Clustering moving object trajectories: Integration in CROSS-CPP analytic toolbox

Alberto Herranz 1*, Juan-Ignacio Caballero 2, Albert Zilverberg 3, Christian Wolff 3, Alejandro Rodríguez-Gonzalez 1,2, Ernestina Menasalvas 1,2

1 Universidad Politécnica de Madrid, Centro de Tecnología Biomédica, 28223 Pozuelo de Alarcón, Madrid, Spain
2 Universidad Politécnica de Madrid, Escuela Técnica Superior de Ingenieros Informáticos, 28660 Boadilla del Monte, Madrid, Spain
3 ATB Bremen, Germany.

* Correspondence: abherranz@upm.es, juanignacio.caballero@upm.es, zilverberg@atb-bremen.de, wolff@atb-bremen.de, alejandro.rg@upm.es, ernestina.menasalvas@upm.es

Version March 24, 2021 submitted to Journal Not Specified

Abstract: Mobile devices equipped with sensor are generating an amount of geospatial related data that properly analyzed can be used for future applications. In particular, being able to establish similar trajectories is crucial to analyze events on those points common in the trajectories. CROSS-CPP is an European project which main aim is to provide tools to store data in a data market and to have a toolbox to analyze the data. As part of these analytic tools, a set of functionalities have been developed to cluster trajectories. Based on previous work on clustering algorithms we present in this paper Quickbundels algorithm adaptation to trajectory clustering algorithm. Experiments using different distance measures show that Quickbundles outperforms spectral clustering using different distance metrics, being the geodesic distance the one that provides the best results.

Keywords: trajectory mining; mobile data; clustering; anomaly detection.

1. Introduction

Nowadays there is a growing number of devices connected to internet and it is expected an exponential rise along with the development of 5G communication standards. The availability of data and the increase of computation capacity have boosted the proliferation of new applications. Geolocation services have attracted great interest due to the easy collection of geospatial data through all kind of devices. This type of data can proceed from taxi services, customer to customer geolocation based e-commerce, electrical micro-mobility (bikes and scooters), wearable (specially health focused), . . . . These new services have realised the relevance of analysing the moving patterns of their customers as a key characteristic to improve their value proposition and competitiveness. Obtaining knowledge from moving agents data has applications such as routes optimization, prediction of traffic, customized trips or fleets management in the mobility field. Additionally, the maturity of the mentioned services is impacting the mobility in the cities. The study of citizens moving patterns may push forward more customized transportation and other public services.

The surge in data from moving objects faces several challenges. One of them is how to extract useful insights of the data. Clustering of trajectories refers to grouping them based on a similarity measure. It is a task that serves as a exploratory step in the discovery of patterns in moving objects and enables to synthesize the information by establishing a representative trajectory for each cluster. By so, it serves as a base where to start deeper analysis about the characteristics of the trajectories.
A second challenge is related with quality assurance and trust. In the diversity and randomness typical of the data of moving objects it is vital to establish the data that corresponds to real behaviours and which is due to errors in the sensors or systems. This task is referred as outlier/anomaly detection. It is domain specific and it is also used for detection of fraudulent activities, since these activities usually fall apart from expected common behaviour.

Another challenge in the application of trajectories data analysis is the development of user-friendly tools that enable the development of services to extract knowledge from integrated data sources. CROSS-CPP \(^1\) is a project which main aim is the development of a system for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features. Its purpose is to offer new cross sectorial services and focus on the commercial confidentiality, privacy and ethical issues using a context sensitive approach. In the framework of this project a toolbox has been developed to provide the analysis services. The toolbox includes algorithms for different machine learning tasks both on stream and batch mode. In particular, we focus in this paper on the clustering of moving objects.

Quickbundles \(^2\) is known to be a clustering algorithm for tractography segmentation that has shown good performance in the health domain. Tractografies, same as moving-objects trajectories may be defined as streamlines. Thus, we propose in this paper to adapt Quickbundles for trajectory clustering and to integrate it into CROSS-CPP analytic toolbox. The adaptation is not straightforward and some challenges have to be faced: i) Quickbundles assumes streamlines being defined by a settled number of points, as a consequence, in this paper we solve the problem to define different trajectories under the same number of points; ii) Distance among streamlines: the measure of the distances is adjusted for moving objects trajectories.

The paper presents the results of the experiments carried out with two datasets: i) Porto Kaggle dataset\(^2\) and ii) CROSS-CPP dataset\(^3\). We will also show how the integration looks like in CROSS-CPP and how it can be accessed. Consequently, the main contributions of the paper are as follows: i) The adaption of Quickbundles algorithm for mobile data, ii) The analysis of its performance in real trajectories datasets and iii) the integration into the CROSS-CPP environment.

The rest of the paper has been organized as follows: Section 4 reviews the advances of other similar approaches in the literature. In section 5 the proposed solution is detail and the implementation guidelines are given. In section 6 the results of experiments using the well known taxi trajectories and the data extracted from CROSS-CPP databases are shown. Section 8 presents the pros and cons of the presented approach. To end with, section 9 presents the conclusions as well as the future research lines.

2. Preliminaries

Prior to presenting the approach to cluster trajectories some preliminary definitions are introduced.

2.1. Definition of trajectory

A trajectory is the path followed by a moving object. Formally, a trajectory \(s\) is defined as a chronological sequence of multi-dimensional locations which is denoted by \(s = P_1, P_2, ..., P_n\) being \(n\) the number of points. Each point \(P_n\) is represented as \(Location_j\) tuple at time \(T_j\). \(^2\)

2.2. Distance measurements

The concept of distance among trajectories refers to the criteria to measure the similarity of trajectories. In this paper the similarity focuses on the geometric patterns of trajectories. Two trajectories

\(^1\) https://www.cross-cpp.eu/
\(^2\) https://www.kaggle.com/crailtap/taxi-trajectory
\(^3\) Due to privacy constrains the dataset is not publicly available
are identical if they have the same length and they are defined by the same positions independently of
the time. This kind of similarity is well suited for analysis of routes in cities and countries: commuting
routes from homes to work places, hotspot areas and holidays road trips.

3. QuickBundles and Minimum Direct-Flip similarity metric

QuickBundles (QB) and Minimum Direct-Flip (MDF) [1] similarity metric were developed for
simplification of tractography data, this is a 3D modeling technique that represents nerve tracts. In
this section, the similarity metric MDF and the QB clustering algorithm are explained.

MDF, see equation 1, is suitable only when the trajectories are defined by the same number of
points. The metric has no preferred orientation.

\[
\begin{align*}
    d_{direct}(s, t) &= d(s, t) = \frac{1}{K} \sum |s_i - t_i|, \\
    d_{flipped}(s, t) &= d(s, t^F) = d(s^F, t) \\
    MDF(s, t) &= \min(d_{direct}(s, t), d_{flipped}(s, t))
\end{align*}
\] (1)

Where \( |s_i - t_i| \) denotes the Euclidean distances between two points, \( s \) is a trajectory and \( t \) is the
mean of the Euclidean distance. Figure 1 provides an insight of the working of the metric.

![Figure 1. Operation of MDF.](image)
QB is a simple clustering algorithm with linear time performance with respect to the number of trajectories. In the following lines the main ideas and the algorithm are presented, for further detail see [1].

QB computing times depend linearly to the number of samples because i) it uses MDF with linear complexity; and ii) there is no reassignment of the trajectories clustering. This means trajectories are evaluated only once when performing the clustering and opposed to K-means, where there is amalgamating phase. Clusters are defined by a triplet $c = (I, h, n)$ where $I$ is a list of indexes of the trajectories in the cluster, $n$ is the number of trajectories in the cluster, and $h$ is the trajectories sum. $h$ is a $K \times 2$ matrix which can be updated for new trajectories. (see 2)

$$h = \sum_{n=1}^{n} s_i \quad (2)$$

This way, the centroid of the cluster is defined by $v$:

$$v = \frac{h}{n} \quad (3)$$

Among the strengths of this clustering algorithm we have its mentioned lineal complexity, the easiness to interpret the clusters and the versatility provided by changing the number of points that define each trajectory and the threshold value to consider trajectories are similar. All of these become very useful in the exploration of a dataset of trajectories.

4. Related work

As it has been mentioned above, the proposed approach adapts the algorithm presented in [1] for clustering and performs anomaly detection based on that. Consequently, in what follows we review previous approaches for trajectory clustering and detection of anomalies.

4.1. Clustering of trajectories

Trajectory clustering is a common practice in trajectories data mining, it consists in grouping the trajectories based on a similarity measure. The nature of trajectories as multidimensional data enables different clustering approaches that may be domain or application specific.

4.1.1. Distances between trajectories

An important aspect in the clustering of trajectories is the computing of the distances among them. The first point to be accounted is that moving objects travel in the Earth in most of the cases. So then the distance metric may consider the curvature of its surface. This is the case for the great circle and ellipsoids geodesic distance. In other cases the effect of the curvature of the Earth is negligible, then the most used metric is the euclidean, but there are others like the Hausdorff or the longest common subsequence. [2].

Distances between two points on the Earth: The shortest distance between two points on the surface of the Earth is referred as the geodesic distance. There are several formulas to calculate the geodesic with lesser or greater error depending on the position in the Earth’s surface. In this regard the most simple model is to approximate the Earth as a sphere, then the geodesic distance is also called the great-circle or orthodromic distance. Another approach is to consider an ellipsoidal model of the Earth, then again there are several models of ellipsoids with variable error depending on the position. The WGS-84 ellipsoid is the most globally accurate. Among the algorithms to compute the geodesics Karney’s stands out [3] due to the implementation of various geodesic capabilities into a single library. In many cases trajectories are constrained to spatial locations of the size of an area small enough to assume it is flat, so then, they neglect the curvature of the planet. It is this case the most common in the literature with examples like taxi routes analysis, see [4] for a review of different approaches.
4.1.2. Spatial clustering of trajectories

Once the distances among trajectories are computed and stored, different clustering algorithms may be applied. Each of them offer pros and cons and the best solution is domain specific.

This project focuses on geometric properties of the trajectories, defined by characteristics like spatial distribution, proximity or length. The predominant algorithms are those based on the density such as DBSCAN [5] or K-means [6], where the proximity is the most important feature. One of the drawbacks of K-means is its tendency to form spherical clusters, which is inadequate for clustering several streamlines as in the present case.

For clusters with irregular, or non-spherical shapes one of the most efficient algorithm is the spectral clustering [7], it uses the eigenvalues, also referred as the spectrum.

Quickbundles, presented before at 3 is a clustering algorithm developed to reduce the dimensionality of brain streamlines datasets, provided by tractography. It stands out by its linear complexity with the number of streams and its interpretability.

5. Quickbundles preparation stage

5.1. Data preparation

Quickbundles is an algorithm to overcome the complexity of segmentation of the brain fibers. The proposed algorithm for clustering trajectories is based on the QuickBundles method for tractography simplification. As it has been mentioned the clustering algorithm requires of the following:

- All trajectories to be defined by the same number of points.
- A distance threshold to specify the maximum value to which two trajectories are clustered together.

5.2. Distance measures

As it was mentioned in the related work the geodesic distance is the shortest distance between two points located at the surface of the Earth and there are several models to compute this distance:

- Harvesine: Being the central angle (\(\Theta\)) between two points in an sphere:
  \[
  \Theta = \frac{d}{r}
  \]
  where \(d\) is the distance between two points along a great circle of the sphere and \(r\) is the radius of the sphere.

The Haversine (\(hav(\Theta)\)) is computed directly from the latitude and longitude of the two points:

\[
\text{hav}(\Theta) = \text{hav}(\varphi_2 - \varphi_1) + \cos \varphi_1 \cos \varphi_2 \text{hav}(\lambda_2 - \lambda_1)
\]

where \(\varphi_2, \varphi_1\) are the latitude of point 1 and 2. \(\lambda_2, \lambda_1\) is the longitude of points 1 and 2.

The haversine function computes half a versine of the angle \(\theta\):

\[
\text{hav}(\theta) = \sin^2 \frac{\theta}{2}
\]

Then the distance between the two points:

\[
d = r \text{ archav}(\text{hav}(\Theta)) = 2r \arcsin(\sqrt{\text{hav}(\Theta)})
\]

\[
d = 2r \arcsin(\sqrt{\text{hav}(\varphi_2 - \varphi_1) + \cos \varphi_1 \cos \varphi_2 \text{hav}(\lambda_2 - \lambda_1)})
\]
Great circle: This formula was manually implemented and it is defined:

\[ d = R \arccos (\sin \varphi_2 \sin \varphi_1 + \cos \varphi_2 \cos \varphi_1 \cos (\lambda_2 - \lambda_1)) \]  

(8)

where the \( d \) is the distance, \( R \) is the Earth radius, a constant with \( R = 6371 \) Km. Same notation as before for the latitude (\( \varphi \)) and longitude (\( \lambda \)) of each point.

Euclidean: By neglecting the curvature of the Earth, the Euclidean distance between the two points is defined:

\[ d = \sqrt{(\varphi_2 - \varphi_1)^2 + (\lambda_2 - \lambda_1)^2} \]  

(9)

Geodesic WS84: In this case the model of the Earth is an ellipsoid, in particular the WS84 is the most globally accurate model. The used implementation is the included in the GeoPy library, which uses the method given by Kerney [3].

5.3. Parameter setting

Two clustering algorithms: Quick bundles and spectral were compared. In order to validate results, 10 different experimentations with batches of 2000 trajectories were conducted for each algorithm and distance metric. This is, a total of 80 experiments were run (10 per pair algorithm-distance metric). In the end, the mean Silhouette Coefficients of each set of 10 experiments was taken as final results.

To build each experimentation batch, a different approach was set depending of the data source:

- Kaggle data: since there were 1704757 trajectories, each experiment run for an algorithm-distance metric pair comprised 2000 different trajectories each time, guaranteeing no repetition of trajectories in each experiment.
- Cross-CPP data: data from only 5852 remained for experimentation after filtering, each experiment was run over a subset of 2000 randomly selected trajectories, unavoidably repeating trajectories between experiments.

In each experiment, the hyperparameters of QuickBundles were adjusted via a 10-fold cross-validation approach. The adjusted hyperparameters were:

- Resampling factor: number of coordinates to which each trajectory is going to be resampled (as the algorithm requires all trajectories to have the same number of coordinates).
- Neighboring threshold: distance at which two trajectory points are considered to be neighbors.

It is to note that no hyperparameters were tuned for the Spectral Clustering algorithm.

5.4. Validation metrics

The silhouette was the used validation metric. Based on the definition of Rousseeuw [8]. Being \( i \) a data point in the cluster \( C_i \). Being \( a \) the mean distance between \( i \) and all other points in the cluster:

\[ a(i) = \frac{1}{|C_i|} \sum_{j \in C_i, j \neq i} d(i, j) \]  

(10)
The mean dissimilarity of point \( i \) to some cluster \( C_k \) is the mean of the distance from \( i \) to all points in \( C_k \). Being \( b \) the smallest distance of \( i \) to a cluster different than \( C_i \):

\[
b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)
\]

(11)

Then the silhouette value for one point \( i \) is given by:

\[
s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \text{ if } |C_i| > 1
\]

(12)

The term silhouette coefficient is defined as the maximum value of the mean \( s(i) \) over all data of the entire dataset:

\[
SC = \max_k t(k)
\]

(13)

where \( t(k) \) represents the mean \( s(i) \) over all data for the number of clusters \( k \).

6. Experiments

6.1. Data set description

The experiments presented in this section are performed using the Kaggle dataset of taxi trajectories. As described in the web it contains trajectory information from 01/07/2013 to 30/06/2014 for 442 taxis running in the city of Porto, in Portugal. In total 1704757 trajectories are available.

Each register contains 9 features, described as follows:

- **TRIP_ID**: (string) It contains an unique identifier for each trip.
- **CALL_TYPE**: (char) It identifies the way used to demand the taxi service. It may contain one of three possible values:
  - A: if the trip was dispatched from the central.
  - B: if the trip was demanded on a specific stand.
  - C: Otherwise. i.e. a trip demanded on a random street.
- **ORIGINCALL**: (integer)
- **TAXI_ID**: (integer)
- **TIMESTAMP**: Unix Timestamp (in seconds). It identifies the trip’s start;
- **DAYTYPE**: (char)
- **MISSING_DATA**: (Boolean)
- **POLYLINE**: (String): It contains a list of GPS coordinates (i.e. WGS84 format) mapped as a string. The beginning and the end of the string are identified with brackets (i.e. [ and ], respectively). Each pair of coordinates is also identified by the same brackets as [LONGITUDE, LATITUDE]. This list contains one pair of coordinates for each 15 seconds of trip. The last list item corresponds to the trip’s destination while the first one represents its start. Some trips have missing data points, then the MISSING_DATA columns is TRUE.

For this paper the TIMESTAMP and POLYLINE features were used.

6.2. Results

The following table resumes the average of the silhouette coefficient for the clustering algorithms and the distance measures in the Kaggle dataset.

---

4 [https://www.kaggle.com/crailtap/taxi-trajectory](https://www.kaggle.com/crailtap/taxi-trajectory)
Table 1. Table Silhouette results for clustering algorithms for Kaggle dataset

| Distance metric | Quickbundles | Spectral |
|-----------------|--------------|----------|
| Harvesine       | 0.753        | 0.479    |
| Great Circle    | 0.772        | 0.332    |
| Euclidean       | 0.732        | 0.514    |
| Geodesic WS84   | 0.812        | 0.601    |

One can observe that clearly Quickbundles algorithm outperforms Spectral clustering. Observe that the geodesic distance clearly yields the best results even when points to calculate distances in the dataset are not so much apart.

7. Integration and experiments CROSS-CPP

As it was mentioned in section 1, the CROSS-CPP is an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features. Consequently, in this section we will firstly review the process that has to be followed to harvest data and make them available for the data analytics toolbox to show how the proposed clustering algorithm has been integrated in Cross-CPP. Once we describe the integration on the CROSS-CPP framework, we will analyze how to run it and which have been the results of different executions using real datasets.

In what follows, the process performed to calculate clustering of trajectories of the data available at the platform is analyzed. The process is divided into two phases: Obtaining access to the data (data harvesting) and performing the desired analysis, clustering in this case.

7.1. Data harvesting

The accessing data phase started by logging in the Agora website at the "Home" tab. It was chosen vehicles origin data. Figure 2 shows the screen where basic statistics are displayed for the selected data:

![Figure 2. Data selection](image)

The process continues by moving to the "Data Discovery" tab and selecting the type of signal to be used. Filtering by vehicles, each signal contains the values for a vehicle feature, i.e., vehicle speed, latitude, temperature . . . . This information is contained in a channel with an unique identifier. The "position", channel number two, signal was chosen, containing the latitude and longitude as it can be seen on 3.
Once the channel is selected, no geographical or temporal filters have been added in the example. Then the platform provides useful insights in the form of heatmap for geographical reference and slide bar for the time distribution of the trajectories. Furthermore, general statistics are provided as well (see figure 5).

| Actions | Signal | Signal Type | Unit | Related channels |
|---------|--------|-------------|------|------------------|
|         | Vehicle speed | numeric | km/h | Vehicle Speed |
|         | Latitude | numeric | ° | Position (Latitude Longitude) |
|         | Longitude | numeric | ° | Position (Latitude Longitude) |
|         | Engine RPM | numeric | RPM | Engine RPM |
|         | ABS (on/off) | numeric | On/Off | ABS (on/off) |
Figure 4. Heatmap of trajectories dataset and time distribution

Figure 5. Insights of the trajectories data
The analytic phase starts upon the approval of the data request. In the "Toolbox" and trajectories tab, the above mentioned data request can be selected. It provides the option of adding temporal or geographical filters again, but none were used. Then in the analysis option "Clustering" was selected as it can be observed in figure 6.

Figure 6. Illustration of the main page to perform analytics
7.2. Experiments performed in CROSS-CPP

7.2.1. Dataset description
The data original unprocessed data included 6094 different trajectories, from which 742 trajectories with less than 100 coordinates were removed. In conclusion, 5852 trajectories were taken into account for the final experimentation.

7.3. Results
The following table resumes the average of the silhouette coefficient for the clustering algorithms and the distance measures in the CROSS-CPP data.

| Distance metric | Quickbundles | Spectral |
|-----------------|--------------|----------|
| Harvesine       | 0.685        | 0.378    |
| Great Circle    | 0.742        | 0.401    |
| Euclidean       | 0.698        | 0.442    |
| Geodesic        | 0.753        | 0.396    |

8. Discussion
In this paper we have presented results of the integration Quickbundles algorithm for the moving objects trajectories clustering in the CROSS-CPP platform. Two main aspects have to be discussed: i) Integration issues, ii) clustering algorithm efficacy and efficiency.

The CROSS-CPP marketplace makes possible for data providers to store data so that service providers can use these data to generate applications based on the knowledge extracted from these data. In order to extract the knowledge from the data, the analytic toolbox has been developed and integrated. One of the main advantages of the analytic toolbox is its flexibility to include new functionalities, in particular in this paper we have shown how clustering for moving objects has been integrated. More precisely Quickbundles algorithm has been adapted from health domain to calculate clustering of trajectories. In the paper we have shown the end-to-end process from data harvesting to data analysis.

As it has been shown the clustering has been performed in batch mode. However, it is important to note that the CROSS-CPP analytic toolbox allows for data-streaming mode. Quickbundles algorithm has been proposed in this paper for trajectory clustering. We chose this instead of center-based clustering algorithms such as K-means due to the fact that spherical cluster are not appropriate for trajectory clustering, for that reason we have compared with spectral clustering and difference distance measures have been used. The results show that quickbundles outperforms spectral. This behaviour is coherent in both datasets: the Porto Kaggle and the CROSS-CPP data.

Despite performance (time-consumption) have not been deeply analysed the spectral algorithm even three times faster for the Quickbundles. Obviously, the parameter setting required for Quickbundles does not help in time performance.

In relation with the distance measure. Despite, not observing a high difference in silhouette coefficient among distance measures in both datasets. We can say that geodesic WS84 yields a better result. As it was mentioned before the geodesic distance uses the WS84 ellipsoid model what helps in minimizing the global error. Note that this distance improved performance even the case of Porto where the difference of distances is not that big.

9. Conclusions
In this paper we have presented results of the integration Quickbundles algorithm for the moving objects trajectories clustering in the CROSS-CPP platform. It has been successfully integrated and its
available in the CROSS-CPP project. The reason to be chosen was its outperformance with relation to other algorithms of the literature. In particular, we have verified this fact when comparing its behaviour with the spectral clustering algorithm.

Future work includes the inclusion of streaming clustering algorithms.

Author Contributions: ABH and JIC are responsible for design and implementation issues as well as drafting the manuscript. CW and AZ are the coordinator of the CROSS-CPP project and consequently, responsible of the coordination of all of the work performed. Their contribution in reviewing the paper has been crucial. EM and AR are responsible for coordination of the modelling of the algorithms, validation steps and final writing of the paper.

Funding: This research was partially funded by CROSS-CPP EU grant number 780167.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Garyfallidis, E.; Brett, M.; Correia, M.M.; Williams, G.B.; Nimmo-Smith, I. QuickBundles, a Method for Tractography Simplification. *Frontiers in Neuroscience* **2012**, *6*. doi:10.3389/fnins.2012.00175.

2. Yuan, G.; Sun, P.; Zhao, J.; Li, D.; Wang, C. A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review* **2017**, *47*, 123–144. doi:10.1007/s01462-016-9477-7.

3. Karney, C.F.F. Algorithms for geodesics. *Journal of Geodesy* **2013**, *87*, 43–55. doi:10.1007/s00190-012-0578-z.

4. Castro, P.S.; Zhang, D.; Chen, C.; Li, S.; Pan, G. From taxi GPS traces to social and community dynamics: A survey. *ACM Computing Surveys* **2013**, *46*, 1–34. doi:10.1145/2543581.2543584.

5. Ester, M.; Kriegel, H.P.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *p. 6.*

6. Lloyd, S. Least squares quantization in PCM. *IEEE Transactions on Information Theory* **1982**, *28*, 129–137. doi:10.1109/TIT.1982.1056489.

7. von Luxburg, U. A Tutorial on Spectral Clustering. *arXiv:0711.0189 [cs] 2007*. arXiv: 0711.0189.

8. Rousseeuw, P.J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* **1987**, *20*, 53–65. doi:10.1016/0377-0427(87)90125-7.

© 2021 by the authors. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).