REPLICATION PROCEDURE

Monetary Policy and Racial Differentials in Labor Market Outcomes

Tsvetomir Petkov and Charles L. Weise

December 22, 2025

1. Prerequisites

1.1. Hardware

Both the data generation and individual-level regression procedures are computationally expensive and we do not guarantee that any machine would successfully run all of the code that we provide. For reference, we used a Windows 11 PC with 32 GB of RAM and an Intel Core Ultra 7 258V CPU.

Before executing any code which stores data, ensure that you have sufficient disk space on your machine. Saving the data that is needed for Python and MATLAB routines requires around 6 GB of space, while saving the data needed for Stata routines (optional) requires additional 20 GB.

1.2. Software and Programming Languages

1.2.1. Python

We utilize Python (e.g., the 'pandas' library) to collect and format all of the data used in the paper and the 'statsmodels' library to obtain individual-level regression estimates via weighted least squares.

We recommend using a Python distribution such as Anaconda which can be freely downloaded here and provides you with access to the JupyterLab IDE.

The following Python packages are utilized and should be installed prior to running any data generation or statistical estimation code:

```
pip install pandas numpy scipy statsmodels ipumspy fredapi
```

1.2.2. MATLAB

We utilize MATLAB to develop SVAR-IV models and to collect IRF estimates. A MATLAB trial can be initiated here.

1.2.3. X-13ARIMA-SEATS

We utilize X-13ARIMA-SEATS (produced by the U.S. Census Bureau) to seasonally adjust labor market indicators. The ASCII version of the program can be obtained here.

After downloading, the X-13ARIMA-SEATS executable file should be extracted, renamed to 'x13as.exe', and placed in the 'PW1/Python' folder of the extracted replication package.

1.2.4. Stata (Optional)

Access to Stata is not required. However, we run all of our individual-level regressions in Stata after doing so in Python as a sanity check and, thus, provide the Stata command files as a reference.

We find that differences in the weighted least squares estimates produced via 'statsmodels' (Python) and via Stata are negligible (past the seventh decimal for binary outcome models). For that reason, we deem the choice between utilizing Python and Stata to be inconsequential for the task at hand.

1.3. Data

To successfully replicate our results in an automated fashion, you need direct access to the Current Population Survey (CPS) data via IPUMS and to macroeconomic data via FRED.

1.3.1. Obtaining an IPUMS API Key

Obtain your unique API key from IPUMS by creating an account here. Additional instructions and information can be found here.

1.3.2. Obtaining a FRED API Key

Obtain your unique API key from FRED by creating an account here. Additional instructions and information can be found here and here.

2. Data Collection

We do not directly provide any of the data that we use in our paper because we do not have ownership or distribution rights. However, this set of instructions and replication materials should help you obtain the exact data that we use as long as: 1) it is still available online; and 2) the tools/packages we use to collect and process it work properly as they did for us.

2.1. Manual Collection

Several macroeconomic and financial data series are not available on FRED and should be collected manually. Below, we point you to their online location. As of this document's date, all of the web links that we provide are functioning. We cannot guarantee that this will be the case in the future.

Individual placeholder CSV files for each of these manually-obtained data series are created in 'PW1/Python/data/macro'. Once you obtain the necessary data series (outlined below), paste them in the appropriate column (left blank) in each of the CSV files. When doing so, ensure that each series corresponds to its respective (abbreviated) file name and that within a particular file, each monthly value you recover corresponds to a correct date. Lastly, remove the '_temp' suffix from populated CSV files. It is used to indicate that the file originally served as a template/placeholder.

2.1.1. Commodity Research Bureau (CRB) Price Index

The Commodity Research Bureau (CRB) all commodities spot market price index can be obtained here.

If the CRB series is obtained at daily frequency, filter out all nonnumerical records, and aggregate to monthly level by keeping the last observation in each month.

2.1.2. Gilchrist and Zakrajšek (2012) Excess Bond Premium (EBP)

The Gilchrist and Zakrajšek (2012) Excess Bond Premium (EBP) can be obtained here.

2.1.3. Wu-Xia Shadow Federal Funds Rate (WXR)

The Wu-Xia shadow federal funds rate (WXR) can be obtained here.

2.1.4. Jarociński and Karadi (2020) Monetary Policy Instrument

The monetary policy instrument used in Jarociński and Karadi (2020) can be obtained here. We label this series 'MPI_JK'.

The CSV file provided by Jarociński and Karadi (2020) (the location in their replication package is 'data/data_var/us_ea_variables_shocks/us_shocks.csv') contains both the FF4 ('ff4_hf') and SP500 ('sp500_hf') high-frequency instruments. The former is what we refer to as the Jarociński-Karadi monetary policy instrument, used extensively throughout our paper.

2.1.5. Jarociński and Karadi (2020) "Poor Man's" Monetary Policy Instrument

To calculate the "poor man's" monetary policy instrument used in Jarociński and Karadi (2020), use the Excel formula below on a new column in their file ('us_shocks.csv') and apply it contemporaneously for each month of available data. We label this series 'MPI_JK_PM'.

```
= [ff4_hf] * ([ff4_hf] * ([sp500_hf] < 0))
```

2.1.6. Miranda-Agrippino and Ricco (2021) Monetary Policy Instrument

The monetary policy instrument used in Miranda-Agrippino and Ricco (2021) and updated in Degasperi and Ricco (2021) can be obtained here. We label this series 'MPI_MAR'.

2.1.7. Bauer and Swanson (2023) Monetary Policy Surprises

The monetary policy surprises (regular and orthogonalized versions) used in Bauer and Swanson (2023) can be obtained here. We label these series 'MPS_BS' and 'MPSO_BS', respectively.

2.2. Automated Collection

The instructions below assume basic knowledge of the utilized programming languages and capability to navigate the utilized software. Before continuing, you should have downloaded/extracted our replication package from GitHub and loaded the required software such as JupyterLab and MATLAB.

2.2.1. Generating CPS Data

Navigate to 'PW1/Python/gen_cps_data.ipynb'. Replace 'YOUR_CPS_API_KEY' with your actual CPS API key in the second cell. Replace 'YOUR_FRED_API_KEY' with your actual

FRED API key in the third cell. Refer to Prerequisites if you do not have either of the API keys. Run all four cells with code consecutively.

The expected output is a Pickle data file 'PW1/Python/data/cps_final.pkl', which contains all the needed CPS data from Jan 1992 to Feb 2020 along with the lagged 12-month average seasonally-adjusted state-specific unemployment rate abbreviated as 'sur_sa_1y_avg'.

This data file is used to obtain granular labor market indicators (described in detail in the third section of our paper) and to collect individual-level regression estimates (described in detail in the fourth section of our paper).

2.2.2. Generating Macroeconomic Data

Navigate to 'PW1/Python/gen_macro_data.ipynb'. Replace 'YOUR_FRED_API_KEY' with your actual FRED API key in the second cell. Refer to Prerequisites if you do not have this API key. Run all three cells with code consecutively.

The expected output is a CSV data file 'PW1/MATLAB/data/macro_data.csv', which contains all the needed macroeconomic and financial monthly data from Jan 1992 to Feb 2020 along with a selection of monetary policy instrument series.

This data file is used in MATLAB to run SVAR-IV models and to collect IRF estimates (described in detail in the third section of our paper).

2.2.3. Generating Granular Labor Market Indicators

Note that completing this step successfully requires that CPS data is saved (discussed above) and that 'x13as.exe' is present in the 'PW1/Python' folder (see Prerequisites).

Navigate to 'PW1/Python/gen_labor_data.ipynb'. Run all nine cells with code consecutively.

The expected output is a CSV data file 'PW1/MATLAB/data/labor_gaps_data.csv', which contains all the granular labor market indicator (unemployment rate and employment-population ratio) gaps from Jan 1992 to Feb 2020 used in the paper. For data labeling comprehension, refer to the mapping dictionaries (the third cell of 'gen_labor_data.ipynb').

This data file is used in MATLAB to run SVAR-IV models and to collect IRF estimates

(described in detail in the third section of our paper).

The last two cells of 'gen_labor_data.ipynb' print summaries of the monthly group-specific CPS observation counts, reported in Appendix G of the paper.

2.2.4. Generating Marginal Effects Data (Optional)

This is an optional step for obtaining the data needed to collect marginal effects estimates via Stata. Note that completing this step successfully requires that CPS data is saved (discussed above).

Navigate to 'PW1/Python/gen_me_data.ipynb'. Run all four cells with code consecutively.

The expected outputs are 96 CSV data files stored in the 'PW1/Stata/me_data' folder. These data files can be used to collect the estimates presented in Subsection 4.1 of the paper via Stata (see Estimates Retrieval).

2.2.5. Generating Oaxaca-Blinder Data (Optional)

This is an optional step for obtaining the data needed to collect regression estimates used in Oaxaca-Blinder decomposition via Stata. Note that completing this step successfully requires that CPS data is saved (discussed above).

Navigate to 'PW1/Python/gen_ob_data.ipynb'. Run all four cells with code consecutively.

The expected outputs are 6 CSV data files stored in the 'PW1/Stata/ob_data' folder. These data files can be used to collect regression estimates for Oaxaca-Blinder decompositions presented in Subsection 4.2 of the paper via Stata (see Estimates Retrieval).

3. Estimates Retrieval

3.1. SVAR-IV IRFs

Note that completing this step successfully requires that macroeconomic and labor market indicator gaps data are saved (see Data Collection).

Navigate to 'PW1/MATLAB/RunVAR.m' and run it to obtain a plot that is equivalent to Figure 1 in the paper. To generate the remaining SVAR-IV figures, make the following changes within the same file.

To obtain a plot equivalent to Figure 3:

```
VAR.singleVAR = false;
```

To obtain a plot equivalent to Figure 5:

```
VAR.singleVAR = false;
VAR.groupType = 'age';
VAR.gapType = 'BW';
```

To obtain a plot equivalent to Figure 6:

```
VAR.singleVAR = false;
VAR.groupType = 'age';
VAR.gapType = 'HW';
```

To obtain a plot equivalent to Figure 7:

```
VAR.singleVAR = false;
VAR.groupType = 'educ';
VAR.gapType = 'BW';
VAR.ageType = 'A2';
```

To obtain a plot equivalent to Figure 8:

```
VAR.singleVAR = false;
VAR.groupType = 'educ';
VAR.gapType = 'HW';
VAR.ageType = 'A2';
```

To obtain a plot equivalent to Figure 9:

```
VAR.singleVAR = false;
VAR.groupType = 'educ';
VAR.gapType = 'BW';
VAR.ageType = 'A3';
```

To obtain a plot equivalent to Figure 10:

```
VAR.singleVAR = false;
VAR.groupType = 'educ';
VAR.gapType = 'HW';
VAR.ageType = 'A3';
```

To obtain respective employment-gap plots, documented in Appendix A:

```
VAR.laborType = 'EPR';
```

To include Cholesky IRF estimates in a plot:

```
VAR.doCholVAR = true;
```

3.2. Marginal Effects

Note that completing this step successfully requires that CPS data is saved (see Data Collection).

Navigate to 'PW1/Python/run_me_regs.ipynb'. Run all seven cells with code consecutively.

The expected output is a CSV data file 'PW1/Python/results/me_lpm_estimates.csv', containing all of the marginal effects estimates obtained via weighted least squares along with heteroskedasticity-consistent standard errors, reported in Section 4.1 of the paper. In addition, the fifth and sixth cells of the code file print the results reported in Appendices I and H (Table H.1), respectively.

3.3. Oaxaca-Blinder Decomposition

Note that completing this step successfully requires that CPS data is saved (see Data Collection).

Navigate to 'PW1/Python/run_ob_regs.ipynb'. Run all eight cells with code consecutively.

The expected output is a CSV data file 'PW1/Python/results/ob_lpm_estimates.csv', containing most of the relevant Oaxaca-Blinder regression and decomposition estimates obtained via weighted least squares, reported in Section 4.2 of the paper. In addition, the sixth and seventh cells of the code file print the results reported in Table 1 and Appendix H (Table H.2), respectively.

3.4. Marginal Effects Regressions via Stata (Optional)

This is an optional step for obtaining marginal effects estimates via Stata. Note that completing this step successfully requires that marginal effects data is saved (see Data Collection).

Navigate to 'PW1/Stata/me_regs.do' and run it.

The expected output is a CSV data file 'PW1/Stata/me_lpm_estimates.csv', containing all of the marginal effects estimates obtained via weighted least squares along with heteroskedasticity-consistent standard errors, reported in Section 4.1 of the paper.

3.5. Oaxaca-Blinder Regressions via Stata (Optional)

This is an optional step for obtaining regression estimates used in Oaxaca-Blinder decomposition via Stata. Note that completing this step successfully requires that Oaxaca-Blinder data is saved (see Data Collection).

Navigate to 'PW1/Stata/ob_regs.do' and run it.

The expected output is a CSV data file 'PW1/Stata/ob_lpm_estimates.csv', containing Oaxaca-Blinder regression estimates obtained via weighted least squares as described in Section 4.2 of the paper.