traffic tutorial

a toolbox for processing and analysing air traffic data

Xavier Olive, ONERA

17th November 2021

Agenda

1. General introduction (1h)
    - Introduce yourself
    - History: why a traffic library?
    - Overall structure of the library
    - Demonstration
2. Hands-on exercices (1h+)
3. Bring your own problem (?)
4. Perspectives

History

Started early 2018, based on two frustrations:

- how to download data from OpenSky Impala shell
- a lot of boilerplate one shot code to copy/paste between two papers

and one question:

- how did you make those maps? (hold my beer...)

Frustration with existing tools:

- `pandas` misses semantics for trajectories
- `geopandas` suits well statical geometrical shapes, but not time series

Structure of the library

- Aeronautical data
- Core structures
- Reference datasets
- Visualisation

Demonstration

- Code: https://github.com/xoolive/traffic/
  Documentation: https://traffic-viz.github.io/

  also https://atmdata.github.io/ is a good reference

- Let's go through (live coding !), in particular:

  - **aeronautical data**: airports, airways, airspaces, etc.
  - **access to external data**: DDR, ADS-B
  - exploring `Flight`, `Traffic`, `LazyTraffic`, `FlightIterator`

Hands-on exercices

Problem 1

Find all trajectories doing more than one go-around in the `landing_zurich_2019` dataset.

Build a `Traffic` structure containing those trajectories.

Plot them with a different color according to the landing runway.

Problem 2

Download data for aircraft landing at Toulouse airport and within the bounds of the TMA (`LFBOTMA`) between the 1st and the 10th of January 2020.

Toulouse airport has two parallel runways:
count how many aircraft performed a runway change without go-around.

Problem 3

Compute an occupancy plot (number of aircraft inside an airspace at any time) over airspace LFEE5R on 15 October 2021, between 7am and 11pm **local time**.

Problem 4 (at home)

Pick up a city pair, highlight significant segments not aligned with any navigational beacon on the usual flight plan. Try to explain the most surprising situations.

Problem 5 (at home)

Pick one day of traffic landing at Amsterdam airport: identify top of descent. Plot areas of constant Mach, constant CAS on a Northern Europe map.

Bring your own problem?

Perspectives

- Provide more reference data sets, enriched with metadata
- “Standardise” definitions of aircraft trajectory processing
  Encourage implementations in more programming languages: R, Javascript, etc.
- Better tests, better documentation! (Enrico?)
- Go more scalable (Spark?)
- Interactive visualisation tools for in depth analysis (Three.js?)

Key take-aways

- Open-source is better than closed source
  … but it does not mean it is perfect ××× DISCLAIMER ×××
- Data has no immediate value
- Information is valuable, but the extraction process is hard
- Code is not precious, expertise is
- Humans read code, help experts read it too
- Use declarative style for better reproducibility