HAFiscal Replication Package

DOI 10.5281/zenodo.18132886 Docker llorracc/hafiscal-qe

Paper: Welfare and Spending Effects of Consumption Stimulus Policies **Authors**: Christopher D. Carroll, Edmund Crawley, William Du, Ivan Frankovic, Håkon Tretvoll **Journal**: Quantitative Economics **Submission Repository**: https://github.com/llorracc/HAFiscal-QE

Generated: $(date +"%Y-%m-%d")

Overview

This repository contains the complete replication package for "Welfare and Spending Effects of Consumption Stimulus Policies", submitted to Quantitative Economics.

What is included: - LaTeX source for the paper (HAFiscal.tex) - All code for computational results (Python, HARK framework) - Data files and download scripts - Complete reproduction workflow - Computational environment specifications

The 'with-precomputed-artifacts' branch has the computed results prohibited from the main branch.

Estimated time to reproduce: - Minimal verification: ~1 hour - Full replication: 4-5 days (computational) - Paper compilation: 5-10 minutes

---

Quick Start

Prerequisites: Clone with All Branches

Before running any reproduction commands, ensure you have cloned the repository with all branches:

"'bash # Clone repository git clone https://github.com/llorracc/HAFiscal-QE cd HAFiscal-QE git fetch origin main

Fetch the with-precomputed-artifacts branch (REQUIRED for reproduction)

git fetch origin with-precomputed-artifacts "'

Why: The ‘with-precomputed-artifacts‘ branch contains generated objects (‘.bib‘, ‘.obj‘, ‘.csv‘ files) needed for reproduction. These files are excluded from ‘main‘ per QE requirements but the ‘reproduce.sh‘ script will automatically fetch them from this branch when needed. NOTE: These precomputed artifacts are NOT needed to run ‘./reproduce.sh –comp full‘ which takes 4-5 days and computes the needed objects.

Set up and test LaTeX and Python environments

“‘bash ./reproduce.sh –envt

Or set them up one at a time

./reproduce.sh –envt texlive # Latex ./reproduce.sh –envt comp_uv # Python “‘

Build the Paper

Do It Yourself ... “‘bash # Compile paper PDF from existing results ./reproduce.sh –docs

Note: Manual pdflatex/bibtex commands wont work because .bib files

are excluded from QE submission (only .bbl files are included).

Always use the reproduce.sh script to build documents.

“‘

... or in Overleaf

1. In Overleaf, select “New Project” -> “Import from GitHub”
2. Enter the repository URL: ‘https://github.com/llorracc/HAFiscal-QE‘
3. Overleaf will import the default branch (‘with-precomputed-artifacts‘)
4. Compile ‘HAFiscal.tex‘ using the standard Overleaf compilation process

Minimal Reproduction (Verify Code Runs)

“‘bash ./reproduce.sh –data # Calculate moments from SCF 2004 ./reproduce.sh –comp min # Minimal computational verification (~1 hour) ./reproduce.sh –docs main # Rebuild paper from results “‘

Note: If you havent fetched the ‘with-precomputed-artifacts‘ branch, the script will attempt to fetch it automatically from remote for this minimal reproduction. However, for offline use or faster access, fetch it during initial clone.

Full Replication (Reproduce All Results)

“‘bash ./reproduce.sh –data # Calculate moments from SCF 2004 ./reproduce.sh –comp full # Full computational replication (4-5 days) ./reproduce.sh –docs # Rebuild all documents “‘

Detailed Documentation

The following sections are excerpted from README/REPLICATION.md for convenience:

---

1. Data Availability and Provenance

Survey of Consumer Finances 2004

Source: Board of Governors of the Federal Reserve System
URL: https://www.federalreserve.gov/econres/scf_2004.htm
Access: Publicly available, no restrictions
Data License: Public domain (Federal Reserve data)

Data Files Used:

- `rscfp2004.dta` - Summary Extract Public Data (replicate-level data)
- `p04i6.dta` - Full Public Data Set (implicate-level data)

Download Method: Automated download via `Code/Empirical/download_scf_data.sh`

Variables Used:

- Normal annual income (permanent income proxy)
- Liquid wealth components (cash, checking, savings, money market accounts, stocks, bonds, mutual funds)
- Credit card debt (liquid debt component)
- Demographic variables (age, education)

Citation: Board of Governors of the Federal Reserve System (2004). Survey of Consumer Finances, 2004. Available at https://www.federalreserve.gov/econres/scfindex.htm

Data Construction: We follow Kaplan et al. (2014) methodology for constructing liquid wealth, as detailed in Section 3.2.2 of the paper.

Important Note: The Federal Reserve periodically updates older SCF data to adjust for inflation. If dollar values don’t match the paper exactly, this is

likely due to inflation adjustment. The relative statistics (percentages, ratios, distributions) should match closely.

Norwegian Population Data

Source: Fagereng, Holm, and Natvik (2021), "MPC Heterogeneity and Household Balance Sheets"
Access: Summary statistics and moments used for model calibration (published in the paper)
Note: Individual-level data not publicly available (Norwegian administrative data)

Data Files Included in Repository

The following data files are included in the `Code/Empirical/` directory:

- **`rscfp2004.dta`** - Summary Extract data for SCF 2004 in Stata format
- **`ccbal_answer.dta`** - Small file created from full public data set in Stata format

These files are also available from the Federal Reserve Board website:

Federal Reserve Board - 2004 Survey of Consumer Finances

Download and unzip the following files to reproduce our results:

- Main survey data (Stata version): **scf2004s.zip** -> **p04i6.dta** (`ccbal_answer.dta` is generated using this file)
- Summary Extract Data set (Stata format): **scfp2004s.zip** -> **rscfp2004.dta**

Place these `.dta` files in the directory `Code/Empirical` before running the file `make_liquid_wealth.py` (or a script that calls that file).

Data Processing

Some statistics hard-coded into the computational scripts are calculated from the SCF 2004. To reproduce these statistics, run the following do file:

```
./reproduce.sh --data
```

This script:

1. Loads the SCF 2004 data files
2. Constructs liquid wealth measures following Kaplan et al. (2014)
3. Calculates summary statistics used in calibration
4. Outputs results used in Table 2, 4 and 5 and in Figure 2

Additional data processing occurs in Python scripts located in `Code/HA-Models/`:

- `Target_AggMPCX_LiquWealth/` - Uses empirical moments for calibration
- Various scripts read the processed output files from `make_liquid_wealth.py`

Summary of Data Availability

- [OK] All data **are** publicly available
- [OK] No access restrictions or special permissions required
- [OK] Data can be downloaded automatically via provided scripts
- [OK] Data files included in repository for convenience
- [OK] Complete documentation of data sources and construction

Data Citations

In Bibliography The following data sources are cited in `HAFiscal-Add-Refs.bib`:

SCF2004:

```
@misc{SCF2004,
  author       = {{Board of Governors of the Federal Reserve System}},
  title        = {Survey of Consumer Finances, 2004},
  year         = {2004},
  howpublished = {\url{https://www.federalreserve.gov/econres/scfindex.htm}},
  note         = {Data files: Summary Extract Public Data (rscfp2004.dta) and
                  Full Public Data Set (p04i6.dta).
                  Available at \url{https://www.federalreserve.gov/econres/scf_2004.htm}.
                  Accessed November 2025}
}
```

In Paper Text The data is cited in the paper at:

- `Subfiles/Parameterization.tex` (Section 3.1, paragraph 4): First mention of SCF 2004 data
- `Subfiles/Parameterization.tex` (Section 3.2): Discussion of sample selection and construction of liquid wealth distribution

Ethical Considerations

This research uses publicly available secondary data from government sources. No primary data collection was performed. No Institutional Review Board (IRB) approval was required.

---

2. Computational Requirements

Hardware Requirements

Minimum:

- CPU: 4 cores, 2.0 GHz
- RAM: 8 GB
- Storage: 2 GB free space
- Internet connection (for data download)

Recommended:

- CPU: 8+ cores, 3.0+ GHz
- RAM: 16 GB
- Storage: 5 GB free space

Hardware Used for Results in Paper:

- CPU: Apple M2 (8 performance cores)
- RAM: 16 GB
- OS: macOS 14.4

Software Requirements

Required:

- **Python**: 3.9 or later
- **LaTeX**: Full TeX Live distribution (2021 or later)
- **Git**: For repository management
- **Unix-like environment**: macOS, Linux, or Windows WSL2

Python Package Manager:

- **uv** (recommended) or **conda**

Python Dependencies (automatically installed):

- numpy >= 1.21.0
- scipy >= 1.7.0
- matplotlib >= 3.4.0
- pandas >= 1.3.0
- econ-ark >= 0.13.0
- numba >= 0.54.0
- jupyter >= 1.0.0

LaTeX Packages: Included in `@local/texlive/` directory (no system LaTeX packages needed beyond base TeX Live).

Platform Support

- [OK] **macOS**: Fully supported and tested
- [OK] **Linux**: Fully supported and tested (Ubuntu 20.04+, Debian 11+)
- [OK] **Windows (WSL2)**: Supported via Windows Subsystem for Linux 2
- [NO] **Windows (native)**: Not supported

---

3. Installation Instructions

Step 1: Clone Repository

IMPORTANT: You must clone with all branches to access generated objects needed for reproduction.

```
# Clone repository
git clone https://github.com/llorracc/HAFiscal-QE.git
cd HAFiscal-QE


# Fetch the main branch for QE (with no precomputed artifacts or data)
git checkout main

# Fetch the with-precomputed-artifacts branch (REQUIRED for reproduction)
git fetch origin with-precomputed-artifacts
```

Why this is needed: The `with-precomputed-artifacts` branch contains generated files (`.bib`, `.obj`, `.csv`) that are excluded from the `main` branch per QE requirements but are needed for reproduction. The `reproduce.sh` script will automatically fetch files from this branch when needed, but the branch must be available locally or remotely.

Step 2: Set Up Python Environment

Option A: Using uv (Recommended)

- Install uv if not present:

curl -LsSf https://astral.sh/uv/install.sh | sh

- Then let the script handle the setup:

./reproduce.sh –envt comp_uv

- This will create a virtual environment named something like `.venv-linux-arm64`
  - you should activate this environment with the command `source .venv-[os-arch]/bin/activate`

Option B: Using conda

```
# Create environment from environment.yml
conda env create -f environment.yml
conda activate HAFiscal
```

Step 3: Verify Installation

```
# Check Python version
python --version  # Should be 3.9+
```

```
# Check key packages
python -c "import numpy; print(f'numpy: {numpy.__version__}')"
python -c "from HARK import __version__; print(f'econ-ark: {__version__}')"

# Check LaTeX
pdflatex --version
```

---

4. Execution Instructions

Main Reproduction Script

The primary way to reproduce results is via the `reproduce.sh` script, which provides a unified interface with multiple modes:

```
# View all available options
./reproduce.sh --help

# Make sure the appropriate computational environment is present
./reproduce.sh --envt

# Calculate moments from the SCF 2004
./reproduce.sh --data

# Minimal computational validation (~1 hour)
./reproduce.sh --comp min

# Full computational replication (4-5 days)
./reproduce.sh --comp full

# Quick document generation (5-10 minutes)
./reproduce.sh --docs

# Complete reproduction (all steps)
./reproduce.sh --all
```

Reproduction Modes Explained

`--data` - Data processing (1-2 minutes) Discussed in Section 1 above.

`--docs` - Document Generation Only (5-10 minutes) Compiles the paper PDF from existing computational results:

- Runs LaTeX compilation
- Generates bibliography
- Creates final HAFiscal.pdf

- **Does not** run computational models

Use case: Quick validation of LaTeX environment, or generating PDF after computational results are complete.

```
./reproduce.sh --docs
```

`--comp min` - Minimal Computational Test (~1 hour) Runs a subset of computational models with reduced parameters:

- Tests model infrastructure
- Validates Python environment
- Generates sample figures/tables
- Suitable for continuous integration testing

```
./reproduce.sh --comp min
```

`--comp full` - Full Computational Replication (4-5 days) Runs all computational models with paper-reported parameters:

- Solves heterogeneous agent models
- Generates all figures and tables
- Performs Monte Carlo simulations
- Replicates all quantitative results in paper

```
./reproduce.sh --comp full
```

Warning: This mode requires substantial computational resources and time. See Section 5 for detailed timing estimates.

`--all` - Complete Reproduction Pipeline (4-5 days + compilation) Runs full computational replication followed by document generation:

```
./reproduce.sh --all
# Equivalent to:
# ./reproduce.sh --comp full && ./reproduce.sh --docs
```

Running Individual Components

For more granular control, individual reproduction scripts can be run directly:

Environment Setup:

```
bash reproduce/reproduce_environment.sh
```

Data Processing

```
base reproduce/reproduce_data_moments.sh
```

Data Download:

```
bash Code/Empirical/download_scf_data.sh
```

Computational Models Only:

```
# Minimal test
bash reproduce/reproduce_computed_min.sh

# Full computation
bash reproduce/reproduce_computed.sh
```

Document Generation Only:

```
bash reproduce/reproduce_documents.sh
```

Standalone Figures/Tables (useful for debugging):

```
# Compile individual figure
cd Figures
latexmk -pdf Policyrelrecession.tex

# Compile individual table
cd Tables
latexmk -pdf calibration.tex
```

Benchmarking Your Run

To measure and record reproduction time on your system:

```
# Run with benchmarking
./reproduce/benchmarks/benchmark.sh --docs      # pdf documents
./reproduce/benchmarks/benchmark.sh --comp min  # everything quick
./reproduce/benchmarks/benchmark.sh --comp full # everything in paper
./reproduce/benchmarks/benchmark.sh --comp max  # robustness checks

# View benchmark results
./reproduce/benchmarks/benchmark_results.sh
```

See `reproduce/benchmarks/README.md` for detailed benchmarking documentation.

---

5. Expected Running Times

Reference Hardware (High-end 2025 laptop):

- CPU: 8+ cores, 3.0+ GHz (e.g., Apple M2, Intel i9, AMD Ryzen 9)
- RAM: 32 GB
- Storage: NVMe SSD
- OS: macOS / Linux / Windows WSL2

Reproduction Modes

Mode	Command	Duration	Output
Data moments	`./reproduce.sh --data`	1-2 minutes	Code/Empirical/Data
Document Generation	`./reproduce.sh --docs`	5-10 minutes	HAFiscal.pdf
Minimal Computation	`./reproduce.sh --comp min`	~1 hour	Validation results
Full Computation	`./reproduce.sh --comp full`	4-5 days	All computational results
Complete Pipeline	`./reproduce.sh --all`	4-5 days + 10 min	Everything

Individual Script Times

Script	Duration	Output
`reproduce_environment_comp_uv.sh`	2-5 minutes	Python/LaTeX environment
`download_scf_data.sh`	30 seconds	SCF 2004 data files
`reproduce_data_moments.sh`	5-10 minutes	Empirical moments
`reproduce_computed_min.sh`	~1 hour	Quick validation
`reproduce_computed.sh`	4-5 days	All figures and tables
`reproduce_documents.sh`	5-10 minutes	HAFiscal.pdf

Hardware Scaling

Minimum Hardware (4 cores, 8GB RAM, SATA SSD):

- Document generation: 10-20 minutes
- Minimal computation: 2-3 hours
- Full computation: 6-10 days

Mid-range Hardware (6-8 cores, 16GB RAM, NVMe SSD):

- Document generation: 7-12 minutes
- Minimal computation: 1-1.5 hours
- Full computation: 4-5 days

High-performance Hardware (16+ cores, 64GB RAM, NVMe SSD, GPU):

- Document generation: 5-8 minutes
- Minimal computation: 30-45 minutes

- Full computation: 2-3 days

Timing Variability

Running times may vary significantly based on:

- **CPU**: Core count, clock speed, architecture (x86_64 vs ARM)
- **RAM**: Amount and speed (impacts parallel solver performance)
- **Storage**: Type (NVMe > SATA SSD > HDD) affects I/O-heavy operations
- **Python packages**: Different BLAS/LAPACK implementations (OpenBLAS, MKL, Accelerate)
- **Compiler optimizations**: Numba JIT compilation settings
- **System load**: Background processes and resource contention
- **Random seeds**: Monte Carlo simulations have inherent variability

Benchmark Data

The times above are based on empirical benchmark measurements collected via the reproduction benchmarking system. To contribute your own benchmark or view detailed results:

```
# Run a benchmark
./reproduce/benchmarks/benchmark.sh --comp min

# View all benchmark results
./reproduce/benchmarks/benchmark_results.sh

# View benchmark documentation
cat reproduce/benchmarks/README.md
```

Benchmark Data Location: `reproduce/benchmarks/results/`
Documentation: `reproduce/benchmarks/BENCHMARKING_GUIDE.md`

For the most accurate estimate for your hardware, run `./reproduce.sh --comp min` first. This provides a reliable predictor: if minimal computation takes X hours, full computation typically takes 72-96 x X.

---

6. Results Mapping

For detailed provenance information about all figures and tables, including: - Data sources for each result - Generating scripts and functions - Source file locations - Computational dependencies

See: README/provenance.md

This document provides comprehensive documentation of: - **Figure provenance**: Source PDFs, generation scripts, and data dependencies for each figure - **Table provenance**: Generated content sources, creation scripts, and data inputs

for each table - **Parameter definitions**: Locations of model parameters and calibration values

---

7. File Organization

```
HAFiscal-QE/
|-- README.md                     # This file
|-- README.pdf                    # PDF version of this file
|-- LICENSE                       # See LICENSE file for license terms
|-- environment.yml               # Conda environment specification
|-- pyproject.toml                # Python dependencies (uv format)
|-- requirements.txt              # Python dependencies (pip format)
|-- HAFiscal.tex                  # Main LaTeX document
|-- HAFiscal.bib                  # Bibliography
|-- reproduce.sh                  # Main reproduction script
|-- reproduce.py                  # Python mirror (cross-platform)
|-- reproduce/                    # Reproduction scripts
|   |-- reproduce_computed.sh     # Run all computations
|   |-- reproduce_computed_min.sh # Quick validation test
|   |-- reproduce_data_moments.sh # Produce moments from SCF 2004
|   |-- reproduce_documents.sh    # Generate LaTeX documents
|   |-- reproduce_environment.sh  # Set up Python environment
|-- Code/                         # All computational code
|   |-- HA-Models/                # Heterogeneous agent models
|   |   |-- do_all.py             # Step-by-step computation of results
|   |-- Empirical/                # Empirical data processing
|       |-- download_scf_data.sh  # Download SCF data
|       |-- make_liquid_wealth.py # Construct liquid wealth measure
|       |-- *.dta                 # Data files (downloaded)
|-- Figures/                      # Figure LaTeX files
|-- Tables/                       # Table LaTeX files
|-- Subfiles/                     # Paper section files
|-- @local/                       # Local LaTeX packages
|-- @resources/                   # LaTeX resources and utilities
```

---

8. Known Issues and Workarounds

Issue 1: Windows Native Environment

Symptom: Scripts fail on native Windows (outside WSL2)

Cause: Bash scripts require Unix-like environment

Impact: Cannot run reproduction scripts on native Windows

Workaround: Use Windows Subsystem for Linux 2 (WSL2):

```
# In PowerShell (Administrator)
wsl --install
wsl --set-default-version 2
```

Then follow Linux instructions inside WSL2.

Issue 2: Long Computation Times

Symptom: Full replication takes many hours

Cause: Heterogeneous agent models are computationally intensive

Impact: Patience required for full replication

Workaround: Use `reproduce_computed_min.sh` for quick validation

Issue 3: Symlink Handling (Windows Users)

Symptom: Git shows changes to files in `images/` directory, or LaTeX compilation fails to find figures

Cause: Symlinks in `images/` directory not properly handled

Impact: - Repository may not clone correctly - Figures may not load during LaTeX compilation - Git may show spurious changes

Requirements: - **Symlink support required**: The `images/` directory contains symlinks to source figures - **Windows**: MUST use WSL2 (Windows Subsystem for Linux) - **Clone location**: MUST clone inside WSL filesystem (`~/` or `/home/`), NOT in `/mnt/c/`

Workaround (if symlinks were accidentally converted to regular files):

```
# Restore symlinks from git
git checkout HEAD -- images/

# Ensure git respects symlinks
git config core.symlinks true

# WSL2 users: Make sure you cloned in WSL filesystem, not /mnt/c/
pwd  # Should show /home/username/..., not /mnt/c/...
```

Why symlinks? - Single source of truth: Figures are generated in `Code/HA-Models/` subdirectories - No duplication: `images/` symlinks point to the source figures - Auto-update: Changes to source figures automatically visible - Pre-commit hooks protect symlink integrity

---

9. Contact Information

Technical Issues

For technical issues with replication:

- Open an issue: Go to the online GitHub repo and click on 'issues'
- Email: ccarroll@jhu.edu (Christopher Carroll)

Data Questions

For questions about SCF data:

- Federal Reserve SCF page: https://www.federalreserve.gov/econres/scfindex.htm
- Email: scf@frb.gov

Paper Content

For questions about the paper content:

- See author emails in paper
- Christopher Carroll: ccarroll@jhu.edu
- Edmund Crawley: edmund.s.crawley@frb.gov

---

10. Citation

If you use this replication package, please cite:

```
@misc{carroll2025hafiscal,
  title={Welfare and Spending Effects of Consumption Stimulus Policies},
  author={Carroll, Christopher D. and Crawley, Edmund and Du, William and Frankovic, Ivan ar
  year={2025},
  howpublished={Journal submission version},
  note={Available at \url{https://github.com/llorracc/HAFiscal-QE}}
}
```

Last Updated: January 2026
README Version: 1.1
Replication Package Version: 1.0

Version 1.1 Changes:

- Added comprehensive `reproduce.sh` documentation with all modes
- Updated timing data to use benchmark system measurements (not placeholders)
- Added hardware scaling examples (minimum, mid-range, high-performance)
- Integrated benchmark system references and instructions

- Added timing variability factors and explanations