# Replication package for “What Caused U.S. Pandemic-Era Inflation?” by Ben Bernanke and Olivier Blanchard

## Overview

The code in this replication package reproduces Figures 1-14 and Tables 2-5 in the paper. The code also reproduces Tables A2 and A3 from the appendix. The replicator should expect the code to run in under twenty minutes.

## Data Availability

Some of the data used in the paper are not publicly available and are thus not included in this replication package. Access to Bloomberg and Haver is required for successful replication.

To recreate the Bloomberg datasets, open the data/input\_data/confidential folder and open the Bloomberg\_Commodity\_data.xlsx file. Then, log into the Bloomberg terminal and, using Bloomberg’s Excel add-in, refresh the sheet. Bloomberg data should now populate the file.

To recreate the Haver datasets, open the data/input\_data/confidential folder and open the SP\_GSCI.xlsx file. Then, using Haver’s Excel add-in, press ctrl+d. Haver data should now populate the file.

## Computational requirements

### Software Requirements

* Stata
  + Software version 17
* MATLAB
  + Software version R2022a
  + The necessary packages are listed below.
    - Dynare 5.4

Dynare is a free software, both Windows and macOS compatible. The replicator should configure MATLAB (any version from 2014a to R2023a) for Dynare following instructions [here](https://www.dynare.org/resources/quick_start/#configuring-matlab-for-dynare-on-windows). After installation, the package can be used in any directory of the replicator’s computer. Dynare can installed [here](https://www.dynare.org/download/). The remaining packages needed for the .m files are listed below.

* + - Optimization Toolbox Version 9.4
    - Simulink Version 9.4
    - Simulink Test Version 9.4
    - Symbolic Math Toolbox Version 9.4
    - System Identification Toolbox Version 10.0
    - MATLAB Coder Version 5.5
    - Partial Differential Equation Toolbox Version 3.9
* Python
  + Software version 3.8.8
  + The necessary packages are listed below. They can all be installed by running the following command from Mac OS’s terminal or Windows’s Command Prompt after navigating to the code/Python folder:  
     pip install -r requirements.txt:
    - certifi==2023.5.7
    - charset-normalizer==3.1.0
    - contourpy==1.0.7
    - cycler==0.11.0
    - DateTime==5.1
    - econtools==0.3.2
    - et-xmlfile==1.1.0
    - fonttools==4.39.4
    - idna==3.4
    - importlib-resources==5.12.0
    - joblib==1.2.0
    - kiwisolver==1.4.4
    - lxml==4.9.2
    - matplotlib==3.7.1
    - numpy==1.24.3
    - openpyxl==3.1.2
    - packaging==23.1
    - pandas==2.0.1
    - pandas-datareader==0.10.0
    - Pillow==9.5.0
    - pyparsing==3.0.9
    - python-dateutil==2.8.2
    - pytz==2023.3
    - requests==2.31.0
    - scikit-learn==1.2.2
    - scipy==1.10.1
    - six==1.16.0
    - threadpoolctl==3.1.0
    - tzdata==2023.3
    - urllib3==2.0.2
    - xlrd==2.0.1
    - zipp==3.15.0
    - zope.interface==6.0

### Memory and Runtime Requirements

Approximate time needed to reproduce the analyses on a standard 2022 desktop machine: <20 minutes. The code was last run both on a Macbook laptop with an 8-core M1 Pro chip, 16GB RAM, Mac OS version 12.3 and a Dell XPS laptop with a core TM i7-1280P, 32 GB RAM, 12TH Gen Intel (R).

## Description of code and instructions for replication

### General Instructions

To reproduce all the figures and tables in the main paper and the online appendix, the steps below were taken in order. All necessary Python, Stata and Matlab packages should be installed prior to running scripts (see section Software Requirements for a complete list of packages).

1. From the code/Stata folder, run ImportData.do
   1. Note: file path must be changed to the user’s replication\data\input\_data\public directory at the beginning of the script
2. From the code/Stata folder, run empirical\_eq\_simulations.do
   1. Note: file path must be changed to the user’s replication\data directory at the beginning of the script. The local variable outfilename, found at the beginning of the script, should also be changed.
3. From the code/MATLAB folder, run the files simulation\_full\_irfs.m, run\_conditional\_forecast.m, run\_dynamic\_simuls.m, and Simplemodel.m
   1. Note that conditional\_forecast.m and dynamic\_simul.m define functions, which are then run by run\_conditional\_forecast.m and run\_dynamic\_simuls.m
   2. Simplemodel.m runs the dynare scripts defined in Simple\_Eq\_simulations\_weak.mod and Simple\_Eq\_simulations\_strong.mod and produces the file simple\_model\_irf\_results.xls, found in the data/output\_data folder. Note that to run .mod files separately, simply type into MATLAB’s command window “dynare insert\_file\_name\_here.mod”
   3. If the user decides to change the estimated coefficients from what they were originally defined as in the replication package, they will need to update line 110 of conditional\_forecast.m manually to run accurate forecasts. See the “MATLAB code” section below for further details.
   4. If the user is receiving an “Error using print. PNG library failed: Could not open file” error, please pause syncing on all cloud services (e.g. Dropbox and OneDrive) and try again.
4. Run get\_commmodity\_data.py in the code/Python folder
5. Run the rest of the Python scripts in the code/Python folder, namely figure\_1\_2\_simple\_model\_graphs.py , figure\_3\_7\_8\_9\_10\_11\_empirical\_model\_graphs.py, figure\_4\_5\_pca\_graphs.py, figure\_6\_car\_graphs.py, figure\_12\_13\_decomposition.py, and figure\_14\_cond\_forecast\_graph.py.
   1. Note: universal\_params.py does not need to be run or changed. The pickled file is already included; the file simply defines some variables that are helpful to use throughout the Python scripts.
6. To produce the appendix tables, run appendix\_tableA2.do and appendix\_tableA3.do from the code/Stata folder.

### Stata Code

ImportData.do uses the data in data/input\_data/public/InfoData.xlsx and formats it to be used in empirical\_eq\_simulations.do. It outputs the file InfoData.dta, found in the data/intermediate\_data folder.

Empirical\_eq\_simulations.do takes the file InfoData.dta from the data/intermediate\_data folder and runs regressions corresponding to the four equations (price equation, wage equation, 1-year inflation expectation equation, and 10-year inflation expectations equation). It produces the file eq\_coefficients.xlsx, which contains the results of the regressions, which are later used in future MATLAB scripts. Empirical\_eq\_simulations.do also produces equation\_simulations\_data.xls, found in the data/intermediate\_data folder.

### MATLAB code

The MATLAB scripts can be run in any order; further descriptions of variables and options are available within each script.

Simulation\_full\_irfs.m reads equation\_simulations\_data.xls as well as eq\_coefficients.xlsx to run IRFs. The program runs through multiple scenarios where different shocks are introduced to the model. The program is capable of producing MATLAB graphs, which are not formatted as nicely as the Python graphs, but will showcase a much larger set of data. The option to update graphs can be turned off to decrease runtime (note that graphs will not be updated).

Run\_conditional\_forecast.m recursively calls the function defined in conditional\_forecast.m for various specifications. These specifications can easily be changed by the user for alternate scenarios. Run\_conditional\_forecast.m and conditional\_forecast.m each reads equation\_simulations\_data.xls as well as eq\_coefficients.xlsx to run conditional forecasts. Similar to simulation\_full\_irfs, this program is also capable of producing MATLAB graphs.

1. Note: if the estimated coefficients change from what was originally included in the replication package, the constant for the wage equation (found in line 110) in run\_conditional\_forecast.m must be manually changed to run accurate forecasts. Adjusting the constant in the wage equation is needed to allow for v/u\* to be set to a desired level (in this case, 1.2); we make this adjustment since the implied v/u\* from estimated parameters reflect v/u\* over the entire estimation period. The new estimated wage equation constant can easily be computed by opening the file wage\_constant\_calculations.xlsx, found in the data/intermediate\_data folder, and changing the constants found on each sheet to the updated coefficients. Further changes to the wage constant can be made for different assumptions of magpty, shortage, and vu\_star. Please see the readme tab in the wage\_constant\_calculations.xlsx file for additional information.

Run\_dynamic\_simuls.m was used to create the data needed for the decomposition of sources of inflation. Similar to run\_conditional\_forecast.m, run\_dynamic\_simuls.m recursively calls the function defined in dynamic\_simul.m to run the full model under various circumstances. This program is also capable of producing MATLAB graphs, and this option can be turned off to decrease runtime.

### Python code

As outlined in the Software Requirements section, a set of packages must be installed prior to running Python scripts. The packages can be easily installed by navigating to the code/Python folder and running pip install -r requirements.txt.

Get\_commodity\_data.py was used to calculate the first principal component of a basket of 19 commodities. These commodities included: crude oil (WTI), heating oil, natural gas, corn, soybeans, live cattle, gold, aluminum, copper, sugar, cotton, cocoa, coffee, nickel, wheat, lean hogs, orange juice, silver, RBOB gasoline, unleaded gasoline, iron, lead, and zinc. Get\_commodity\_data.py reads confidential data from Bloomberg and Haver, stored in the data/input\_data/confidential folder in the files Bloomberg\_Commodity\_Data.xlsx and SP\_GSCI.xlsx. It outputs the file Bloomberg\_Commodity\_Data\_Cleaned\_Quarterly.xlsx. Please see instructions in the Data Availability section on how to retrieve Bloomberg and Haver data. Commodity data is only needed for Figures 4 and 5 and is not needed to run any MATLAB simulations.

Figure\_1\_2\_simple\_model\_graphs.py takes data produced from dynare mod file Simple\_Eq\_simulations, stored in simple\_model\_irf\_results.xlsx

Figure\_3\_7\_8\_9\_10\_11\_empirical\_model\_graphs.py takes data from eq\_simulation\_data.xls containing results from the regressions, predictions and formatted series produced by Stata. The first part of the script produces Figure 3, 7, 8 and 9. The script then pulls data from empirical\_model\_irf\_results.xls containing results produced by MATLAB. The last part of the script produces Figure 10 and 11 of the paper.

Figure\_4\_5\_pca\_graphs.py takes data from Bloomberg\_Commodity\_Data\_Cleaned\_Quarterly.xlsx, found in the data/output\_data folder, and ﻿eq\_simulations\_data, found in the data/intermediate\_data folder, to produce Figures 4 and 5.

Figure\_6\_car\_graphs.py takes data from FRED using the FRED API as well as from data from google trends, which is contained in the googletrends\_quarterly.xls file in the data/input\_data/public folder. This script produces Figure 6 of the paper.

Figure\_12\_13\_decomposition.py takes data from conditional\_forecast\_output.xls, found in the data/output\_data folder, and produces Figures 12 and 13, found in the figures/pngs folder.

Figure\_14\_cond\_forecast\_graph.py takes data from conditional\_forecast\_output.xls, found in the data/output\_data folder, and creates Figure 14, found in the figures/pngs folder.

Universal\_params.py creates the file universal\_params, which contains a dictionary of useful parameters used throughout other Python scripts.