Data". Prepared data is stored in "Datasets\Prepared_Data" for the subsequent analysis. The code as provided imposes the sample selection and variable construction of the benchmark. For most of the other cases reported in Table B.5 of the appendix, "version_robust" in line 967 has to be adjusted to the corresponding number, as described in the code. The different earnings thresholds have to be adjusted in lines 1099 to 1101. To not control for education, the regression in line 1112 has to be replace with 1113.

Survey of Consumer Finances. We use the 2013 wave of the SCF to compute statistics on the distributions of income and wealth (Table 4 in the paper). We use both the full public data set and the summary extract public data set, the former for income variables described in detail in Appendix B.2 and the latter for the net worth variable. Both data sets are freely available from the Federal Reserve Board

[1] There is an additional folder "Matlab_Figure_Files" with a couple of files used for plotting, available at https://www.mathworks.com/matlabcentral/fileexchange/34055-tightfig-hfig) (Crozier, 2018).

at https://www.federalreserve.gov/econres/scf_2013.htm. "SCF_Main.do" loads the data and computes the income and wealth distributions.

3 Analytical Model

Results and figures for section 2 are produced by three m-files: "AM_Formula", "AM_Global", and "AM_Rates".[2] "AM_Formula" evaluates the analytical welfare formula and produces Figure A1 in the appendix of the paper. "AM_Global" solves the analytical model numerically and produces Figure 1 in the paper. "AM_Rates" produces Figure 2 in the paper using the results from "AM_Global".

Table 1 is produced by the "Main.m" Matlab file in the folder "Analytical_Model/Mirlees". This set of codes builds on Heathcote and Tsujiyama (2021). The file "RamseyHSVAffine.m" computes the optimal tax system with a loglinear function and a transfer, using the function "Ramsey_plus_affine_without_epsilon_grid.m".

4 Quantitative Model

Below we discuss the content of each folder within the "Quantitative_Model" folder.

Discretization. The discretization of the productivity process is done in Matlab, by the file "Discretization_Main.m." This file produces files containing the productivity grids and transition matrices to be read by the Fortan code solving the model. x_vec is the productivity grid for all calibrations of the model except, "No Pareto", "Richer Poor", and "Low σ & φ", for which the productivity grids are x_vec_nopareto, x_vec_richerpoor, and x_vec_pref (see Tables 7 and D.2 for the different calibrations). Px_GMAR contains the transition matrix for all cases except "No Pareto Normal" (Px_AR), "No Pareto Persistent" (Px_AR_Robust), and "Low σ & φ" (Px_Pref). The discretization of the GMAR process uses the Matlab package provided by Alexis Akira Toda at https://github.com/alexisakira/discretization, which is included here for convenience.

Tax Function Illustrations. "Tax_Fct_Plots.m" produces Figures 4 and C.1 in the paper.

Modules. This folder contains Fortran routines that are used across cases. Note that the tax-and-transfer function is written in slightly different notation compared to the paper. Parameter names are $\theta = \gamma$ and $\xi = rt$. Also, in contrast to the text the exponent of the tax function is θ rather than 2θ; numbers are scaled accordingly. Finally, the functional form in the code in the code is more flexible than what we use in the end with an additional parameter $ot = 0$.

Modules Loglinear. This folder contains Fortran routines with the loglinear tax function.

Modules Elasticities. This folder contains Fortran routines for computing labor supply elasticities.

Modules Elasticities Short Run. This folder contains Fortran routines for computing labor supply elasticities.

Optimal Systems. This folder contains Fortran codes computing the initial stationary equilibrium given calibrated taxes and transfers, the final stationary equilibrium given optimal taxes and transfers, and the transition between the two for a number of cases. We use the Intel Fortran compiler and the Intel MKL library. Across the following cases the structure is identical, so we describe the benchmark in more detail and then list the remaining cases. At the end, we also describe the codes for the global optimizer.

- *Benchmark_TT.* "Main_Benchmark_TT.f90" runs the computation for the optimal system with progressive taxes and phasing-out transfers for the "Benchmark" calibration. A basic make-file, which compiles this and the modules into the executable, is included. Inputs such as parameters and the discretized productivity process are in the "Input" subfolder. Results are stored in the "Output" subfolder. Calibration targets are reported after the initial steady state is computed. The "Output" folder also contains a number of Matlab files to compute further results,

[2] This section requires that the CompEcon Toolbox of Miranda and Fackler (2004) is on the Matlab path. The toolbox is available at https://github.com/PaulFackler/CompEcon.

"QM_Consumption_Equivalents.m" for consumption equivalents (Figure 9)[3], "QM_Rates.m" for plots of tax rates for the calibration and the optimum (Figures 6 and 7), "QM_Transition_Paths" for the paths of some variables along the transition (Figure 8), and "QM_Wealth_Effects" for the wealth effects.

- *Benchmark_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "Benchmark" calibration.
- *Benchmark_Affine.* Optimal system with affine taxes for the "Benchmark" calibration.
- *Benchmark_Loglinear.* Optimal system with loglinear taxes for the "Benchmark" calibration.
- *Benchmark_SteadyState.* Optimal system when maximizing steady state welfare with progressive taxes and phasing-out transfers for the "Benchmark" calibration.
- *Benchmark_BEGS.* Welfare decomposition à la Bhandari et al. (2022) for the benchmark calibration with phasing-out transfers.
- *Benchmark_Elasticities (short run).* Labor supply elasticities for different persistence of the shock for the benchmark calibration with phasing-out transfers.
- *NoPareto_TT.* Optimal system with progressive taxes and phasing-out transfers for the "No Pareto" calibration.
- *NoPareto_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "No Pareto" calibration.
- *RicherPoor_TT.* Optimal system with progressive taxes and phasing-out transfers for the "Richer Poor" calibration.
- *RicherPoor_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "Richer Poor" calibration.
- *NoParetoNormal_TT.* Optimal system with progressive taxes and phasing-out transfers for the "No Pareto Normal" calibration.
- *NoParetoNormal_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "No Pareto Normal" calibration.
- *NoParetoPersistent_TT.* Optimal system with progressive taxes and phasing-out transfers for the "No Pareto Persistent" calibration.
- *NoParetoPersistent_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "No Pareto Persistent" calibration.
- *LowSpending_TT.* Optimal system with progressive taxes and phasing-out transfers for the "Low Spending" calibration.
- *LowSpending_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "Low Spending" calibration.
- *LowSigmaPhi_TT.* Optimal system with progressive taxes and phasing-out transfers for the "Low σ & φ" calibration (note: for this version the preference parameters in "Globals.f90" in "Modules" have to be changed to the values described in the paper).
- *LowSigmaPhi_UBI.* Optimal system with progressive taxes and lump-sum transfers for the "Low σ & φ" calibration (note: for this version the preference parameters in "Globals.f90" in "Modules" have to be changed to the values described in the paper).

[3]Note that for Figure 9 in the paper we combine the consumption equivalents of this case with those of the optimal loglinear plan. This file produces the figure without the loglinear case.

- *TikTak.* We solve for the optimal tax systems using the global optimization algorithm proposed in Arnoud, Guvenen, and Kleineberg (2022) and Guvenen (2011), available at https://github.com/serdarozkan/TikTak#tiktak. We store a number of unchanged files in "Modules_TikTak." Additionally, we include the files required for optimization across different cases: optimizing with phasing-out transfers (*TT_TikTak*), with lump-sum transfers (*UBI_TikTak*), steady state only (*SteadyState_TikTak.*), and steady state only with lump-sum transfers holding progressivity fixed (*GKV_TikTak.*). To run an optimization, one needs to combine the files for the respective case with the generic "TikTak" files, inputs as in the "Input" folders for all the cases, and the generic "Modules." Additionally, TikTak requires a configuration file as input. These are included in the "Configuration_Files" folder. The order of compilation is given by ifort -o Main.exe Sort.f90 Toolbox.f90 Globals.f90 ModuleInitialize.f90 ModuleSaving.f90 Functions_wa.f90 ModuleDistributions.f90 Functions_wa_TR.f90 ModuleSolve_GLO_LOC.f90 ModuleAggregates.f90 ModuleAggregates_QTR.f90 ModuleSteady.f90 ModuleSolveEqm.f90 ModuleQERRORS.f90 ModuleQTRANSITION.f90 ModuleOptTransition.f90 nrtype.f90 stateControl.f90 genericParams.f90 utilities.f90 simplex.f90 objective.f90 minimize.f90 TiktakGlobalSearch.f90 -qopenmp -mkl. The program can be executed with "Main.exe 0 config.txt a", where the respective configuration file has to be named "config.txt." To speed up computation, after starting the first instance more programs can be started with "Main.exe 1 a." Typically, we run one optimization on a high-performance computing cluster, where the model solution is parallelized on 16-40 CPUs within a node and 8-20 parameterizations are run on separate nodes simultaneously.

- *Grid_Searches.* We solve for the optimal tax systems using a grid search for the following cases: optimal affine, optimal loglinear; and to build Figure 10. A basic make-file, which compiles this and the relevant modules into the executable, is included. To run the code, it is necessary to input sequence numbers from 1 to no_prog specified in line 32 or 33 of the Main_Pretesting.f90 file. Files "read_info.m" and "Figure_10.m" have to be run sequentially to generate Figure 10.

References

Arnoud, Antoine, Fatih Guvenen, and Tatjana Kleineberg (2022). "Benchmarking Global Optimizers". In: *Working Paper*.

Bhandari, Anmol, David Evans, Mikhail Golosov, and Thomas J. Sargent (2022). "Efficiency, Insurance, and Redistribution Effects of Government Policies". In: *Working Paper*.

Crozier, Richard (2018). "tightfig(hfig)". In: *Matlab Central File Exchange*.

Flood, Sarah, Miriam King, Renae Rodgers, Steven Ruggles, J. Robert Warren, and Michael Westberry (2021). "Integrated Public Use Microdata Series, Current Population Survey: Version 9.0 [dataset]. Minneapolis, MN: IPUMS, 2021." In: *Dataset*.

Guvenen, Fatih (2011). "Macroeconomics with heterogeneity: a practical guide". In: *Federal Reserve Bank of Richmond Economic Quarterly* 97.3, pp. 255–328.

Habib, Bilal (2018). "How CBO Adjusts for Survey Underreporting of Transfer Income in Its Distributional Analyses". In: *Working Paper*.

Heathcote, Jonathan and Hitoshi Tsujiyama (2021). "Optimal income taxation: Mirrlees meets Ramsey". In: *Journal of Political Economy* 129.11, pp. 3141–3184.

Miranda, Mario J. and Paul L. Fackler (2004). *Applied computational economics and finance*. MIT Press.