UNIVERSITÄT LEIPZIG

Peter Bräuer (@pb866)

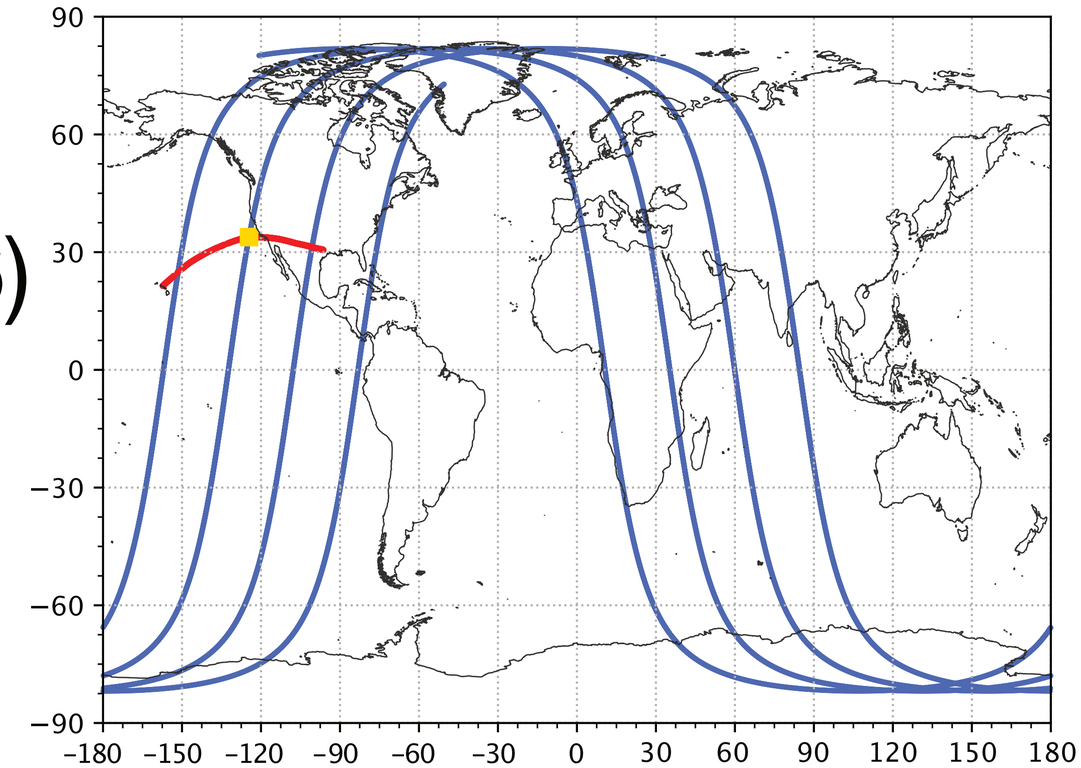
Leipzig University, Institute of Meteorology,
Stephanstr. 3, 04103 Leipzig, Germany

peter.braeuer@uni-leipzig.de

TrackMatcher

A tool for finding intersections in trajectories

https://github.com/pb866/TrackMatcher.jl.git



MAKE OUR PLANET GREAT AGAIN

SPONSORED BY THE

Federal Ministry of Education and Research

Motivation

- **Polar orbiting satellites** are an effective means to study global climate effects in detail
- To **study effects of single vessels/objects on climate**, **find intersections between** the object's and satellite's **trajectories**
  - Find time/lat/lon/altitude of intersections
  - Identify time delay between satellite overpass and object at intersection
  - Save further interesting data in the vicinity of intersections and monitor meteorological conditions
- **Highly adaptable package for various data types**
  - Current focus on aircraft data from various sources and CALIOP data onboard the CALIPSO satellite
  - Planned to extent for cloud tracking
  - Adaptable to other possible uses such as ship tracks
- **Application-driven development**
  - Current aim: Investigate contrail effects on pre-existing cirrus clouds

Aim of the presentation

- **Trigger discussion** about **best algorithm**
- Discuss **efficient ways to store and process big data**
- Discuss **helpful available packages**
  - Discuss currently missing methods to **read HDF4** satellite files and for **PCHIP interpolation**
  - Currently done with MATLAB
- **Show use case** in climate research
- **Focus on computational methods** (not scientific results)

The TrackMatcher tool

- Current focus on **flight data from different sources and CALIPSO satellite data**
- **Calculate intersections**
  - Switches to **control data storage**
  - Switches to **compromise between computation time and accuracy** of results
  - **Save meteorological conditions** at flight level at the intersection
- Flight/satellite/intersection data are stored in structs
  - Instantiation of structs with switches to control data storage and accuracy of intersection calculations
- Tested against latest stable Julia version (1.4.2) and long-term support version (1.0.5)

Main challenges

- **Quantity of data**
  - **Large number of flights** needed for statistically relevant analyses (associated with **high procurement costs**)
  - **Long computation times; huge data files**
- **Accuracy of track data**
  - May lead to inaccurate results or duplicate intersection finds (see next pages)
  - **Large gaps in the flight-track data**
- **Choice of algorithm/Julia packages**

The track-matching algorithm

- **Filter** relevant data (e.g., altitude levels)
- **Split track into segments** with strictly monotonic x data (see also Fig. 1a)
  - Use lat as x data for satellite data
  - Chose lat or lon as x data for flights depending on prevailing flight direction
- **Interpolate tracks** with MATLAB's PCHIP method with a preset accuracy (default: 1 km)
- **Find minimal distance** between interpolated **lat/lon pairs** of flight and sat track within a threshold
- **Find analytic solution** by linear interpolation between 4 closest points to intersection (see Fig. 1b+c)

Current limitations

- Track data needs **splitting into segments** with strictly monotonic x data
  - Lat for satellite data; lat or lon for flight data
  - Only segments with at least 3 data points are considered
  - Many small or disregarded segments for inaccurate track data (see Fig. 2 next page)
  - Can lead to duplicate intersection finds at segment borders (Fig. 2)
- **Dependence on MATLAB licence**
  - For reading CALIOP HDF4 files
  - For track interpolation with PCHIP method
  - Can be rewritten using open source software, e.g., a Python wrapper or directly in Julia

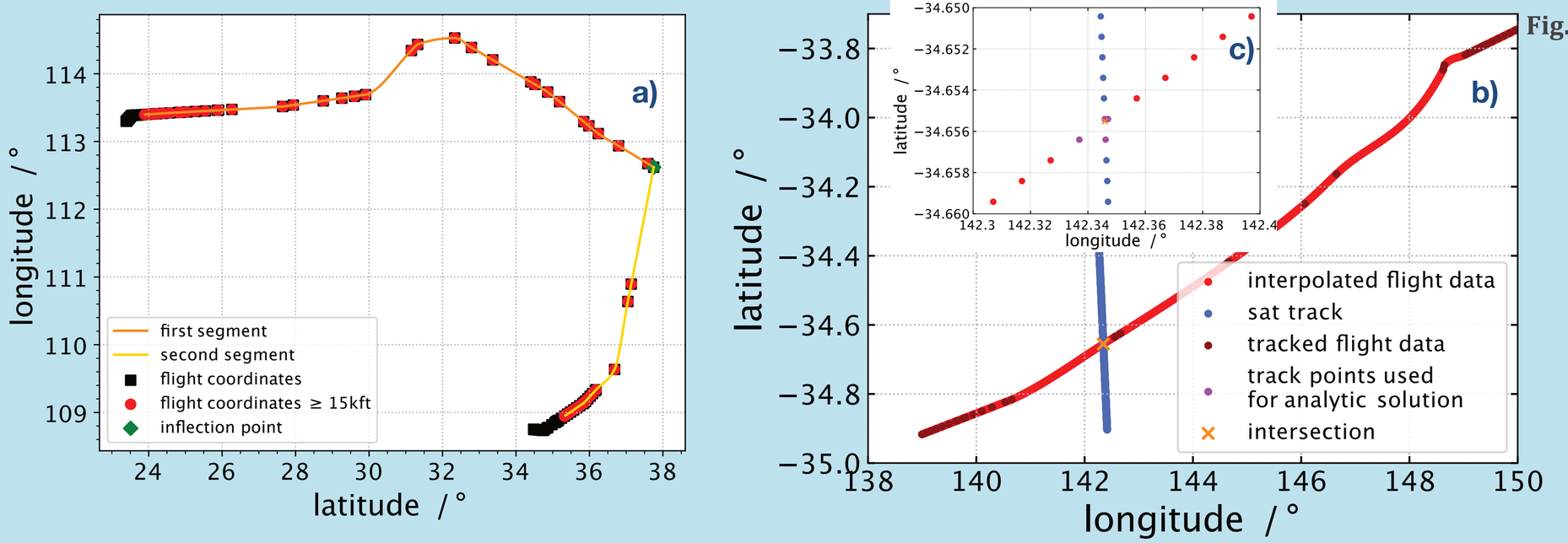


Fig. 1 Illustration of the TrackMatcher algorithm principles. Track data are filtered and relevant data split into segments with strictly monotonic x data (a). Segments are interpolated and the four closest points to a possible intersection are used to determine an analytic solution by linear interpolation (b+c).

Application in test cases

- VOLPE AEDT test set
  - Global routes for 1./2. Jan 2012
- **Very accurate results** within a few meters (see Tab. 1)
  - **Only few outliers** demonstrated by the mean in the range of the 95 percentile
  - Only 172 (2.8%) of intersection accuracy values are significant different from 0 (> std. dev.)
- **Large dataset needed** to study contrail effects on cirrus clouds (one aim of this study)
  - Only 440 cases (0.3% of total flights; see Tab. 2)
- **Large gaps in the flight-track data** over open ocean possible
  - Demonstrated by a maximum distance of an intersection to the nearest track point of 1420 km and a mean distance of 22.3 km (see Tab. 1)

Tab. 1 Accuracy of the intersection calculation (first column) and distance to the nearest available flight-track point (second column). All data is given in meters.

	Intersection	Flight track
Minimum	0.0	1.17
5 percentile	0.015	669.9
Median	0.188	$1.00 \cdot 10^4$
Mean	11.56	$2.23 \cdot 10^4$
95 percentile	8.94	$7.17 \cdot 10^4$
Maximum	996.6	$1.42 \cdot 10^6$
Standard dev.	75.93	$5.34 \cdot 10^4$

Tab. 2 Total number of flights, number and percentage of intersections between flight and satellite tracks found, and meteorological conditions at the intersections.

	counts	percentage
Total flights	149867	100.00
Total intersections	6036	4.03
Single intersection/flight	5338	3.56
Multiple intersection/flight	698	0.47
No lidar signal	125	2.07*
Clear sky	5270	87.31*
Cirrus	**440**	**7.29***
Deep convective clouds	179	2.97*
Dust	13	0.22*
Clean continental aerosol	1	0.02*
Stratospheric aerosol/cloud	8	0.13*

*percent of total intersections

Conclusions & outlook

- Find **intersections between trajectories**
  - Meant for intersections between satellite and flight tracks
  - Highly adaptable to other data, e.g., ship or cloud tracks
- **Open source** Julia package on GitHub
- Needs **further refinements of the algorithm**
- **Performance improvements**
- Further switches needed
  - Exclude intersections with only distant measured track points
  - Tolerance threshold for finding closest interpolated points to a possible intersection (currently set to interpolation step width)
- Test use of **Mercator projection together with linear interpolation** to compensate for inaccurate data (as in Fig. 2)

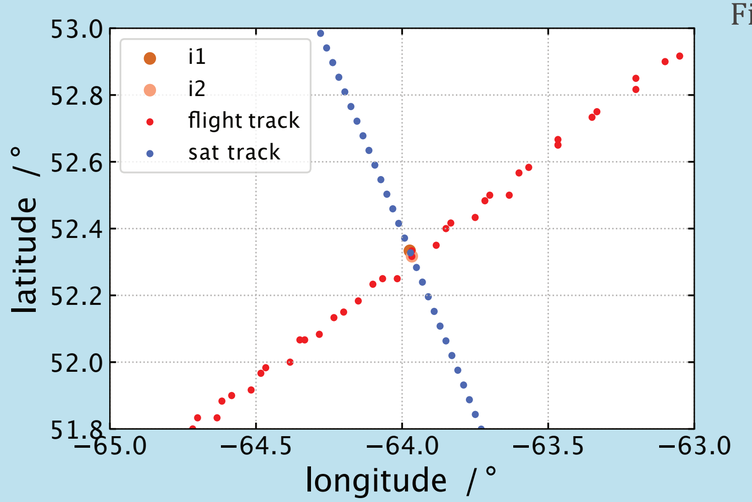


Fig. 2 Inaccurate flight-track data causing fragmentation of the flight track in many small segments with segment boundaries in the vicinity of the intersection. Points of each segment closest to the intersections are taken to derive the intersection. If the spread in the data is to wide, filters cannot find duplicate counts of intersections.