Replication Archive Readme File

Measuring Economic Growth With A Fully Identified Three-Signal Model

Andrea Civelli* Arya Gaduh† Ahmed Sadek Yousuf ‡

University of Arkansas *University of Arkansas, NBER* *North South University*

Abstract

This brief document provides a guidance through the replication package of "Measuring Economic Growth With A Fully Identified Three-Signal Model" published in *The Review of Economics and Statistics*.

Keywords: Remote sensing; Nighttime lights; Urban expansion; Economic activity measurement.
JEL Classification: E01, O11, O47, O57.

*University of Arkansas, Walton College of Business, Department of Economics, Business Building 402, Fayetteville, AR. Email: andrea.civelli@gmail.com.

†University of Arkansas, Walton College of Business, Department of Economics, Business Building 402, Fayetteville, AR. Email: agaduh@walton.uark.edu.

‡North South University, Department of Economics. Email: ahmed.sadek87@gmail.com.

1 Overview

The code in this replication package for *"Measuring Economic Growth With A Fully Identified Three-Signal Mode"* provides routines for building the analysis dataset and producing the estimation results we present in the paper. There are three separate pieces of code contained in this replication package, which correspond to the three main folders of the package:

1. Code that uses the analysis dataset to produce the results presented in our paper – folder `1 Paper Output`.
2. Code to create our analysis datasets by merging different sources of data – folder `2 Main Dataset`.
3. Code to clean and process the raw data – folder `3 Raw Data Processing`.

To start, the replicator must unzip the replication package file to the desired location, making sure that the structure of these three folders in the package is maintained.

All data sources used in this paper are publicly accessible from online resources. Nevertheless, we also supply the raw data used to obtain the final analysis dataset, as well as the codes necessary to clean and pre-process the data. Note that some of the procedures for the dataset elaboration might be difficult to follow exactly, especially due to precise software requirements and configurations for ArcGIS and Python. We report below the setup details of the software and hardware we used.

2 Data Availability

The raw data we work with are publically available and can be freely downloaded from online sources.

Nightlights: Both the baseline DMSP nightlights data for the period 1992-2013 and the VIIRS-based extension through 2019 can be found on the Earth Observation Group website here:
https://eogdata.mines.edu/products/dmsp/.

Urban Land Cover: The land cover data are obtained from the European Space Agency Climate Change Initiative (ESA-CCI). The CCI website provides the land cover maps already in .tif format for the period 1992-2015 here:
https://maps.elie.ucl.ac.be/CCI/viewer/download.php#ftp_dwl.
Consistent data for the period after 2015 are available from the Copernicus Climate Change Service (CS3) website here:
https://cds.climate.copernicus.eu/cdsapp#!/dataset/satellite-land-cover?tab=form.

- Note: a simple registration to the two service is required to access and download the dataset.

Nitrogen Dioxide The NO_2 data can be found on the ESA Tropospheric Emission Monitoring Internet Service here:
https://www.temis.nl/airpollution/no2col/no2regioomimonth_qa.php.

GDP: The 2023 World Development Indicators (WDI) dataset of the World Bank provides GDP data and is available here:
https://databank.worldbank.org/World-GDP/id/acce53b.

Statistical Capacity Score: The statistical capacity score index is derived from the 2010 World Bank's Statistical Capacity Indicators from 2010 available here:
https://databank.worldbank.org/-2010-Statistical-Capacity-Score-(Overall-Average)/id/ef8b267e.

Administrative Boundaries: The national administrative boundaries shapefiles are provided by the Database of Global Administrative Areas (GADM) website, available here:
https://https://gadm.org/download_world36.html.
Version 3.6 is used in the paper.

3 Running the Main Analysis

In this section, we describe the code for conducting the empirical analysis that we present in our paper. This analysis can be conducted directly from the main dataset that we supply, without necessarily re-processing the entire dataset first. This `.dta` file is found in `.../2 Main Dataset/tostata/paper_dataset.dta`.

A main `.do` file runs all of the code to generate the tables and figures in the main body of the paper and the online Appendix – 2 tables and 3 figures in the paper; 5 tables and 2 figures in the Appendix.

Computational and Software Requirements. The code was last run on an Intel Core i7-10700 CPU - 2.90GHz computer with 16GB-memory and OS Windows 10 - version 22H2. We recommend using Stata/MP 18 (4 core) for replicating this archive (although versions as low as 15 also seem to do fine). The replicator should expect this code to run for less than 5 min for analysis in Stata/MP 18 (4 core).

Folder Information. This replication folder `.../1 Paper Output/` includes the following sub-directories:

- `/stata_do/`: containing all `.do` files needed for producing tables and figures
- `/stata_dta/`: an empty folder for storing the intermediate `.dta` files generated by the analysis

Table 1: Program and Table/Figure List

Figure/Table #	Program	Output file(s)	Notes
Table 1	REStat_PaperOutput.do	table1.tex	
Table 2	REStat_PaperOutput.do	table2.tex	
Table C.1 – App.	REStat_PaperOutput.do	tableC1PanelA.tex;tableC1PanelB.tex	
Table C.2 – App.	REStat_PaperOutput_0519.do	TableC2.tex	
Table C.3 – App.	REStat_PaperOutput_0519.do	tableC3PanelA.tex;tableC3PanelB.tex; tableC3PanelC.tex	
Table C.4 – App.	REStat_PaperOutput.do	TableC4.dta	dta file to export to LaTex
Table C.5 – App.	REStat_PaperOutput.do	TableC5.dta	dta file to export to LaTex
Figure 1	REStat_ScatterFigs.do	Figure1a.png;Figure1b.png	
Figure 2	REStat_PaperOutput.do	Figure2.png	
Figure 3	REStat_Africa_VerificationEx.do.do	Figure3.png	
Figure C.1 – App.	REStat_PaperOutput.do	FigureC1.png	
Figure C.2 – App.	REStat_ScatterFigs.do	tableC2a.tex;tableC2b.tex; tableC2c.tex	

Notes: All .do files are stored in the /stata_do folder.

- /figures/: an empty folder for storing the generated figures
- /tables/: an empty folder for storing the generated tables

Replication Instructions.

- The main do file for running the analysis can be found here:
  .../1 Paper Output/stata_do/REStat_Master.do

  This file calls on several subsequent .do files, also stored in the same folder. Before running the REStat_Master.do file, the replicator must change the global variable user defined on line 2 to reflect their local directory structure.

- For our main tables and figures, a detailed list of the individual, table-specific .do files can be found in Table 1.

- After the Stata routines are finished, a file in the repository called
  .../1 Paper Output/compile_tabfigs.tex
  can be compiled to reproduce the tables and figures in our paper and online Appendix, except for Appendix Tables C4 and 5 (i.e., tables presenting the samples of countries in the analysis and their statistical capacity which are simply reported in .dta format).

4 Merging Different Data Sources and Creating Analysis Dataset

In this section, we describe the code for building the analysis dataset that we use in our paper. A main `.do` file, `REStat_BuiltUp.do`, runs all of the processes to generate the dataset of the paper by merging different sources. This script uses a set of Excel tables with the re-elaborated raw data produced by GIS in the data cleaning stage of the replication. The script can be run without necessarily re-processing the raw data since we provide all the inputs for this stage of the replication in the folder `.../3 Raw Data Processing/tostata`.

Computational and Software Requirements. The code was last run on an Intel Core i7-10700 CPU - 2.90GHz computer with 16GB-memory and OS Windows 10 - version 22H2. We recommend using Stata/MP 18 (4 core) for replicating this archive (although versions as low as 15 also seem to do fine). The replicator should expect this code to run for less than 2 min in Stata/MP 18 (4 core).

File Information. This replication folder `.../2 Main Dataset/` includes the following sub-directories:

- `/stata_do/`: containing the `.do` file needed for elaboration and merging
- `/stata_dta/`: containing all `.dta` files needed for processing. Intermediate `.dta` files generated by the process are also saved in this folder
- `/tostata/`: the folder for storing the final analysis dataset of the paper
  - Note that the final dataset `paper_dataset.dta` is pre-saved in this folder for the users who want to run the analysis codes without reprocessing the raw data.

Replication Instructions. The main `.do` file for running the elaboration and merging of the dataset can be found here:

`.../2 Main Dataset/stata_do/REStat_BuiltUp.do`

Before running the `REStat_BuiltUp.do` file, the replicator must change the global variable `user` defined on line 2 to reflect their local directory structure.

5 Raw Data Elaboration

In this section, we describe the procedure for the cleaning and elaboration of the raw data, based on codes written in Python for ArcGIS - ArcMap. A main `.do` file then runs all of the processes to generate the analysis datasets from the initial data sources.

5.1 Software Requirements

The setup we rely on is as follows:

- ESRI ArcMap by ArcGIS Desktop 10.8.2
  - A separate license is required for the Spatial Analyst package.
  - Installation of Background Geoprocessing (64-bit) on top of ArcGIS Desktop could be helpful to perform operations with large amounts of RAM on large files in this section that may have otherwise failed in a 32-bit environment – for example, with `pixresizer_lights.py` in **Part I:** of the code sequence.
  - *Note:* ESRI is in the process of replacing ArcMap with ArcGIS Pro, with a final transition stage set for 2026. Although most of the processes in ArcMap should still work in ArcGIS PRO, some of the codes might require some adjustments.
- Python 2.7.18.4
  - Python 2.7.18.4 is integrated in the default installation of ArcGIS Desktop 10.8.2.
  - *Note:* Each ArcGIS Desktop version supports a different Python 2.7 deployment. This may affect the way Python deploys the `arcpy` package used in the code. Although the scripts are generally compatible with other versions of ArcGIS Desktop, users must be aware that possible small differences in the final output could result if versions of ArcGIS Desktop other than 10.8.2 are used.
- PythonWin32 (by Mark Hammond) - release 214 as Python GUI:
  - Although the procedure could be launched from the ArcGIS IDLE as well, the PythonWin GUI provides a more flexible development environment for this multi-layer script.
  - Instructions for the installation of the GUI are provided below.
- Stata/MP 18 (4 core)
- Matlab 2021b (or newer)

5.2 Installing PythonWin32

The scripts for the GIS processing code are developed in PythonWin32 for ArcMap. Release 214 of PythonWin32 is used since it correctly integrates with the underlying Python 2.7 installation used by ArcMap 10.8.2. This is an older release

of PythonWin32 and the source file `pywin32-214.win32-py2.7.exe` is directly provided in the replication package.

The installation is quite straightforward. At installation, ArcGIS 10.8.2 creates the folder `\Python27\ArcGIS10.8` in the home directory of the local disk of the machine where it is installed. This is the folder where PythonWin32 must be installed. Launch the PythonWin32's installation by double clicking on the `pywin32` execution file, and follow the Setup Wizard instructions as necessary.

5.3 Memory and Runtime Requirements

The code was last run on an Intel Core i7-10700 CPU - 2.90GHz computer with 16GB-memory and OS Windows 10 - version 22H2. The user should expect the overall script to take between 10 and 15 hours to complete.

5.4 File Information.

This replication folder `.../3 Raw Data Processing/` includes the following sub-directories:

- Three folders containing the source files and corresponding `.py` scripts needed for the processing of the geo-referenced datasets:
  - `/LC Urban/` for the urban cover variable
  - `/Nightlights/` for the luminosity variable
  - `/NO2 OMI/` for the NO_2 pollution variable
  - These folders also contains the `/scalingmask.tif` used to adjust the data rasters for the earth curvature
- `/WDI GDP/`: containing the GDP data and the `.do` file needed to link them to the GIS component of the dataset
- `/Shapefiles/`: containing the shapefile of the administrative boundaries of the World countries and other GIS work files
  - This folder also contains the `/mergedflares.shp` used to adjust the nightlight rasters for gas flares
- `/tostata/`: the folder for storing the final processed data used as an input to build the dataset of the paper
  - Note that these final excel tables are pre-saved in this folder for the users who want to run the analysis codes without reprocessing the raw data.

5.5 Replication Instructions

1. In Stata, run the script
   `.../3 Raw Data Processing/WDI GDP/REStat_GDPcompile.do`
   to map the GDP data into the geo-referenced data space defined by the World countries shapefile.

   (a) At the end of this step, the replicator must generate the shapefile `World_All_Countries_GADM_toGIS.shp` directly in ArcMap to be used as reference shapefile for processing the geo-referenced raw data. The following steps are required:

      i. In ArcMap, load the shapefile `.../3 Raw Data Processing/Shapefiles/gadm36_0.shp` and the CSV file `.../3 Raw Data Processing/WDI GDP/World_All_Countries_GIStable 36.csv`

      ii. Right-click on the shapefile → Joins and Relates → Join, with parameters:
      - join field == GID_0
      - join on == CountryCode
      - keep only matching records

      iii. Right-click again on shapefile → Data → Export Data, with parameters:
      - all features
      - use coordinate system of the layer's source data
      - save to location
        `.../3 Raw Data Processing/Shapefiles/World_All_Countries_GADM_toGIS.shp`

   (b) The ArcGIS `mxd` file
   `.../3 Raw Data Processing/Shapefiles/Create_toGIS_shapefile.mxd`
   collects these files embedding the Join & Save steps above.

   (c) Alternatively, if the replicator has access to ArcGIS Pro, the code
   `.../3 Raw Data Processing/Shapefiles/tostata_shape.py`
   can be used to generate the same output. The replicator must adjust the directory path in this script on line 14 file before running it.

2. In PythonWin (see in Section 5.2 for installation), launch the script
   `.../3 Raw Data Processing/mainchain.py`
   to run all the sub-routines necessary to complete the processing of nightlight luminosity, urban cover, and NO_2 pollution. More details about these sub-routines are provided in Section 5.6.

(a) The replicator must specify the directory path where the replication package has been copied before running `mainchain.py`. This is done at the beginning of the script within the definition of the function `reroot()`.

3. **Note:** The rasters `/scalingmask.tif` with the factors to adjust datasets for the earth curvature are provided as part of the input files in each of the working folders. However, they can also be generated in Matlab by running the script `.../fixing_pixels.m` in each folder.

   An additional step is required for the urban land cover dataset, to facilitate the creation of the mask given the large size of the files:

   - In this case, four rasters are generated by `.../LC Urban/fixing_pixels.m`, with name `/scalingmask1.tif` - `/scalingmask4.tif`. These files are also provided in the zip file `.../3 Raw Data Processing/LC Urban/scalingmask_urban.zip`.
   - The tool `Mosaic To New Raster` from the Data Management toolbox in ArcMap is used to merge the four rasters into a single `/scalingmask.tif`

5.6 Python Code Details

The python script `mainchain.py` calls and executes all the sub-routines necessary to obtain the GIS-based data necessary to build the final paper dataset:

- **Script:** `mainchain.py`
- **Location:** `.../3 Raw Data Processing/`
- See Table 2 for input/output information.

The procedure in `mainchain.py` consists of three main parts.

- **Part I:** The nightlight luminosity variable is created in `.../3 Raw Data Processing/Nightlights`. The routines used here are:
  - `makelights.py` creates the yearly data raster files as an average of the raw data raster files available from different satellites, joining DMSP and DVNL data
  - `adjust_flare.py` creates a raster mask of the areas with significant natural gas flares from extraction activities
  - `pixresizer_lights.py` adjusts the raster datasets for gas flares and earth curvature
  - `statacsv_lights.py` uses zonal statistics to calculate the nightlight luminosity variable by country

- *Note:* The input shapefile with the gas flare mask, `mergedflares.shp`, is obtained by merging the national flare masks provided by NOAA (National Oceanic and Atmospheric Administration) here: https://ngdc.noaa.gov/eog/interest/gas_flares_countries_shapefiles.html.
  The script `merger_flares.py` provides a simple description of the steps involved in the merger.

- **Part II:** The urban land cover variable is created in `.../3 Raw Data Processing/LC Urban`. The routines used here are:
  - `makeurban.py` extracts the urban component from the raw land-cover raster files by year
  - `pixresizer_urban.py` adjusts the raster dataset for earth curvature
  - `statacsv_urban.py` uses zonal statistics to calculate the urban variable by country
  - If raw data is downloaded from the source in .nc format, it must be converted to .tif format to be used in the rest of the process. The script `netCDF2tif.py` provides a template for the conversion, using year 2016 as an example.

- **Part III:** The NO_2 pollution variable is created in `.../3 Raw Data Processing/NO2 OMI`. The routines used here are:
  - `months2years.py` creates the yearly data raster files as an average of the raw monthly data raster files
  - `makeno2.py` adjusts the yearly rasters for negative observations and defines their projection
  - `pixresizer_no2.py` adjusts the raster dataset for earth curvature
  - `statacsv_no2.py` uses zonal statistics to calculate the NO_2 pollution variable by country
  - The script `download_unzip.py` shows how to download via API and unzip the raw data files from the source.

Table 2: Inputs and outputs in `mainchain.py`

Part I: Nightlight Luminosity `.../3 Raw Data Processing/Nightlights`			
Input Files	**Codes**	**Output Files** `.../3 Raw Data Processing`	**Notes**

`/FXXYYYY.tif`
`/DVNL_YYYY.tif`
`...Processing/Shapefiles/mergedflares.shp`
`/scalingmask.tif`
	`makelights.py`
	`adjust_flare.py`
	`pixresizer_lights.py`
	`statacsv_lights.py`
		`/tostata/tostata_DMSPintYYYY.xls`
		`/tostata/tostata_denom_DMSPint.xls`
			XX indicates the satellite number
			YYYY indicates the year

Part II: Urban Land Cover `.../3 Raw Data Processing/LC Urban`			
Input Files	**Codes**	**Output Files** `.../3 Raw Data Processing`	**Notes**

`/ESACCI-LC-L4-LCCS-Map-300m-P1Y-YYYY-vX.X.X.tif`
`/scalingmask.tif`
	`makeurban.py`
	`pixresizer_urban.py`
	`statacsv_urban.py`
		`/tostata/tostata_URBANesaYYYY.xls`
		`/tostata/tostata_denom_URBANesa.xls`
			X.X.X indicates the dataset version
			YYYY indicates the year

Part III: NO_2 **Pollution** `.../3 Raw Data Processing/NO2 OMI`			
Input Files	**Codes**	**Output Files** `.../3 Raw Data Processing`	**Notes**

`/OMI_YYYY_XX.grd`
`/scalingmask.tif`
	`months2years.py`
	`makeno2.py`
	`pixresizer_no2.py`
	`statacsv_no2.py`
		`/tostata/tostata_NO2omiYYYY.xls`
		`/tostata/tostata_denom_NO2omi.xls`
			XX indicates the month of the year
			YYYY indicates the year