Four things that change the answer when nothing changed

The clock, the seed, the file order and the locale. Each produces a different result from identical code and identical data, each is invisible in a diff, and this platform hit three of the four in production.

Pierre Robentz Cassion

Lesson 4 of 8

What this lesson covers

- One: the clock
- Two: the seed
- Three: the order files come back in
- Four: the locale
- The test that catches all four
- The deterministic-output habit
- Report it whole
- What comes next

One: the clock — In Python

```
from datetime import date
report_date = date.today()          # different every day, by design
```

One: the clock

- Anything that reads the current time makes every run differ — A generated-on footer, a filename with today's date, a...
- The fix is to take the date from the data or from a parameter

One: the clock — In Python

```
import pandas as pd

survey = pd.read_csv("data/raw/household-survey-2025.v1.csv")
as_of = pd.to_datetime(survey["interview_date"]).max()   # from the data
```

One: the clock — In R

```r
as_of <- max(as.Date(survey$interview_date))
```

One: the clock

- This platform got this wrong and the failure is instructive — The course handouts and lesson decks originally took…
- The date is now scoped to the artefact — A handout's date comes from its own course and lessons, a deck's from its own…

Two: the seed — In Python

```
import numpy as np

rng = np.random.default_rng(20260729)      # stated, committed, reproducible
sample = rng.choice(households, size=200, replace=False)
```

Two: the seed — In R

```
set.seed(20260729)
sample(households, 200)
```

Two: the seed

- Anything random needs a seed and the seed belongs in the code — Bootstrap intervals, random sampling for verification,...
- Set it once, at the top, visibly — A seed buried three functions deep is a seed someone will move
- And say so in the output — The regression course's bootstrap interval is reported as "400 resamples, seed 20260729" for...
- Every dataset on this platform is generated by a seeded script — which is what makes `pnpm datasets:generate` a...

Three: the order files come back in — In Python

```
import glob
for path in glob.glob("data/raw/*.csv"):      # order is filesystem-dependent
    ...
```

Three: the order files come back in

- `glob` and `os.listdir` return files in an order that differs between operating systems and sometimes between runs — If...

Three: the order files come back in — In Python

```
for path in sorted(glob.glob("data/raw/*.csv")):
    ...
```

Three: the order files come back in — In R

```r
for (path in sort(list.files("data/raw", full.names = TRUE))) { }
```

Three: the order files come back in

- Sort it. Always. It costs six characters
- The same applies to dictionary and group order — `groupby` in pandas sorts by default and `dplyr::group_by` does not; a...

Four: the locale — In Python

```
float("1,234")          # ValueError, or 1.234, depending on where you are
```

Four: the locale

- The decimal separator, the thousands separator, the date format and the sort order of accented characters are all...

Four: the locale — In Python

```
survey = pd.read_csv(path, decimal=".", thousands=None)      # stated, not inferred
dates = pd.to_datetime(survey["date"], format="%Y-%m-%d")    # explicit
```

Four: the locale — In R

```
readr::read_csv(path, locale = locale(decimal_mark = ".", date_format = "%Y-%m-%d"))
```

Four: the locale

- State the format rather than letting the reader infer it — `pd.to_datetime` without a format is a function that…
- Sorting is the subtler one — `sorted()` on French commune names orders `Étroit` after `Zone` under one locale and…

The test that catches all four — Shell

```
python run.py && cp -r outputs outputs-first
python run.py && diff -r outputs outputs-first
```

The test that catches all four

- Run it twice and diff — Anything that differs is one of the four, and the diff names the file
- Run it twice on different machines — and you additionally catch the locale and the file ordering, which a single...
- Put it in CI — which is a clean machine every time, and the check runs whether or not anyone remembers

The deterministic-output habit

Source of churn	Fix
Timestamps in file metadata	`SOURCE_DATE_EPOCH`, or strip them
A generation date in a footer	Take it from the content
Compression with a timestamp	A zip whose entry times are pinned
Floating-point summation order	Sort before reducing where it matters
An embedded random ID	Derive it from a hash of the content

The deterministic-output habit

- This platform pins `SOURCE_DATE_EPOCH` for both pdfTeX and pandoc — because a `.pptx` is a zip whose entry times and…
- The payoff is that a diff means something — When rebuilding produces no change, a change in the diff is a real change —…

Report it whole — Example

Determinism

All randomness is seeded: seed 20260729, set in src/config.py and reported with every bootstrap interval in this report.

Dates are taken from the data (the latest interview date) rather than from the clock. No output contains a generation timestamp.

File iteration is sorted. Group order is stated explicitly rather than inherited from the grouping library's default.

CSV reading states the decimal mark and the date format rather than inferring them.

Verified: running the pipeline twice produces byte-identical outputs, and CI runs it on a clean machine on every push.

Report it whole

- The last line is the claim and the four above it are how it was achieved — A report that asserts determinism without. . .

What comes next

- A reproducible pipeline that produces one report is worth having.

Read the full lesson, with runnable code

https://data-analysis.cassion.dev/courses/reproducible-workflows/repro-04-what-makes-a-rerun-differ