README File for Replication Package

Insurance versus Moral Hazard in Income-Contingent Student Loan Repayment

Tim de Silva, Stanford University, tdesilva@stanford.edu

Overview

The code in this replication package performs empirical analyses in the `empirics/` folder and produces the results from the life cycle model described in the paper in the `model/` folder. Two main components run all the code to generate the data for the figures and tables in the main body of the paper: (1) the empirical analysis in Python/Stata that processes administrative and survey data, and (2) the model in Fortran that requires high-performance computing resources. The empirical analysis is expected to take approximately 12 hours on a standard desktop, while the structural model requires an HPC cluster. The runtime of the structural model will vary depending on the available CPU resources in the cluster. The code was developed and run on the `xeon-p8` partition of the MIT SuperCloud Computing Cluster. Producing all the results in the paper required approximately 600,000 CPU hours.

Data Availability and Provenance Statements

☑ I certify that the author(s) of the manuscript have legitimate access to and permission to use the data used in this manuscript.

☑ I certify that the author(s) of the manuscript have documented permission to redistribute/publish the data contained within this replication package.

License for Data

No license is provided for the data as all datasets used are private and cannot be redistributed.

Summary of Availability

☐ All data are publicly available.

☑ Some data cannot be made publicly available.

☐ No data can be made publicly available.

Details on each Data Source

This analysis uses four private Australian datasets that cannot be redistributed:

1. Australian Longitudinal Individual File (ALife) 10% Sample

These data are used in `empirics/code/alife/.` To gain access to these data, follow the

instructions here: https://alife-research.app/info/gaining-access. Access to these data is provided online via a virtual machine, where all information entering or exiting the machine must be cleared and satisfy disclosure requirements. Accessing these data also requires an affiliation with an Australian university. It can take some months to negotiate data use agreements and gain access to the data. The author will assist with any reasonable replication attempts for a period of two years following publication.

2. Australian Longitudinal Individual File (ALife) Full Sample

These data are used in `empirics/code/alife_fullsample/`. These data cannot be accessed by anyone outside of the ATO's ALife division. The organizers of ALife ran the code provided in this replication package on their servers. The code was tested and developed on a virtual machine using a 10% sample and then transmitted to the ALife team to be run via GitHub. It can take some months to negotiate data use agreements and gain access to the data. The author will assist with any reasonable replication attempts for a period of two years following publication.

3. Multi-Agency Data Integration Project (MADIP) from ABS DataLab

These data are used in `empirics/code/datalab/.` To gain access to these data, follow the instructions here: https://www.abs.gov.au/about/data-services/data-integration/accessing-integrated-data. Access to these data is provided online via a virtual machine, where all information entering or exiting the machine must be cleared and satisfy disclosure requirements. Accessing these data also requires physical presence in Australia and affiliation with an Australian University. It can take some months to negotiate data use agreements and gain access to the data. The author will assist with any reasonable replication attempts for a period of two years following publication.

4. Household, Income and Labour Dynamics in Australia (HILDA) Survey

These data are used in `empirics/code/hilda/.` These data can be accessed by request at the following link: https://melbourneinstitute.unimelb.edu.au/hilda/for-data-users. Follow the instructions provided to obtain access.

5. Australian Mortality Tables

These data are used in `empirics/code/miscellaneous/.` These data are publicly available mortality data for males and females located at https://aga.gov.au/publications/life-tables/australian-life-tables-2005-07. The relevant data files are already included in the `empirics/data/mortality/`.

Computational requirements

Software Requirements

For Empirical Analysis in the `empirics/` Directory

- Python 3 with necessary packages. Can be installed with `pip install -r requirements_empirics.txt`
- Stata SE (or MP) 17

For Structural Model in the model/ Directory

- Intel Fortran Compiler Version 2021.9.0 with OpenMP
- Intel MPI Library for Linux OS, Version 2021.9
- Linux operating system (CentOS 7 required)
- SLURM job scheduler for HPC cluster management
- Python 3.9.15 with the following (standard) packages: pandas, numpy, matplotlib, pickle

Hardware Requirements

- *Empirical analysis*: Standard desktop computer
    - The code in `empirics/code/alife` and `empirics/code/datalab` was developed on Windows, which is the OS of the virtual machines provided.
    - The code in `empirics/code/hilda` and `empirics/code/miscellaneous` was developed on Mac OS.
    - The code in `empirics/code/alife_fullsample` was developed on Linux, which is the OS of the ATO's machines that host the population-level ALife dataset.
- *Structural model*: High-performance computing cluster with the same resources available as the MIT SuperCloud Computing Cluster.

Controlled Randomness

Empirical Analysis

Random seeds are set at multiple locations:

- line 5 in `fxns_bootstrap.py` files in `alife/`, `alife_fullsample/`, `hilda/`
- line 10 in `hilda/01_compute_hours_moments.py`
- line 9 in `alife_fullsample/fxns_bunching.py`
- line 19 in `datalab/fxns_bunching.py`

Structural Model

Random seed is set at line 237 of `model/Setup.f90`. This master seed controls all sources of randomness in the model's code.

Memory, Runtime, Storage Requirements

Approximate time needed to reproduce the analyses:

☐ <10 minutes

☐ 10-60 minutes

☐ 1-2 hours

☐ 2-8 hours

☑ 8-24 hours: `empirics/`

☐ 1-3 days

☐ 3-14 days

☑ > 14 days: `model/`

Approximate storage space needed:

☐ < 25 MBytes

☐ 25 MB - 250 MB

☐ 250 MB - 2 GB

☑ 2 GB - 25 GB: `empirics/`

☑ 25 GB - 250 GB: `model/`

☐ > 250 GB

☑ Not feasible to run on a desktop machine: `model/`

Details

Empirical Analysis

The empirical code was designed to run on standard desktop/laptop computers.

Structural Model

The structural model requires high-performance computing resources and was designed for HPC clusters with SLURM job scheduling. The SLURM job parameters are specified in `model/job.slurm`. The code uses shared-memory parallelization with OpenMP and distributed-memory parallelization with MPI.

Description of Code Files

Files in `empirics/code/alife/`

The files in this folder analyze data from the ALife 10% sample. The first two numbers (e.g., 01, 02) of filenames indicate the order in which they must be run. These files perform the build of the necessary datasets and then produce the required empirical results, as well as the file `occupation2_stats.csv`, which is used in `empirics/code/alife_fullsample/`. The file `main.sh` runs all the programs in their required sequence.

Files in `empirics/code/hilda/`

The files in this folder analyze data from the HILDA survey. These files calibrate some of the model's parameters and produce a file called `sdloghours_occupation2.csv` that is used in `empirics/code/alife_fullsample/.` The file `main.sh` runs all the programs in their required sequence.

Files in `empirics/code/alife_fullsample/`

The files in this folder analyze data from the ALife population-level sample. The first two numbers (e.g., 01, 02) of filenames indicate the order in which they must be run. These files perform the build of the necessary datasets and then produce the required empirical results. The file `main.sh` runs all the programs in their required sequence. This code requires uploading two files manually into the specified input directory: `occupation2_stats.csv`, `sdloghours_occupation2.csv`.

Files in `empirics/code/datalab/`

The files in this folder analyze data from the ABS MADIP dataset in DataLab. The first two numbers (e.g., 01, 02) of filenames indicate the order in which they must be run. These files perform the build of the necessary datasets and then produce the required empirical results. The file `main.sh` runs all the programs in their required sequence.

Files in `model/`

These files contain the Fortran code that produces the results from the structural life cycle model. The `.f90` source files correspond to the main code, which is compiled and submitted for execution on the necessary SLURM resources using `compile.sh`. There are three folders: `Input/`, which contains calibrated parameters and empirical moments needed in the code, `Output/`, which is the directory to which the Fortran code writes raw outputs, and `Cleaned/`, which is the directory to which processed outputs are written. The `.py` files contain Python code necessary for the post-processing of Fortran outputs in `Output/` to `Cleaned/`.

License for Code

Instructions to Replicators

Empirical Analysis

1. Obtain access to all datasets through the instructions provided above.
2. Navigate to `empirics/code/hilda/directories.py` and change the necessary directories based on the location of the HILDA files.
3. Run empirics`/code/hilda/main.sh`.
4. Copy `sdloghours_occupation2.csv` into `empirics/code/alife_fullsample/inputs/.`
5. Upload the files in `empirics/code/alife/` to the ALife virtual machine.
6. Change the necessary directories in `directories.py` based on the location of files on the ALife virtual machine.
7. Run `empirics/code/alife/main.sh`.
8. Copy `occupation2_stats.csv` into `empirics/code/alife_fullsample/inputs/`.
9. Submit the files in `empirics/code/alife_fullsample/` to be run by ALife on the full dataset sample. Whoever is running the code needs to set the directories in `directories.py`.
10. Upload the files in `empirics/code/datalab` to the ABS virtual machine.
11. Change the necessary directories in `directories.py` based on the location of files on the ABS virtual machine.
12. Run `empirics/code/alife/main.sh`.
13. Navigate to `empirics/code/miscellaneous/directories.py` and change the necessary directories.
14. Run `empirics/code/miscellaneous/main.sh`.

Structural Model

1. Obtain access to an HPC cluster with the necessary hardware and software requirements.
2. Use SFTP to transfer the files in `model/` to a location on the cluster.
3. SSH into the cluster and cd into the location to which `model/` was copied.
4. Modify the following lines of code based on the software available on the cluster:
    a. Set the temporary directory location on line 29 of `compile.sh`.
    b. Set the name of the module that loads the Intel Fortran Compiler and MPI on line 30 of `compile.sh`.
    c. Change the name of the partition on line 3 of `job.slurm`.

    d. Set the name of the module that loads the Intel Fortran Compiler and MPI on line 24 of `job.slurm`.
    e. Set the name of the module that loads the necessary Python installation on line 35 of `job.slurm`.
5. Check the resources available in the cluster and determine the number of CPU cores that can be used. Denote this number by `N`.
6. Run the following command in the terminal: `./compile.sh 1 N`. This will start the job that estimates the parameters in the baseline model (column (5) of Table 3).
7. From the `run.out` file that is produced by the Slurm job, copy the parameter estimates to line 26 of `SMMFunctions.f90`.
8. Run the following command in the terminal: `./compile.sh 2 N`. This will start the job that computes the standard errors of the parameter estimates.
9. Run the following command in the terminal: `./compile.sh 4 N`. This will start the job that produces the results for comparing existing repayment policies.
10. Run the following command in the terminal: `./compile.sh 8 N`. This will start the job that produces the results for computing and comparing constrained-optimal repayment policies.

List of tables and programs

The provided code reproduces all the data necessary to produce the figures and tables in the main body of the paper. In most cases, the code creates the exact underlying figure and table. The table below provides the mapping from figures and tables in the draft to the corresponding code files. In a few cases, the tables and figures are manually constructed using output from the code, as described in the table.

The only exceptions for results that have underlying data that are not reproduced by the code are the following:

1. Table 1: This table contains a subset of the outputs produced by the code for brevity.
2. Table 2: This table summarizes the text described in the paper.
3. Table 3: This table contains estimation results for many alternative models. The code only produces the estimation results for the baseline model, which is used to generate the rest of the analyses in the paper.

Exhibit in Draft	Name of Underlying File	Code File that Produces File
Figure I	figure1_real_crop_annotated.pdf	alife/02_bunching.py
Figure II	help_rates_dollars.pdf	miscellaneous/01_rates_dollars.py
Figure III	bunching_fixed_real_2005_2002_2002_2008_separate_debtonly.pdf	alife/02_bunching.py
Figure III	bunching_fixed_real_2005_2002_2005_2008_separate_debtonly.pdf	alife/02_bunching.py
Figure III	bunching_fixed_real_2005_2002_2003_2008_separate_debtonly.pdf	alife/02_bunching.py
Figure III	bunching_fixed_real_2005_2002_2006_2008_separate_debtonly.pdf	alife/02_bunching.py
Figure III	bunching_fixed_real_2005_2002_2004_2008_separate_debtonly.pdf	alife/02_bunching.py

Figure III	bunching_fixed_real_2005_2002_2007_2008_separate_debtonly.pdf	alife/02_bunching.py
Figure IV	occupationplot_slides_chg_label.pdf	alife/03_occupation_scatter.py
Figure V	bunching_binscatter_HRSP_20162016full.pdf	datalab/02_plots.py
Figure VI	bstat_agedebt.pdf	alife/02_bunching_heterogeneity.py
Figure VII	bstat_super.pdf	alife/02_bunching_heterogeneity.py
Figure VII	bunching_binscatter_house_pti_20162016full.pdf	datalab/02_plots.py
Figure VIII	fitplots_bunching.pdf	model/processing/01_plot_fit.py
Figure IX	existingcontracts.pdf	model/processing/01_results_existing.py
Figure IX	optimalcontracts.pdf	model/processing/01_results_optimal.py
Table I	summary_statistics.tex	Manually constructed from outputs to alife/02_summarystats.py
Table II	calibration_table.tex	Manually constructed
Table III	estimation_results.tex	Manually constructed
Table IV	table.tex	model/processing/01_results_existing.py
Table V	table3.tex	model/processing/01_results_optimal.py