traffic

a toolbox for processing and analysing air traffic data

Xavier Olive

An open-source library

xoolive / **traffic**

A toolbox for processing and analysing air traffic data

traffic-viz.github.io/

MIT License

- Code: https://github.com/xoolive/traffic/
  Documentation: https://traffic-viz.github.io/
- Development started early 2018
- traffic, a toolbox for processing and analysing air traffic data,
  Journal of Open Source Software (4), 2019. DOI: 10.21105/joss.01518

The researcher's wish list

- **Access to (open) data**
  Includes trajectories, flight plans, airspace structure, weather information, etc.
- **Trajectory preprocessing**
  Clean and filter data, prepare datasets, enrich with metadata
- **Algorithms**
  clustering, detection of operational events, implement KPI
- **Data visualization**
  Maps, interactive visualization

Access to (open) data

- Access to OpenSky Network historical database
  https://github.com/open-aviation/pyopensky
- Access to description of airspace structure
  Parsing facilities for XPlane format, Eurocontrol DDR + AIXM data, FAA open data
- Access to OpenStreetMap data
  https://github.com/xoolive/cartes
- Access to weather data
  METAR history information, ERA5 with https://github.com/junzis/fastmeteo

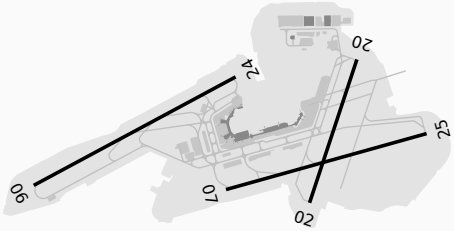
A tabular *tidy* format: pandas, geopandas, xarray

Access to (open) data [code]

```
from traffic.data.samples import *  # for documentation and testing
from traffic.data.datasets import *  # public datasets included in publicatio

from traffic.data import airspaces  # public sources or AIRAC data
from traffic.data import airports  # airports, runways, apron structure
```

```
airports["LFPO"]
```



Trajectory preprocessing

`pandas` misses a semantics for trajectories

`geopandas` suits well geometrical shapes, not time series

There are common noise patterns in data that we learn to process.

`traffic` comes up with a semantics for processing trajectory data

The same semantics applies on:

- individual trajectories (`Flight`) and
- collections of trajectories (`Traffic`)

Trajectory preprocessing [code]

```python
from traffic.core import Flight, Traffic

Flight.from_file(...)  # one single trajectory
Traffic.from_file(...)  # a collection of Flight

flight.duration  # a pd.Timestamp
flight.first("10 minutes")  # a new Flight
flight.intersects(LFBOTMA)  # a boolean
flight.simplify()  # Douglas-Peucker simplification
flight.aligned_on_ils("LFPO")  # iterates on segments on final approach
flight.go_around()
flight.holding_pattern()
```

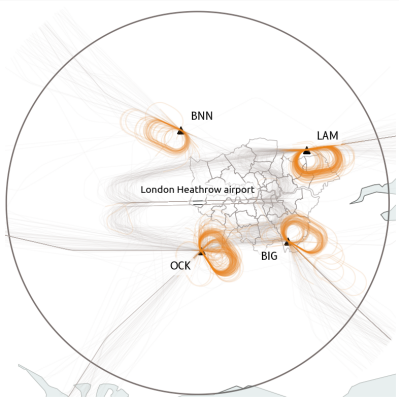
Algorithms

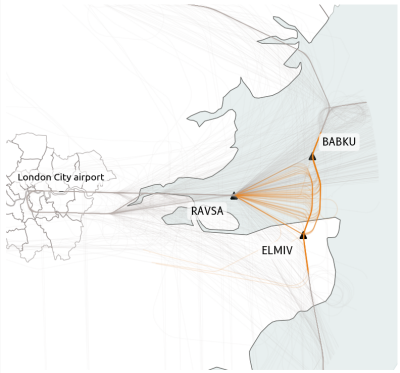
Efficient implementations for common problems:

- **detection of specific events**
  holding patterns, go around, point merge, in-flight refuelling, aerial survey, firefighting, etc.
- **trajectory clustering**
- **trajectory generation**
- **closest point of approach**
- **fuel flow estimation**, with OpenAP
- etc.

Example: select holding patterns and point merge systems

```
collection.has("aligned_on_ils('EGLL')").has("holding_pattern").eval()
```





Example: compute an occupancy graph for a given airspace

```
collection
.within_bbox(airspaces["LFEE5R"])
.intersects(airspaces["LFEE5R"])
.clip(airspaces["LFEE5R"])
.summary(["callsign", "icao24", "typecode", "start", "stop", "duration"])
.eval()
```

Data visualization

Plotting facilities for all data structures with common visualization solutions:

- Matplotlib
- Leaflet (ipyleaflet) explore trajectories in a widget
- Plotly interactive visualizations

and more exploratory options:

- Mapbox (Open GL)
- JavaScript/Observable options

Future developments

- Trajectory prediction
- Interaction with state-of-the-art tools
- Improvements on performance
- Facilitate the usage of the same semantics on real-time data
- Exploration of more visualization techniques

The 12th OpenSky Symposium

7/8 November 2024, Hamburg, Germany

https://symposium.opensky-network.org/