

Lecture 2

Reproducibility

Pierre Biscaye
Université Clermont Auvergne

Data Science for Economics

Note: Materials for this lecture are drawn from Ted Miguel's Development Economics course at UC Berkeley

Agenda

1. Overview of reproducibility and transparency
2. Organizing files
3. Coding transparency and portability
4. Writing code in Python
5. More Python basics

Key references on research reproducibility

- Casey, Glennerster, and Miguel. (2012). “Reshaping Institutions: Evidence on Aid Impacts Using a Pre-analysis Plan”, *Quarterly Journal of Economics*, 127(4), 1755-1812.
- Miguel et al. (2014). “Promoting Transparency in Social Science Research”, *Science*, 10.1126/science.1245317.
- Christensen and Miguel. (2018). “Transparency, Reproducibility, and the Credibility of Economics Research”, *Journal of Economic Literature*, 56(3), 920-980.
- Ferguson et al. (2023). “Survey of open science practices and attitudes in the social sciences”, *Nature Communications*, 14.
- Christensen, Freese, and Miguel. (2019). *Transparent and Reproducible Social Science Research: How to Do Open Science*, University of California Press.

Threats to validity of research

- **Fraud:** undermines public trust in science
 - Open data and code can help uncover
- **Publication bias:** missing studies/ “file-drawer” problem
 - Wasted research effort, misguided policy decisions
 - Author manipulation/“p-hacking”
 - Pre-registration as a counter
- **Failure to replicate:** within study (reproduction) and across settings (replication)
 - Increasing journal data posting requirements
 - Difficulty of getting funding or publishing replications of studies

What do transparency and reproducibility mean in economics research?

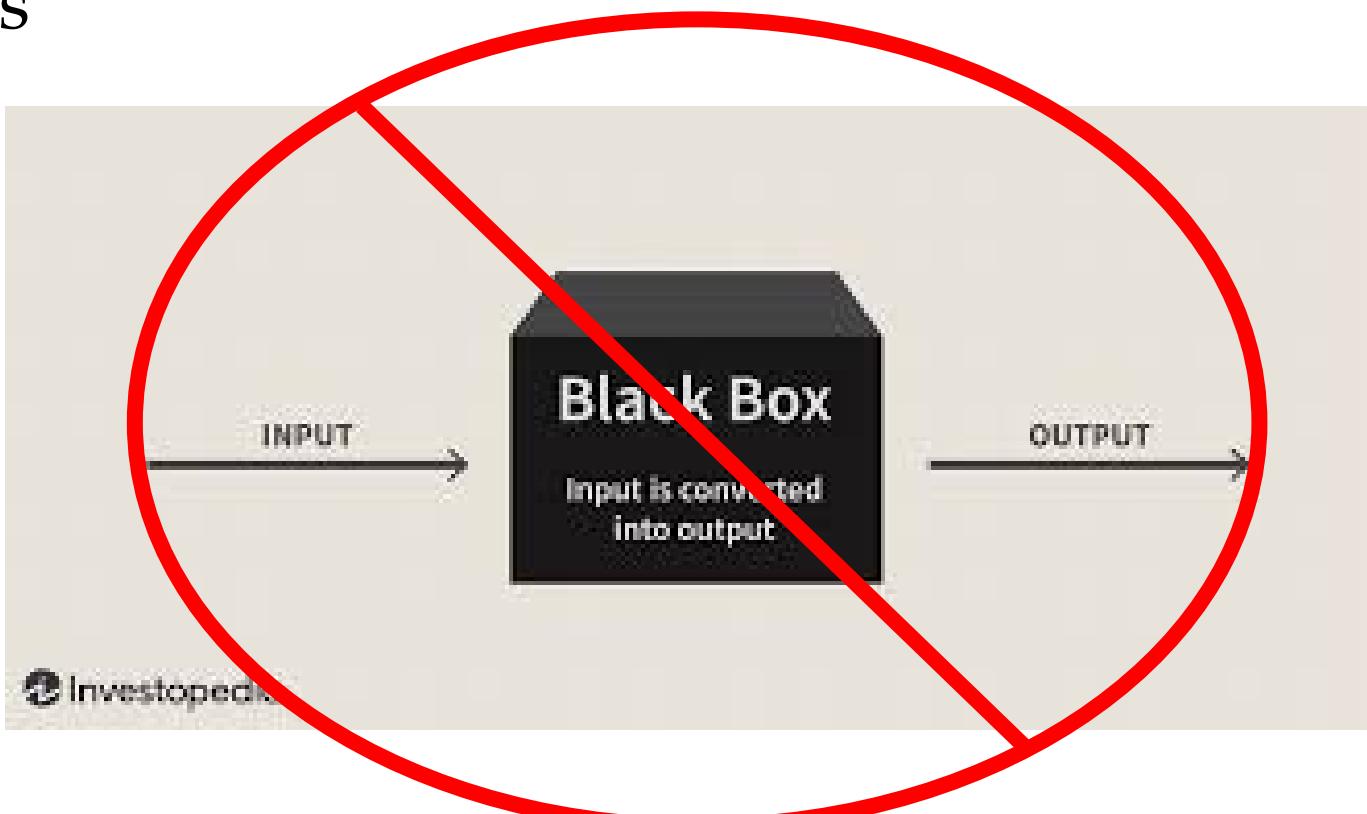
- **Transparency:**
 - Ensuring that all data, methods, and analyses are openly shared and clearly documented.
 - Allows others to understand and evaluate the research process, and to test for investigator bias or sensitivity to alternative methods.
- **Reproducibility:**
 - The ability of others to reproduce the results of a study using the original data and code provided by the researchers.
 - Critical for validating findings, building trust, and advancing knowledge.

Key principles of reproducibility

- **Accessible Data:** Provide well-documented, cleaned, and publicly available datasets (where ethically and legally possible).
- **Open Code:** Share analysis scripts and code in a version-controlled repository (e.g., GitHub) with clear instructions for execution.
- **Clear Methodology:** Document all steps in data collection, preprocessing, and analysis to ensure clarity and consistency.
- **Replication Workflow:** Design workflows that allow others to re-run analyses seamlessly, ensuring compatibility across systems and tools.

Takeaways for this course

- The quality and clarity of the research process matters
- Focus of this lecture: organizing and documenting research methods



Organizing files

The first step to a clear research process is file organization. There should be a clear structure to your folder hierarchy and file names.

```
README.pdf  
data/  
    raw/  
        cps0001.dat  
analysis/  
    combined_data.dta  
    combined_data.csv  
    combined_data_codebook.pdf  
code/  
    01_create/  
        01_readcps.R  
        02_readfred.R  
    02_analysis/  
        01_table1-5.R  
        02_figures1-4.R  
results/  
    table1.tex  
    table2.tex  
    ...  
    figure1.pdf  
    figure2.pdf
```

[Source](#)

Example: Jigawa Floods Project

Basic structure:

1. Code
 1. High-frequency checks for data collection
 2. Analysis: separate scripts for different tasks
2. Data
 1. Raw
 2. Clean
3. Output
 1. Figures
 2. Maps
 3. Tables

Example: Locusts and Conflict Project

- Work in progress
- Mix of archived and active files
- Clear overarching structure
- Clean replication package

Example: Folder of data sources

1. Sub-folders for main data sources/data types
2. Clear organization within folders
3. Example: LSMS-ISA
 1. Country sub-folders
 2. Year/round sub-folders
 3. Data, Questionnaires, Resources sub-folders
 4. Zipped raw data and unzipped folders
 1. Always keep a copy of the original data as a backup!
4. Readme documents
 - <https://databank.illinois.edu/datasets>IDB-1107366>
Retrieved 2/14/23
 - 1. Notes on where/when data were accessed
 - 2. Example: ARES data
 1. ARES, crop-specific exposure to temperature and moisture shocks
 - a. 0.25 degree cells by year by crop
 - b. 1961-2014
 - c. /Users/pierrebiscaye/Dropbox/Data/Spatial/ARES
 - d. .nc files with crop layers

File organization in this class

- Folders for each section
- Subfolders for data, images
- Separate Jupyter Notebooks for different topics
- Clear file naming conventions

Transparent and reproducible code

1. Comment and document thoroughly
2. Use modular design
3. Adopt consistent naming conventions
4. Version control
5. Ensure code is portable

This is important for both your future self and for potential collaborators and reviewers!

Looking back at the code you wrote last month...



1. Comment and document thoroughly

- Include clear comments in your code to explain the purpose of each section and the logic behind key steps.
- Best practice: Use a README file to provide an overview of the project, dependencies, and how to run the code.

Looking at code you wrote more than 6 months ago:



[Source](#)

Commented code

```
14 //  
15 // Prep spatial datasets  
16 //  
17  
18 * Locust Swarms  
19 {  
20 * Data from FAO Locust Hub, coordinates and dates of locust swarm observations globally from 1985-2023  
21 * Some additional detail included there but lots of missing data so focus just on swarm location and timing  
22 import delimited "$data/Locust Hub/Retrieved 2.13.23/Swarms.csv", encoding(UTF-8) clear  
23 drop if objectid==.  
24 drop if missing(x) | missing(y)  
25 destring x, replace  
26 destring y, replace  
27 gen double lat=round((y+0.125)*4)/4 - 0.125  
28 gen double lon=round((x+0.125)*4)/4 - 0.125  
29 gen date = date(substr(startdate,1,10), "YMD")  
30 format date %td  
31 gen year=year(date)  
32 gen month=month(date)  
33  
34 * Output for mapping  
35 preserve  
36 keep x y year  
37 export delimited "$clean/mapping_swarms.csv", replace  
38 restore  
39  
40 * Match to countries, identify countries in Africa and Arabian Peninsula with at least 10 swarms in analysis period  
41 1996-2018  
42 geoinpoly y x using "$data/Country boundaries/Country raw/UIA_World_Countries_Boundaries/WORLD_coor.dta"  
43 merge m:1 _ID using "$data/Country boundaries/Country raw/UIA_World_Countries_Boundaries/WORLD_data.dta", keep(1 3)  
44 gen swarm=1 if year>1995 & year<2019  
45 egen swarms_1996_2018=sum(swarm),by(COUNTRY)  
46 bys COUNTRY: gen first=_n==1  
47 br COUNTRY swarms_1996_2018 if swarms_1996_2018>10 & first==1  
48 * Well over 10: Algeria, Burkina Faso, Chad, Egypt, Eritrea, Ethiopia, Gambia, Guinea, Guinea-Bissau, Libya, Mali,  
49 Mauritania, Morocco/Western Sahara, Niger, Oman, Saudi Arabia, Senegal, Somalia, Sudan, Tunisia, Yemen  
50 * Cabo Verde 42, India 127, Iran 19, Israel 11, Kenya 22, Pakistan 108 also meet the swarm count criteria but outside  
area of interest or are borderline cases (Israel, Kenya)  
51  
52 * Set target geographic area to trim other datasets, based on countries to target  
53 drop if lat<-2.5  
54 drop if lat>37.5  
55 drop if lon<-17.5  
56 drop if lon>60.25
```

Data Readme

This document outlines the data sources used in this study.

A. DATA INCLUDED IN REPLICATION PACKAGE

1. Locust swarms data

- * Retrieved from FAO Locust Hub 2/13/2023 at <https://locust-hub-hqfao.hub.arcgis.com/search?collection=dataset>
- * These data are no longer publicly accessible, so I include the raw downloaded data in the package
- * The data includes coordinates and dates of locust swarm observations globally from 1985-2023
- * Some additional detail included there but lots of missing data so focus just on swarm location and timing

Raw data file: Swarms.csv

2. ACLED Conflict data

- * raw dataset is full conflict data downloaded from ACLED on 11/26/20, <https://acleddata.com/>
- * second datafile downloaded covers 02/11/20-02/03/23 and is merged with the first
- * the data records all conflicts and actors from all events going as far back as the database - 1997
- * selected geographies: East Africa, Middle Africa, Middle East, North Africa, West Africa, South Asia

Raw data files:

- * 1997-01-01-2020-11-26-Eastern_Africa-Middle_Africa-Middle_East-Northern_Africa-South_Asia-Western_Africa.csv
 - * 2020-02-11-2023-02-03-Eastern_Africa-Middle_Africa-Middle_East-Northern_Africa-South_Asia-Western_Africa.csv
-

3. Aggregated analysis data

- * Dataset at 0.25 degree grid cell by year level produced by aggregating swarm, conflict, and other data
- * Description of inputs is below

Processed data file: temp_2025.dta

B. INPUTS TO TEMP_2025.DTA

2. Use modular design

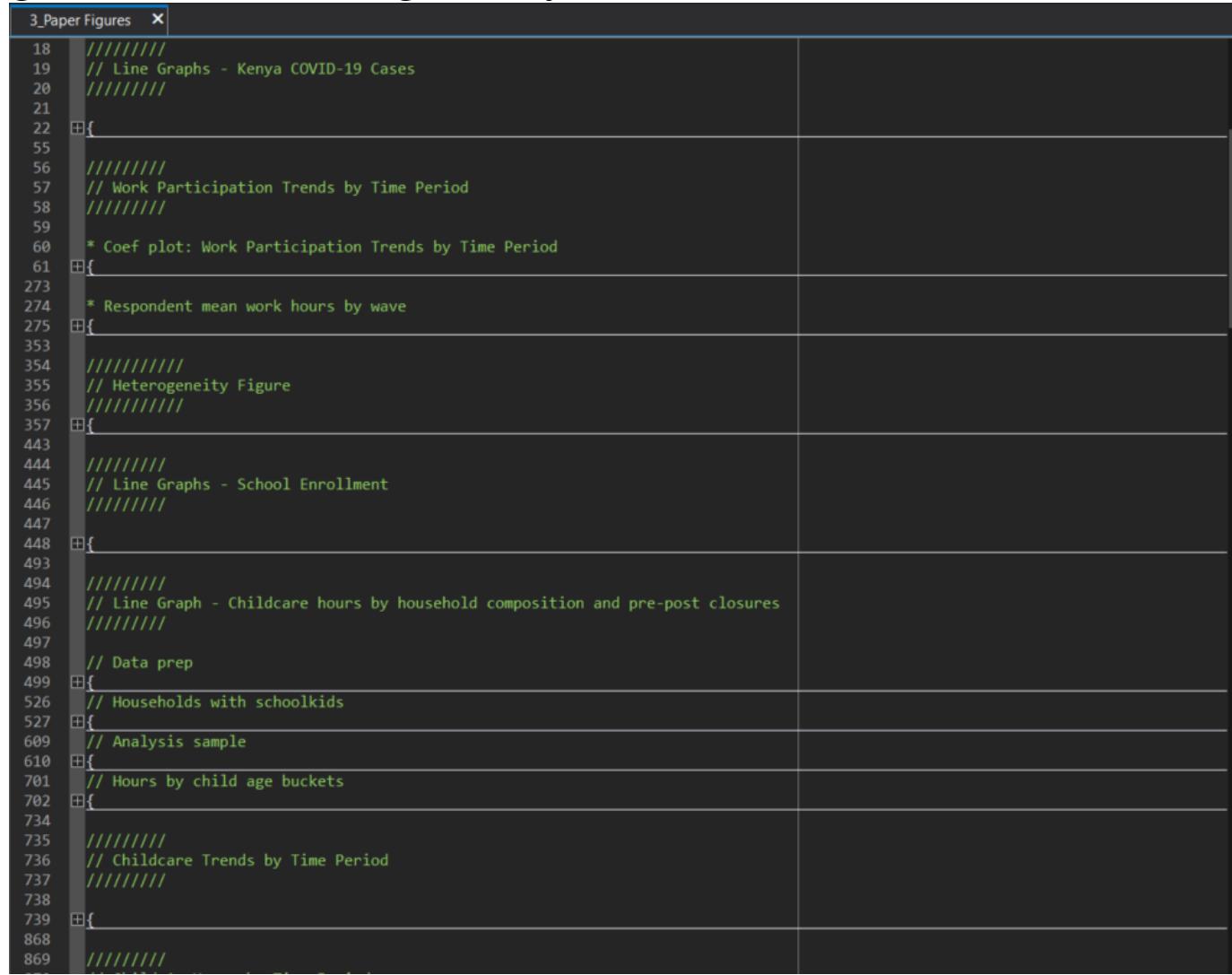
- Break the code into smaller, reusable functions or scripts to make it easier to debug, update, and understand. Separate data cleaning, analysis, and visualization steps logically.

Dropbox > Kenya Labor Supply > do files >			
	Nom	Modifié le	Type
	.ipynb_checkpoints	11/20/2024 1:16 PM	File folder
	Archive	11/20/2024 1:16 PM	File folder
	1_Data Merge	4/12/2024 12:16 AM	DO File
	2_Merged Data Prep	6/25/2024 1:19 AM	DO File
	3_Paper Figures	5/10/2024 7:56 AM	DO File
	4_Paper Regressions	9/13/2023 8:29 PM	DO File

All files / GridWatch Master Folder / Replication code	⚙️
Recents	Starred
Name ↑	
Archive	
cleaning	
.Rhistory	
Balance_reg_fe.do	
Balance_reg_iv.do	
Balance_reg.do	
copy_data.do	
figures.do	
MapSites_TC.R	
prep_survey_data.do	
reg_table.do	
tables.do	

Modular design within scripts

Use clear headings and labels to organize your code



The screenshot shows a Jupyter Notebook cell titled "3_Paper Figures". The code is organized into several sections, each starting with a series of five slashes (/////) followed by a descriptive comment. The sections include:

- // Line Graphs - Kenya COVID-19 Cases
- // Work Participation Trends by Time Period
- * Coef plot: Work Participation Trends by Time Period
- * Respondent mean work hours by wave
- // Heterogeneity Figure
- // Line Graphs - School Enrollment
- // Data prep
 - // Households with schoolkids
 - // Analysis sample
 - // Hours by child age buckets
- // Childcare Trends by Time Period

The code uses horizontal lines and indentation to separate different parts of the analysis.

Jupyter Notebook: modular by default

The screenshot shows a Jupyter Notebook interface with the following structure:

- Header:** jupyter Section1a_Jupyter Notebook Last Checkpoint: 16 days ago
File Edit View Run Kernel Settings Help
- Toolbar:** File, Edit, View, Run, Kernel, Settings, Help, +, X, □, C, ▶, Markdown, ▾
- Section 1: Practice**
 - []: # Please add a cell above here
 - []: # Please add a cell below here
 - []: # Please delete this cell
 - []: ### Please change this code cell to a Markdown cell
- Section 2: Please change this Markdown cell to a code cell**
 - []: # copy this cell and paste below
 - []: # cut this cell and paste it here
 - []: # Please run this cell
a = a+1 # Adding 1 to a
print(a)
 - []: # Please split this cell after this line
Please split this cell
 - []: Please toggle comment on this line
 - []: Please toggle comment on this line and this line

3. Adopt consistent naming conventions

- Use meaningful, consistent names for variables, functions, and files to enhance readability and reduce confusion.
- For example, use `clean_data()` instead of `cd()` for function names.

NAMING VARIABLES



Giving them meaningful names, according to their use.



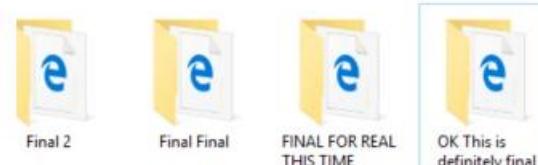
Giving them the most compact names possible, for less storage usage.



Giving them random names like "ahshjdn" or "yeetus".

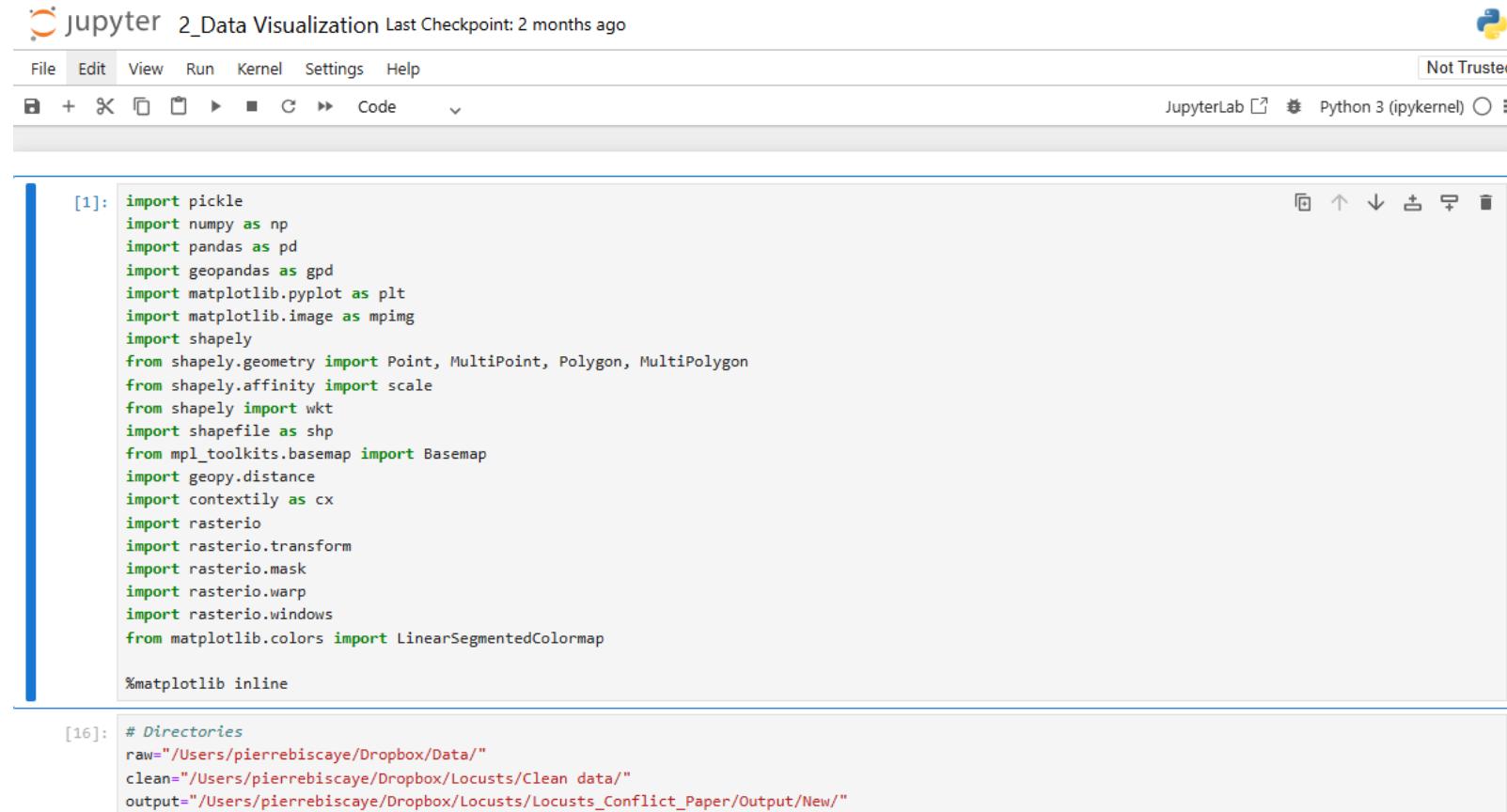
4. Version control

- Use version control systems to track changes in your code and collaborate efficiently.
- Commit changes with descriptive messages and maintain a well-organized repository structure.
- Gold standard: GitHub repository
- Minimum: clear file names indicating version history, well-structured archive folder
 - Back up your code, don't just always overwrite



5. Ensure code is portable

- Avoid hardcoding file paths or machine-specific dependencies.
- Use relative paths and specify software environments to ensure others can run the code seamlessly.
- Often useful to have a “`0_setup` script” with paths and packages to run first.



The screenshot shows a Jupyter Notebook interface titled "jupyter 2_Data Visualization Last Checkpoint: 2 months ago". The notebook has a "Not Trusted" status. The code cell [1] contains imports for various Python libraries including pickle, numpy, pandas, geopandas, matplotlib.pyplot, matplotlib.image, shapely, shapely.geometry, shapely.affinity, shapely.wkt, shapefile, mpl_toolkits.basemap, geopy.distance, contextily, rasterio, rasterio.transform, rasterio.mask, rasterio.warp, rasterio.windows, and matplotlib.colors. It also includes a %matplotlib inline command. Cell [16] contains code to define directory paths: raw="/Users/pierrebiscaye/Dropbox/Data/", clean="/Users/pierrebiscaye/Dropbox/Locusts/Clean data/", and output="/Users/pierrebiscaye/Dropbox/Locusts/Locusts_Conflict_Paper/Output/New/".

```
[1]: import pickle
import numpy as np
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import shapely
from shapely.geometry import Point, MultiPoint, Polygon, MultiPolygon
from shapely.affinity import scale
from shapely import wkt
import shapefile as shp
from mpl_toolkits.basemap import Basemap
import geopy.distance
import contextily as cx
import rasterio
import rasterio.transform
import rasterio.mask
import rasterio.warp
import rasterio.windows
from matplotlib.colors import LinearSegmentedColormap

%matplotlib inline

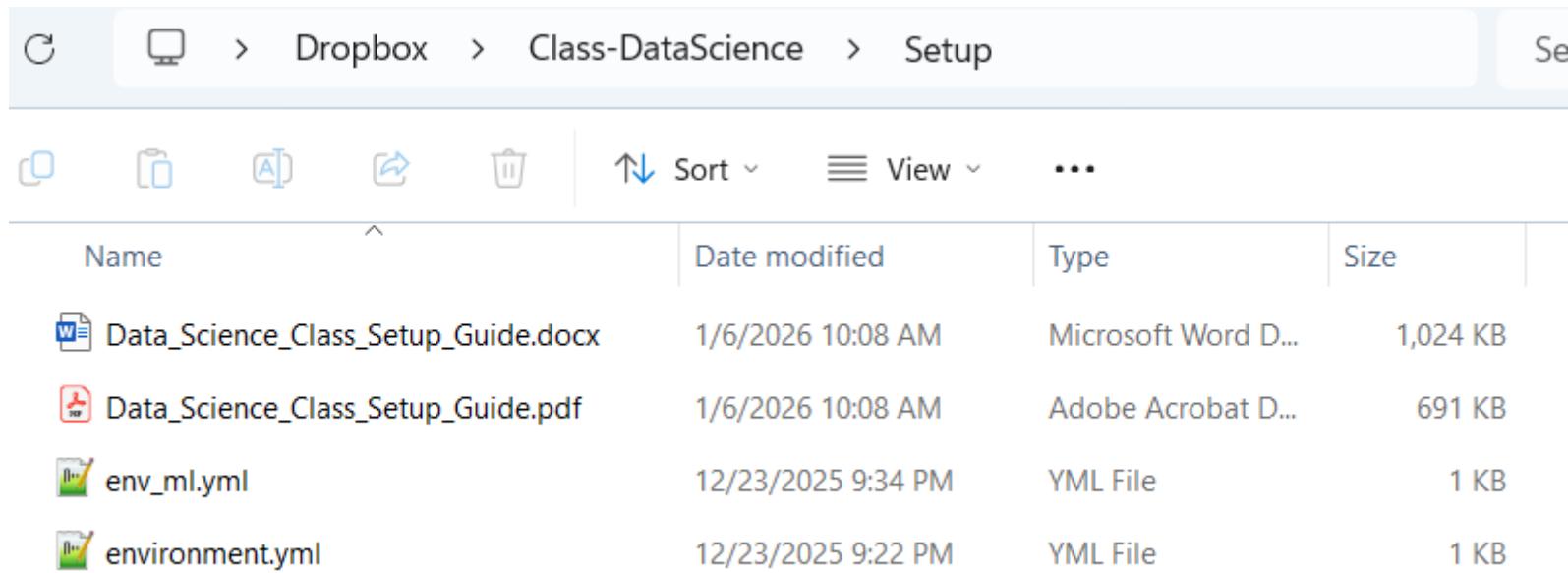
[16]: # Directories
raw="/Users/pierrebiscaye/Dropbox/Data/"
clean="/Users/pierrebiscaye/Dropbox/Locusts/Clean data/"
output="/Users/pierrebiscaye/Dropbox/Locusts/Locusts_Conflict_Paper/Output/New/"
```

Example: directory setting code file

```
Setup.do  X

1  clear all
2  set more off
3  set trace off
4  cap log close
5  set maxvar 10000
6  pause off
7
8  * Set paths
9  if `c(username)'=="pierrebiscaye{
10    global home "/Users/pierrebiscaye/Dropbox"
11  }
12  if `c(username)'=="pibiscay{
13    global home "C:\Users\pibiscay\Dropbox"
14  }
15
16  global rawghsp "$home/Data/LSMS-ISA/Nigeria"
17  global w1 "$rawghsp/Wave 1 (2010-11)/Data/NGA_2010_GHSP-W1_v03_M_STATA/All One Folder"
18  global w2 "$rawghsp/Wave 2 (2012-13)/Data/NGA_2012_GHSP-W2_v02_M_STATA/All One Folder"
19  global w3 "$rawghsp/Wave 3 (2015-16)/Data/NGA_2015_GHSP-W3_v02_M_Stata"
20  global w4 "$rawghsp/Wave 4 (2018-19)/Data/NGA_2018_GHSP-W4_v03_M_Stata12"
21  global epar "$home/Data/EPAR Analysis/335 Indicator Curation/Nigeria GHS/Data"
22  global harmo "$home/Data/LSMS-ISA/Harmonized/Data"
23  global clean "$home/Nigeria Floods GHSP/Clean Data/GHSP"
24  global code "$home/Nigeria Floods GHSP/Code"
25  global tab "$home/Nigeria Floods GHSP/Output/Tables/New"
26  global fig "$home/Nigeria Floods GHSP/Output/Figures/New"
27
28  include "$code/Balance_reg_fe.do"
29  include "$code/Balance_reg.do"
30
31  set scheme s1color // s1color, s2color, tab1
```

Example: reproducible package environment



The screenshot shows a file explorer interface with the following path: C > Dropbox > Class-DataScience > Setup. The 'Setup' folder contains four files:

Name	Date modified	Type	Size
Data_Science_Class_Setup_Guide.docx	1/6/2026 10:08 AM	Microsoft Word D...	1,024 KB
Data_Science_Class_Setup_Guide.pdf	1/6/2026 10:08 AM	Adobe Acrobat D...	691 KB
env_ml.yml	12/23/2025 9:34 PM	YML File	1 KB
environment.yml	12/23/2025 9:22 PM	YML File	1 KB

Writing Python code

- Into Jupyter!