KN - Journal of Cartography and Geographic Information (2020) 70:3–10
https://doi.org/10.1007/s42489-020-00039-y

Open Geospatial Tools for Movement Data Exploration

Anita Graser[1,2] · Melitta Dragaschnig[1]

Received: 5 December 2019 / Accepted: 18 January 2020 / Published online: 6 February 2020

Abstract
Movement data exploration presents a significant challenge due to the heterogeneity of movement datasets and analysis tasks. Furthermore, there is a lack of established tools for the exploratory analysis of movement data, as well as a lack of literature on best practices for applying corresponding concepts using commonly available data analysis tools. To address this gap, we present three open-source technology stacks for the exploratory analysis of movement data and discuss their capabilities and limitations.

Keywords Exploratory data analysis · Mobility · Mobile data · Trajectories · Open source

Freie Werkzeuge für die explorative Analyse von Bewegungsdaten

Zusammenfassung
Explorative Analysen von Bewegungsdaten stellen aufgrund der Heterogenität von Bewegungsdatensätzen und Analyseaufgaben eine erhebliche Herausforderung dar. Es mangelt an etablierten Tools für die explorative Analyse von Bewegungsdaten sowie an Leitfäden für die Umsetzung existierender Bewegungsdatenanalysekonzepte mithilfe allgemein verfügbarer Analysewerkzeuge. Um diese Lücke zu schließen, stellen wir drei Open-Source-Technologiepakete für die explorative Analyse von Bewegungsdaten vor und diskutieren deren Fähigkeiten und Grenzen.

1 Introduction

Movement of people and goods relates to many pressing issues, including climate change and increasing road traffic deaths (WHO 2018). Therefore, analysts and scientists from various domains, such as ecology, health, transport, and safety, collect and analyze movement data. They then face the challenge of extracting relevant information from the collected data. However, the wide range of domains, applications and analysis methods, as well as the rapidly expanding and often complex movement datasets present a major analysis challenge (Long et al. 2018).

Interactive and exploratory visual tools can help make sense of complex datasets. Exploratory data analysis (EDA), as established by Tukey (1977), aims to analyze datasets by summarizing their main characteristics to determine what information the data contains. As illustrated by Fig. 1, EDA thus helps to:

- Suggest hypotheses about phenomena observed in the data and their causes.
- Assess assumptions about the data collection and processing steps.
- Select appropriate tools and techniques for further analysis.
- Provide a basis for further data collection.

In the specific context of movement data, Andrienko et al. (2013) provide an extensive overview of relevant EDA concepts and application examples that goes beyond what can be covered in a single paper. However, there is a lack of established EDA tools for movement data as well as a lack of literature on best practices for applying EDA concepts using commonly available data analysis tools. This limits many concepts to theoretical discussions or prototypical

✉ Anita Graser
anita.graser@ait.ac.at

1 AIT Austrian Institute of Technology, Vienna, Austria

2 University of Salzburg, Salzburg, Austria

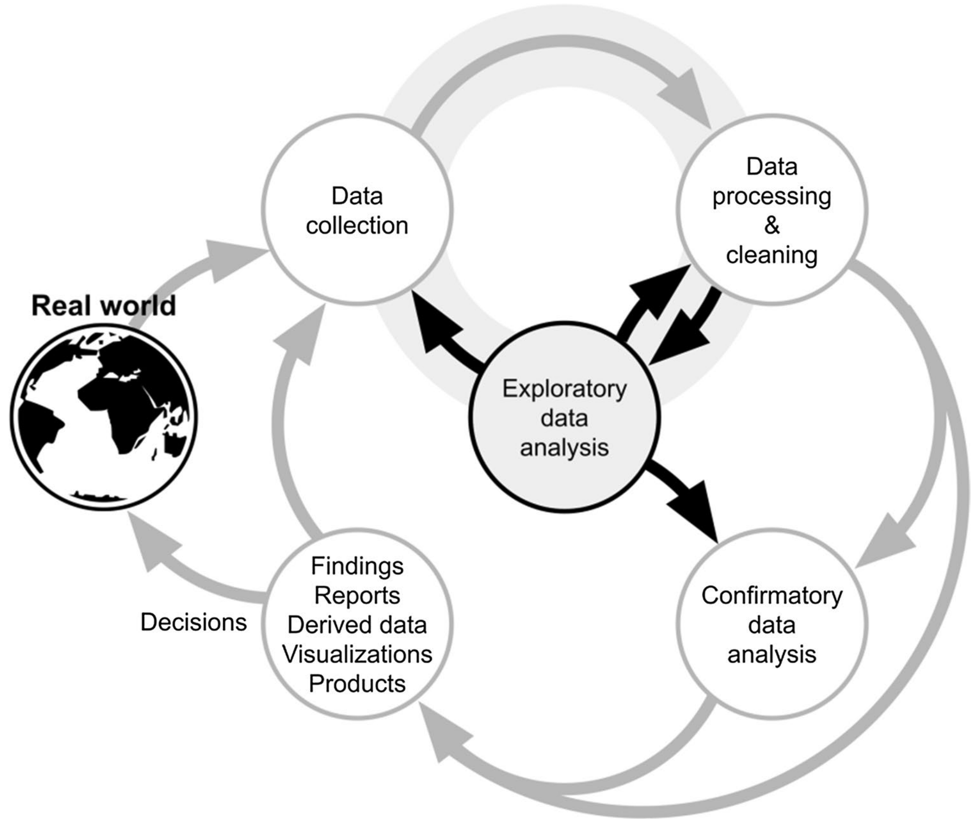

Fig. 1 EDA within the broader data science framework (Graser 2020a)

implementations that are not openly available to other researchers and data analysts. To address this gap, this article presents three open source technology stacks for the exploratory analysis of movement data and discusses their capabilities and limitations.

The remainder of this paper is structured as follows: Sect. 2 introduces a movement data exploration workflow and establishes a framework of data characteristics that influence analyses. Section 3 discusses software tool stacks for movement data exploration that rely on commonly available open source tools. Finally, Sect. 4 summarizes the findings and points out open issues for future research and development.

2 Movement Data Exploration

A general workflow for the exploration of movement data can be summarized into the following four steps.

1. *Establishing an overview* by visualizing raw input data records (including assessment of spatiotemporal extent and gaps in the data)
2. *Putting records into context* by exploring information from consecutive movement data records (such as time between records, speed, and direction)
3. *Extracting trajectories, locations and events* by dividing the raw continuous tracks into individual trajectories, locations, and events
4. *Exploring patterns and outliers* in trajectory and event data by looking at groups of trajectories or events (including similar trajectories, popular locations or events) and how they may challenge preconceived assumptions about the dataset characteristics.

The development of general-purpose tools for the exploration of movement data; however, is complicated by the fact that movement datasets are very heterogeneous. Datasets vary with respect to spatial and temporal extent and resolution, spatial dimensions, movement models, tracking system, data size, as well as movement and privacy constraints (Graser 2019). Consequently, EDA tools should:

- adapt to varying *spatial and temporal extents*, for example, by providing suitable base maps and other contextual information
- take variations in *spatial resolution or positioning accuracy* into account, for example, to avoid misinterpretation due to feigned higher accuracy
- communicate *temporal resolution* correctly, for example, to avoid misleading visualizations, such as density maps of irregularly sampled or mixed resolution data

- provide functionality for *open space* movement as well as *network-constrained* movement data, which needs to be matched to the underlying network
- adapt to the underlying *movement model* (Dodge et al. 2016): Lagrangian (continuous tracking, for example from GPS trackers) or Eulerian (checkpoint-based, for example from Bluetooth beacons or camera traps)
- deal with specifics of the *tracking system*, such as observation gaps, detection errors, or deliberate false information
- support different *dataset sizes*, including increasingly common massive tracking data
- ensure *privacy* when dealing with personal data.

Available data analysis tools implement different aspects to varying degrees. Geographic information systems (GIS) are commonly used due to their strong spatial data handling functionality. However, lacking support for the temporal dimension in current GIS limits their potential for movement data analysis (Graser 2018).

In research areas with lower GIS adoption rates, data exploration tools scripted in R or Python are popular. For example, Joo et al. (2019) list 58 R packages dealing with movement data and Pappalardo et al. (2019) and Graser (2019) present Python libraries for movement data.

To the best of our knowledge; however, there are no openly available visual analytics tools for movement data that implement privacy by design.

Considering the heterogeneity of applications and datasets, it does not seem realistic to expect a single general-purpose movement data exploration tool that could cover all requirements. Instead, specialized exploratory tools and workflows can be built to cover specific reoccurring use cases. The following section presents three different open source technology stacks that we use to perform exploratory movement data analysis tasks.

3 Open Tools for Movement Data Exploration

It is impossible to cover all potential open tools for movement data exploration within the limits of a single paper. Therefore, the following examples present a selection of established tools and novel movement data-specific tools that built on established open source software to explore continuous tracking data.

The first example discusses a solution that combines the open source desktop GIS QGIS (2019) with the relational database system PostgreSQL, with PostGIS extension (PostGIS 2019). The second example presents movement analysis libraries built on the established Python data analysis library Pandas (McKinney 2010) and how they can enable more reproducible workflows. Finally, the third example discusses distributed trajectory processing in Apache Hadoop ecosystems (Apache 2019a) which enable processing of massive movement datasets that cannot be handled with conventional tools.

3.1 Desktop GIS and Spatial Databases (QGIS and PostGIS)

Desktop GIS are among the most common tools for exploring movement data used by people with backgrounds in geography, GIScience, spatial planning, and related disciplines. QGIS, for example, offers multiple tools specialized on spatiotemporal data in general and movement data in particular, including Time Manager for animating spatiotemporal data (Graser 2011), and edge bundling tools (Graser et al.

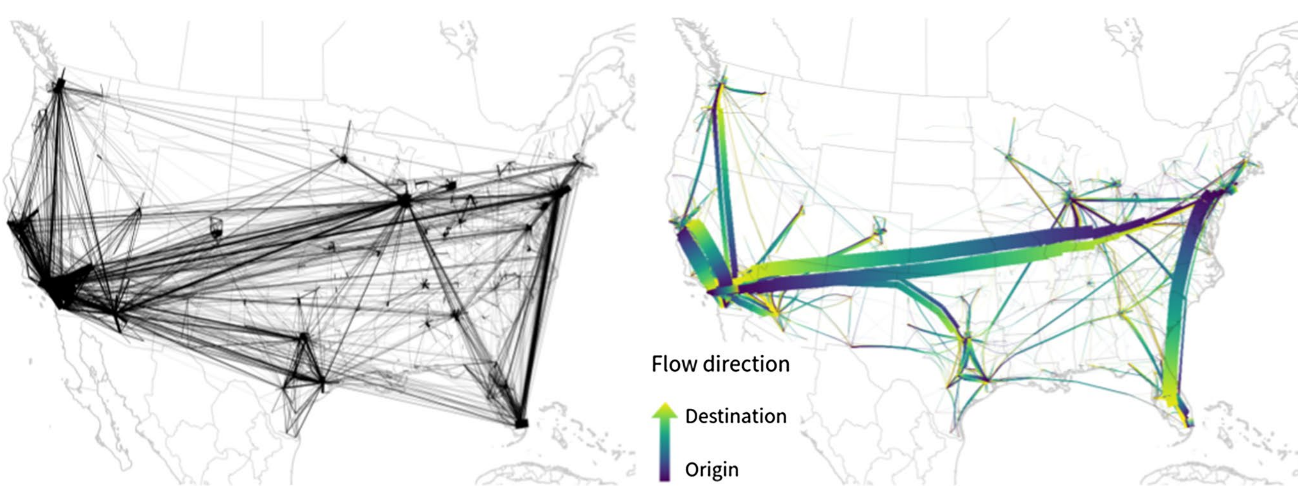

Fig. 2 Edge bundling takes raw OD flows (left) and bundles them along common paths to improve readability (right)

2017) for clearer origin–destination (OD) flow visualizations, as shown in Fig. 2. Furthermore, PostGIS provides some built-in support for trajectory data handling which is covered in detail by Graser (2018).

QGIS provides a wide range of rendering options and good rendering performance. It is easy to add spatial context using different base maps and auxiliary datasets. There are many analysis tools that can be applied without the need for advanced programming skills, including, for example, map-matching tools that can match a trajectory to an underlying network (Jung 2019). Both QGIS and PostGIS are easy to set up and there are big communities that provide commercial as well as community support.

The downside of QGIS and PostGIS (as well as, to our knowledge, all other desktop GIS and spatial databases) is that they provide only limited time dimension support. The key issue is that the OGC Simple Features standard implemented by most GIS does not cover the temporal dimension. Instead, time information is stored in attribute fields that are without any particular significance to the GIS system. Consequently, since there is little built-in time support, there is also almost no movement data support.

The trajectory support implemented in PostGIS bypasses some of the restrictions of Simple Features by storing time information in the measure value of LineStringM features (Graser 2018). This approach makes it possible to create a single LineStringM feature that represents a whole trajectory where every point along the trajectory retains its timestamp stored in the measure value. This enables functions that compute, for example, the closest point of approach between two trajectories. QGIS can access the spatial and temporal information stored in the LineStringM trajectory. For example, Fig. 3 shows how to compute and visualize speed along a trajectory on the fly (without having to split the trajectory into individual segments between consecutive points).

3.2 Interactive Notebook Environments (Jupyter and MovingPandas)

Notebook environments, such as Jupyter (Kelley et al. 2016) and Zeppelin (Apache 2019b), enable interactive documents that combine code, visualizations, and narrative text. This example focuses on Python libraries, since many spatial data analysts are familiar with this language as it is the scripting language of choice in many GIS environments. However, both Jupyter and Zeppelin support a wide range of programming languages, including Python, R, and Scala.

MovingPandas (Graser 2019) and sci-kit mobility (Pappalardo et al. 2019) are the two recently published Python libraries for handling movement data based on the Pandas data analysis library. Pandas provides extensive functionality for time series handling which lends itself to modeling movement data as time series of locations. MovingPandas and sci-kit mobility, both implement dedicated classes for trajectories that enable analysts to interact with movement data in the form of trajectory objects. For example, Fig. 4

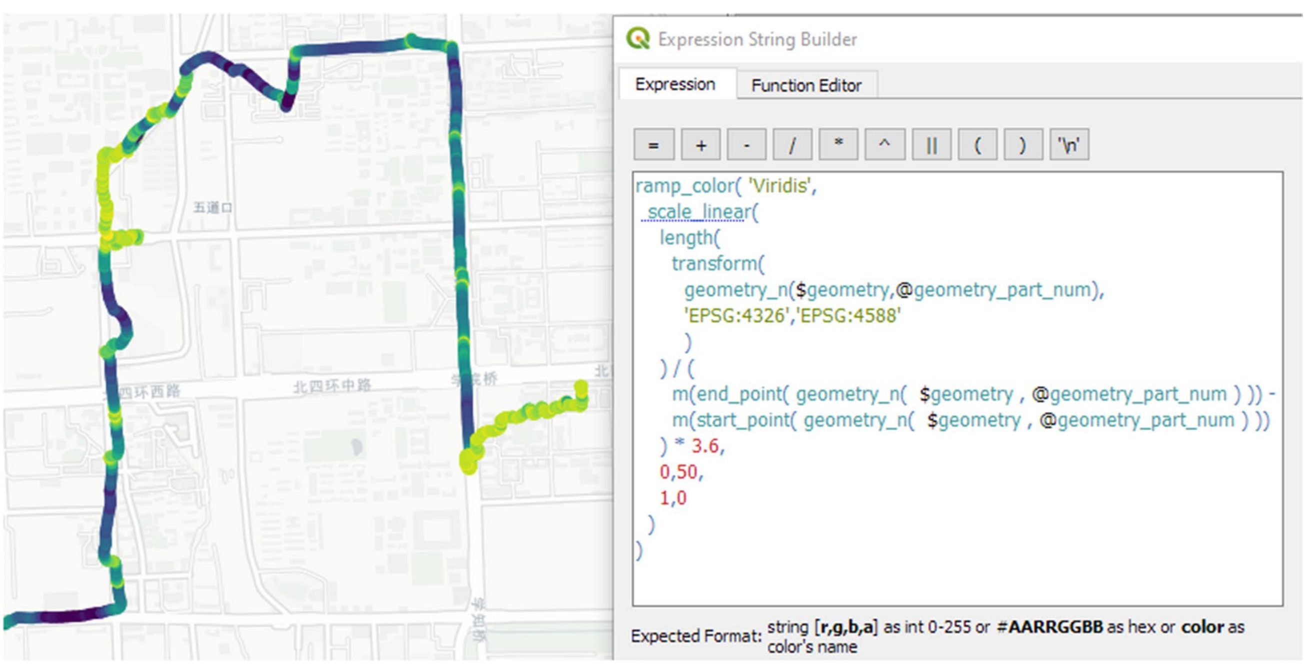

Fig. 3 QGIS screenshot showing the speed along a trajectory modeled as a single LineStringM feature. The data-driven expression computes speed and translates it to lighter colors on the Viridis color scale for lower speed values and darker colors for higher speed values. [Data courtesy of the Geolife project (Zheng et al. 2010)]

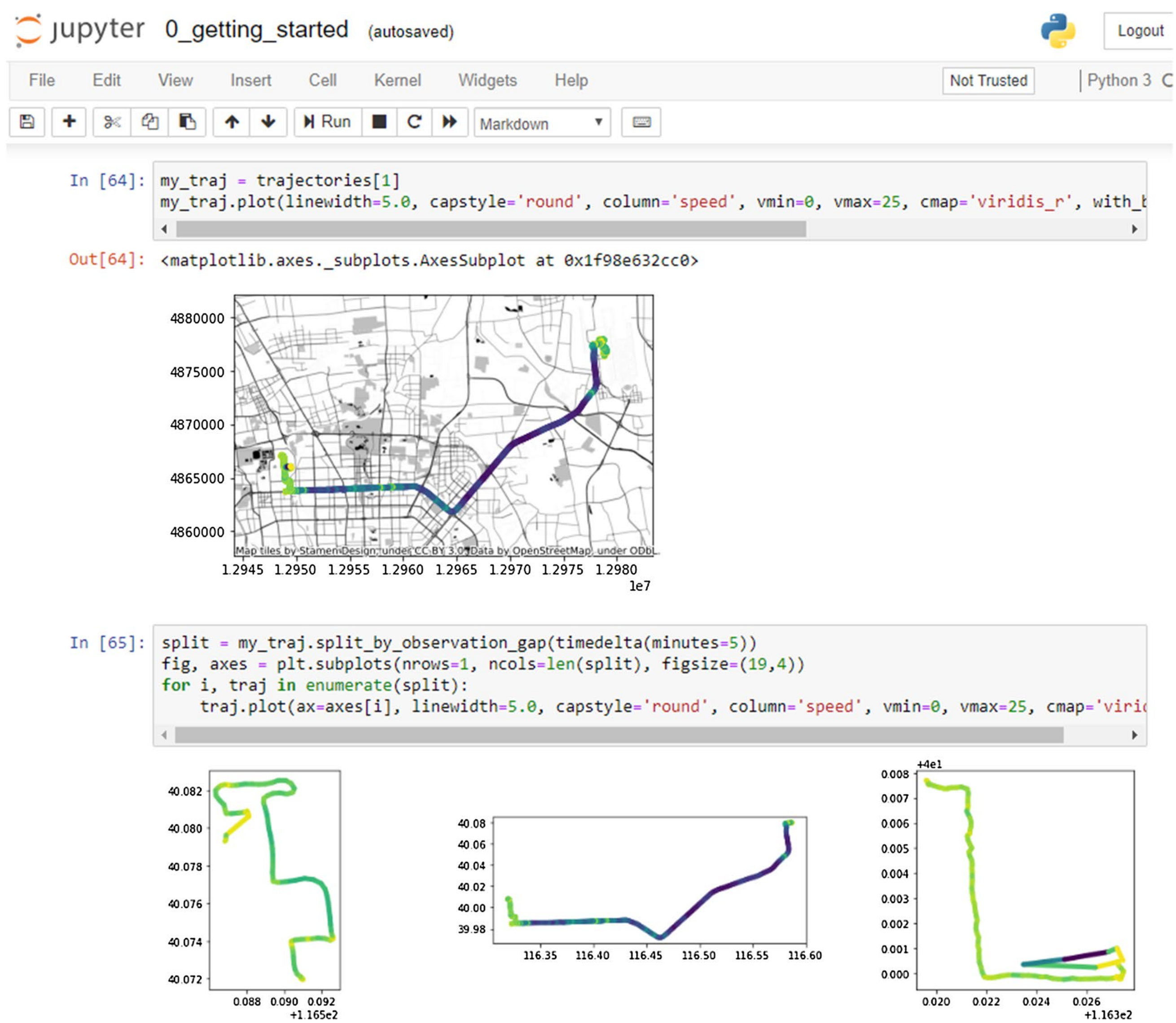

Fig. 4 Jupyter notebook screenshot showing the close integration of Python code and resulting data visualizations. The original trajectory on the top is split into subtrajectories whenever there is a time gap of more than 5 min between consecutive observations. Like in Fig. 3, line color represents movement speed

shows how MovingPandas can be used to split a trajectory into subtrajectories and plot the results. Optionally, base maps can be added to the plots to provide geographic context. The notebook environment ensures that resulting visualizations are presented within the context of the code that generated them. This improves the reproducibility of analysis results.

In contrast to desktop GIS; however, the use of interactive coding notebooks requires some familiarity with programming concepts. The spatial visualization capabilities are more limited than within desktop GIS. There are multiple options for map plots, including Matplotlib (Hunter 2007) for static plots, Folium (Folium 2019) for interactive maps using the popular Leaflet web mapping solution (Leaflet 2019), and hvpolt (Pyviz 2019) which supports a wide range of interactive plots and dashboards.

Due to the specialized nature of dedicated movement data analysis libraries, the corresponding user communities are rather small. Furthermore, performance for big datasets still leaves something to be desired.

3.3 Distributed Computing for Large Datasets (GeoMesa and Spark)

When datasets become too large for conventional systems to handle, distributed computing approaches can be used to process these large datasets more quickly. There are a variety of distributed storage and analysis solutions within

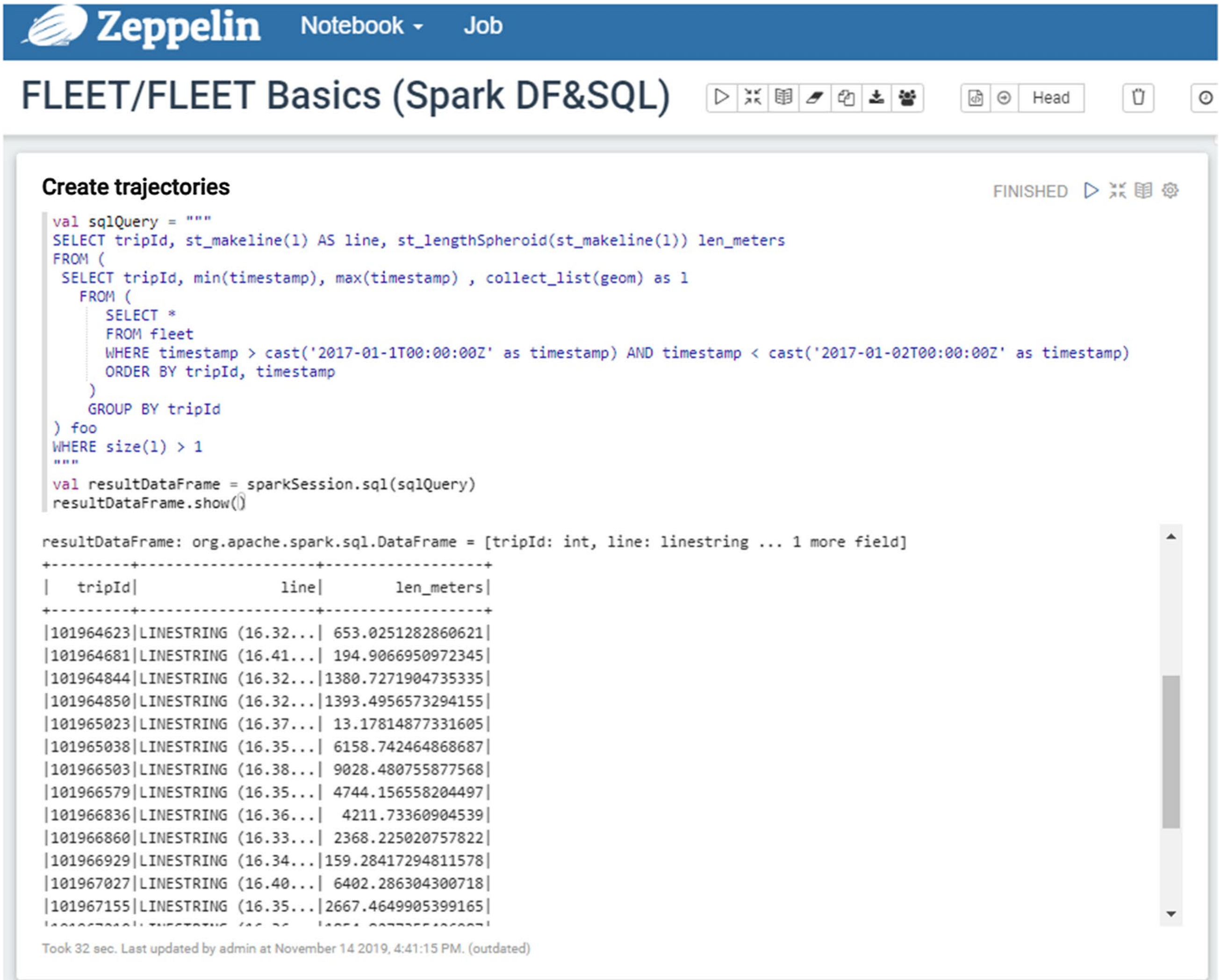

Fig. 5 Zeppelin notebook screenshot showing SparkSQL code for generating trajectory lines from points

the Apache Hadoop ecosystem and beyond. For example, GeoMesa (Hughes et al. 2015) provides a fast spatiotemporal indexing solution to help store and access spatiotemporal data. GeoMesa's spatial support is built on the well-established GeoTools (2019) library and data stored in GeoMesa can be published via GeoServer (2019) using standardized OGC web services, such as WMS and WFS. By supporting these standards, the combination of GeoMesa, GeoServer, and QGIS, for example, makes it possible to use QGIS Time Manager to create dynamic animation of data stored in a GeoMesa datastore.

Spark (Zaharia et al. 2010) is the most common solution to perform analysis on data stored in GeoMesa. Spark is a general-purpose cluster-computing framework. GeoMesa provides spatial analysis functions that can be called by Spark. For example, Fig. 5 shows how SparkSQL with GeoMesa functions can be used to create trajectory lines from individual points and to compute the trajectory length using spheroidal distance. GeoMesa also provides tools to find spatially similar sequences of points and to find points that are spatiotemporally close to a point sequence. However, there is no support for LineStringM features in GeoMesa and, therefore, it is not possible to apply the previously discussed PostGIS trajectory approach and store the time information at every position along the line.

Compared to the previous two technology stacks, this stack presents a steep learning curve due to the large number of components that are under rapid ongoing development and are not (yet) commonly used by movement data analysts. Furthermore, the user communities of spatial big data solutions are rather small which can make it hard to find up-to-date answers to questions that arise while using these tools.

Table 1 Comparison of features of the three presented technology stacks with respect to the four EDA steps

EDA step	QGIS 3.10 and PostGIS 2.5	Jupyter and MovingPandas 0.2	GeoMesa 2.4 and Spark
1. Establishing an overview	Good: interactive maps and support for various base maps and other data sources	Limited: static and interactive maps with various base maps (limited performance for large datasets in interactive maps)	Good: rendering of large datasets using GeoServer WMS integration or more flexibly using WFS that can be rendered in desktop GIS
2. Putting records into context	Good: intervals, speed, and direction derived from LineStringM values	Good: intervals, speed, and direction between consecutive records	Limited: only connections between consecutive points without time information (custom code required for interval, speed, and direction computations)
3. Extracting trajectories, locations & events	Limited: trajectory extraction using Trajectools plugin (Graser 2020b)	Good: built-in functions for splitting continuous location observations into trips	No dedicated functionality
4. Exploring patterns and outliers	Good: distance metrics between LineStrings and closest point of approach between LineStringMs; point clustering for events	Limited: point clustering for events (in combination with scikit-learn)	Limited: similar trajectory search (spatially) and spatiotemporal point search along a trajectory

4 Conclusions and Outlook

We have discussed a four-step EDA workflow for movement data exploration and the different characteristics of movement data that should be considered. Afterwards, we presented three open source technology stacks for the exploratory analysis of movement data to address the lack of guidance for performing movement data exploration using openly available tools. The presented stacks cover the EDA steps to varying degrees, as summarized in Table 1.

While QGIS, PostGIS, and Pandas—by default—run on a single machine, big data tools like GeoMesa have been designed for distributed processing from the start and, therefore, are not limited to a single machine. However, the distinction is not so clear-cut since, for example, it is possible to set up distributed PostGIS databases, and parallel processing of spatial queries is under development (Ramsey 2019). Similarly, Dask (2020) provides distributed computation tools for Pandas.

With the ongoing development of different data analysis environments and the ever-increasing availability and application of tracking solutions to produce continually growing datasets, we can expect reasonable progress of EDA tools for movement data in the future. For example, while concepts for measures between groups of trajectories do exist, they have not been implemented into any openly available tools yet. However, the heterogeneity of applications and datasets presents a major challenge for the development of general-purpose movement data exploration tools.

Numerous scientific and technical challenges remain to be solved to address open questions, such as how to ensure privacy in EDA settings without compromising the utility of the data, how to efficiently visualize large movement datasets, or how to best model trajectories in software libraries for data analysis. Furthermore, to reach a wide audience, including practitioners without programming skills, it will be necessary to develop intuitive graphical user interfaces for movement data exploration. EDA templates, like the Jupyter notebook template provided by MovingPandas (https://exploration.movingpandas.org) can be a first step to lower the entry barrier. Future steps should include integrating more functionality into desktop GIS like QGIS, for example, by extending the Trajectools plugin (Graser 2020b).

Acknowledgements This work was supported by the Austrian Federal Ministry for Transport, Innovation and Technology (BMVIT) within the programme "IKT der Zukunft" under Grant 861258 (project MARNG).

References

Andrienko G, Andrienko N, Bak P, Keim D, Wrobel S (2013) Visual analytics of movement. Springer Science & Business Media, Berlin

Apache Software Foundation (2019a) Apache Hadoop. https://hadoop.apache.org. Accessed 15 Nov 2019

Apache Software Foundation (2019b) Apache Zeppelin. https://zeppelin.apache.org. Accessed 15 Nov 2019

Dodge S, Weibel R, Ahearn SC, Buchin M, Miller JA (2016) Analysis of movement data. Int J Geogr Inf Sci 30(5):825–834

Dask Development Team (2020) https://dask.org. Accessed 17 Jan 2020

Folium Development Team (2019) https://github.com/python-visualization/folium. Accessed 15 Nov 2019

GeoServer Development Team (2019) GeoServer. open source geospatial foundation project. https://geoserver.org. Accessed 15 Nov 2019

Geotools Development Team (2019) Geotools. open source geospatial foundation project. https://geotools.org. Accessed 15 Nov 2019

Graser A (2011) Visualisierung raum-zeitlicher Daten in Geoinformationssystemen am Beispiel von Quantum GIS mit "Time Manager"-Plug-In. In: Proceedings of FOSSGIS2011, Heidelberg, Germany

Graser A (2018) Evaluating spatio-temporal data models for trajectories in PostGIS databases. GI_Forum J Geogr Inf Sci 1:16–33

Graser A (2019) MovingPandas: efficient structures for movement data in python. GI_Forum J Geogr Inf Sci 1:54–68

Graser A (2020a) Data science workflow framework. figshare. Figure. https://doi.org/10.6084/m9.figshare.11638368.v1

Graser A (2020b) Trajectools plugin. QGIS plugin repository. https://plugins.qgis.org/plugins/processing_trajectory. Accessed 31 Jan 2020

Graser A, Schmidt J, Roth F, Brändle N (2017) Untangling origin-destination flows in geographic information systems. Inf Visual 18(1):153–172. https://doi.org/10.1177/1473871617738122

Hunter JD (2007) Matplotlib: a 2D graphics environment. Comput Sci Eng 9(3):90–95

Hughes JN, Annex A, Eichelberger CN et al (2015) Geomesa: a distributed architecture for spatio-temporal fusion. Geospat Inf Fusion Motion Video Anal V 9473:94730

Joo R, Boone ME, Clay TA et al (2019) Navigating through the R packages for movement. arXiv preprint arXiv:1901.0593

Jung C (2019) Offline-MapMatching—a QGIS plugin for matching a trajectory with a network. AGIT Journal für Angewandte Geoinformatik 5:156–163

Kelley B, Pérez F, Granger B, Bussonnier M, Frederic J, Kelley K, Hamrick J, Grout J, Corlay S, Ivanov P, Avila D, Abdalla S, Willing C (2016) Jupyter notebooks—a publishing format for reproducible computational workflows. In: Positioning and power in academic publishing: players, agents and agendas: proceedings of the 20th international conference on electronic publishing (ELPUB). Amsterdam. IOS Press, pp. 87–90

Leaflet Development Team (2019) https://leafletjs.com. Accessed 15 Nov 2019

Long JA, Weibel R, Dodge S, Laube P (2018) Moving ahead with computational movement analysis. Int J Geogr Inf Sci 32(7):1275–1281

McKinney W (2010) Data structures for statistical computing in python. Proc 9th Python Sci Conf 445:51–56.

Pappalardo l, Simini F, Barlacchi G, Pellungrini R (2019) scikit-mobility: a Python library for the analysis, generation and risk assessment of mobility data. arXiv preprint arxiv:1907.07062

PostGIS Development Team (2019) PostGIS. open source geospatial foundation project. https://postgis.org. Accessed 15 Nov 2019

PyViz developers (2019) hvPlot. https://hvplot.pyviz.org. Accessed 15 Nov 2019

QGIS Development Team (2019) QGIS geographic information system. Open source geospatial foundation project. https://qgis.osgeo.org. Accessed 15 Nov 2019

Ramsey P (2019) Waiting for PostGIS 3: parallelism in PostGIS. https://info.crunchydata.com/blog/waiting-for-postgis-3-parallelism-in-postgis. Accessed 17 Jan 2020

Schwalb-Willmann J (2019) MoveVis. R package. https://movevis.org. Accessed 15 Nov 2019

Tukey JW (1977) Exploratory data analysis. Addison-Wesley, Reading.

WHO (2018) Global status report on road safety 2018. Tech. rep., World Health Organization, Geneva. https://apps.who.int/iris/bitstream/handle/10665/276462/9789241565684-eng.pdf. Accessed 10 Nov 2019

Zaharia M, Chowdhury M, Franklin MJ et al (2010) Spark: cluster computing with working sets. HotCloud 10(10–10):95

Zheng Y, Xie X, Ma WY (2010) GeoLife: a collaborative social networking service among user, location and trajectory. IEEE Data Eng Bull 33(2):32–39