Efficient and Effective 4D Trajectory Data Cleansing

TAN Xin[1,2], *SUN Xiaoqian*[1,2], *ZHANG Chunxiao*[1,2], *WANDELT Sebastian*[1,2*]

1. National Key Laboratory of CNS/ATM, Beihang University, Beijing 100191, P.R. China;

2. School of Electronic and Information Engineering, Beihang University, Beijing 100191, P.R. China

(Received 12 March 2020; revised 5 April 2020; accepted 10 April 2020)

Abstract: As the rapid development of aviation industry and newly emerging crowd-sourcing projects such as Flightradar24 and FlightAware, large amount of air traffic data, particularly four-dimension (4D) trajectory data, have become available for the public. In order to guarantee the accuracy and reliability of results, data cleansing is the first step in analyzing 4D trajectory data, including error identification and mitigation. Data cleansing techniques for the 4D trajectory data are investigated. Back propagation (BP) neural network algorithm is applied to repair errors. Newton interpolation method is used to obtain even-spaced trajectory samples over a uniform distribution of each flight's 4D trajectory data. Furthermore, a new method is proposed to compress data while maintaining the intrinsic characteristics of the trajectories. Density-based spatial clustering of applications with noise (DBSCAN) is applied to identify remaining outliers of sample points. Experiments are performed on a data set of one-day 4D trajectory data over Europe. The results show that the proposed method can achieve more efficient and effective results than the existing approaches. The work contributes to the first step of data preprocessing and lays foundation for further downstream 4D trajectory analysis.

Key words: 4D trajectories; data cleansing; outlier detection; repair

CLC number: U8 **Document code:** A **Article ID:** 1005-1120(2020)02-0288-12

0 Introduction

The aviation industry has been developed rapidly in recent years, and the air transportation system is facing with huge challenges for both management and technologies. The emergence and dissemination of data science provide powerful tools to analyse and manage the air transportation system. Air traffic data, especially four-dimension (4D) trajectory data, are used for various analysis tasks, such as aircraft conflict detection and resolution[1], airspace congestion management[2], and air navigation route optimization[3]. The 4D trajectories refer to a series of points composed of the coordinates of longitude, latitude, altitude and the corresponding timestamp of aircraft in the air. With the emergence of the 4D trajectory data, airborne calculations can be used to predict the aircraft sequence at the intersection of the busy or congested airspace. It can facilitate the decision-making process of the air traffic controllers.

As the driving force of scientific and technological innovation, "data" account for a rising proportion of assets, and increasingly become another major factor of production after "land" and "capital". Data cleansing is the prerequisite for the 4D trajectory data analysis. For obtaining more accurate and reliable results, outliers and unreliable data must be cleansed beforehand. Otherwise, it will prolong the data processing time and efforts, and mislead the final result. In the field of air transportation, research focusing on data cleansing is still rare[4-6], and the percentage of abnormal data is not high. However, they are widely distributed[7], almost throughout every trajectory, thereby affecting the reliability of results negatively. The common abnormal data includes data timestamp error[4,7] and missing data, such as longitude and latitude, identification infor-

*Corresponding author, E-mail address: wandelt@buaa.edu.cn.

How to cite this article: TAN Xin, SUN Xiaoqian, ZHANG Chunxiao, et al. Efficient and effective 4D trajectory data cleansing[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2020, 37(2):288-299.

http://dx.doi.org/10.16356/j.1005-1120.2020.02.010

mation and duplicated data[4-5,7]. In addition to these anomalies, there are also some logical errors that are hard to be identified intuitively, such as data jumping[5,7-8] , which indicates too large distance, too long time gap, or too fast speed, etc. Therefore, there is an urgent requirement for 4D trajectory data cleansing method.

Research on data cleansing mainly focuses on improvement and management of the data quality. Although there are significant achievements, the data quality control and data cleansing methods need to be improved continually[9]. Basic principles of data cleansing techniques include identifying determinants which affect the data quality, define the cleansing requirements, and establish the cleansing model[10-13].

Data cleansing is the very first step in the data preprocessing for trajectory data analysis and applications. However, most papers did not provide sufficient justifications for the data cleansing or data preprocessing before trajectory analysis[5,8]. To check the integrity of ADS-B data, Andrisani et al.[5] applied a suite of Kalman filters to smooth noise, identified and suppressed erroneous data, coasted between data dropouts, and provided the current best state estimates. Experiments were performed on the simulated ADS-B data signals and demonstrated that the approach was promising to data integrity check. The 4D trajectory data cleansing was applied by Patroumpas et al.[8], who developed one-pass heuristics to eliminate inherent noise. They provided reliable trajectory representations and presented various bounds for trajectory error detection. Since it handled trajectories online and discarded errors directly, it was inapplicable with data offline. To manage ADS-B data collection efficiently and integrate with other flight related data, Garcia et al.[14] devised AIRPORTS DL with Data Lake architecture to reconstruct gate-to-gate trajectories and to derive parameters, such as the predictability or the fuel consumption. It discarded useless messages or aligned field values to satisfy the AIRPORTS data model, and determined trajectories when different flights use the same call sign as well. However, it required extra information, such as the history trajectory data to detect these data. The traffic library for the Python programming language, introduced by Olive et al.[15], presented how to access different sources of data, leverage processing methods to clean, filter, clip or resample trajectories, and compare trajectory clustering methods on a sample dataset of trajectories above Switzerland. The paper handled missing data, slicing, querying or resampling with Pandas library, but it is not usually sufficient for a high requirement of data cleansing.

Back propagation (BP) neural network algorithm is widely used in various fields, such as aviation industry[16], manufacturing industry[17], engineering[18], medical industry[19], geology[20] etc. Lin et al.[21] established a sensor error correction model which combined particle swarm optimization (PSO) with the BP neural network algorithm to reduce nonlinear characteristics and improve test accuracy of the system. The BP neural network has three or more than three layers, including input layer, hidden layer, and output layer. The upper and lower layers are connected completely. There is no connection between each neuron in the same layer. To construct a BP neuron network, first random parameters of each layer are set; and the output based oninput data and the initial parameters are calculated. Subsequently, following the direction of reducing the loss between the output and actual target, the connection from the output layer weights to the middle layer-by-layer is amended. Finally, the procedure returns to the input layer[22].

In this paper, a rich set of data cleansing methods are presented to deal with 4D trajectory data. The BP neural network is proposed to repair the errors detected by average speed between two adjacent points in the trajectories. Newton interpolation is used to filter out inconsistent data and to fix the frequency of the points in 4D trajectories. To reduce experimental complexity of further analysis while to maintain the characteristics of the trajectories, the unit cube sampling method is used to cut down data size. In addition, the density-based spatial clustering of applications with noise (DBSCAN) is used to identify outliers of trajectories. Experiments on one-day 4D trajectory dataset in European area are conducted. Results show that the proposed techniques can better cleanse the 4D trajectory data. The goal of this study is to clean data for the preparation of further data analysis. A framework for cleansing 4D trajectories is presented in the following section.

1 Methodology

As the recorded 4D trajectory data can be erroneous, the common abnormal data includes data timestamp error, missing data, duplicated data, and off-couse data[4-5,7-8]. Data cleansing is the very first step before further data analysis. Typical statistical outlier detection techniques are first used to detect the errors. Here, it is focused on the flight speed element. The data with high speed beyond the maximum flight speed will be fixed by applying BP neural network, or directly discarded if data are unnecessary. The 4D trajectory data of one flight may be recorded in 8 min, 12 min, or even longer. The sampling frequency is not fixed, in other words, sampling data is missing to some extent. In order to repair the inconsistent data, Newton interpolation is applied according to fixed time or fixed distance. Moreover, similar records are discarded, which are unimportant and occupy storage based on unit cube method. In addition to the aforementioned methods, clustering method to detect outliers of trajectories is also used.

1.1 4D trajectory model

The 4D trajectory data elements obtained in this paper include the information on aircraft number, time, latitude, longitude, altitude, and speed. A 4D trajectory is a sequence of 4D points[23-24]. There are three dimensions for space information and one dimension for time. The 4D point P is

$$P=(L,H,A,T) \tag{1}$$

where L is the latitude of the point P, H the longitude of point P, A the altitude of point P and T the time at point P. Then, a 4D trajectory $\boldsymbol{Tr}$ is defined by

$$\boldsymbol{Tr}=[P_1,P_2,\cdots,P_n] \tag{2}$$

A 4D trajectory dataset D is a collection of 4D trajectories, which is defined as follows

$$D=\{Tr_1,Tr_2,\cdots,Tr_m\} \tag{3}$$

1.2 BP neural network

The BP neural network is introduced to repair errors in this section. Firstly, the average speed between two adjacent points P_1 and P_2 is applied to detect the errors with the following formulations

$$C_{21}=\arccos(\cos(L_{p1})\cdot\cos(L_{p2})\cdot\cos(H_{p1}-H_{p2})+\sin(L_{p1})\cdot\sin(L_{p2})) \tag{4}$$

$$d_{12}=\frac{R\cdot C_{12}\cdot\pi}{180} \tag{5}$$

$$D_{12}=\sqrt{d_{12}^2+(A_1-A_2)^2} \tag{6}$$

$$v_{12}=\frac{D_{12}}{T_2-T_1} \tag{7}$$

where $R=6\,371.0$ km. L_{p1}, L_{p2} are the latitudes of P_1 and P_2; H_{p1}, H_{p2} the longitudes of P_1 and P_2; A_1, A_2 the altitudes of P_1 and P_2; C_{12} the radian between P_1 and P_2; d_{12} the great circle distance between P_1 and P_2; D_{12} the actual distance between P_1 and P_2; and v_{12} the average speed between points P_1 and P_2.

Since the speed of the current commercial plane is less than the speed of sound, if v_{12} is larger than 1 200 km/h, P_2 is taken as an error point. Then, we apply BP neural network to repair P_2 based on the correct points. Fig. 1 is the structure of BP neural network.

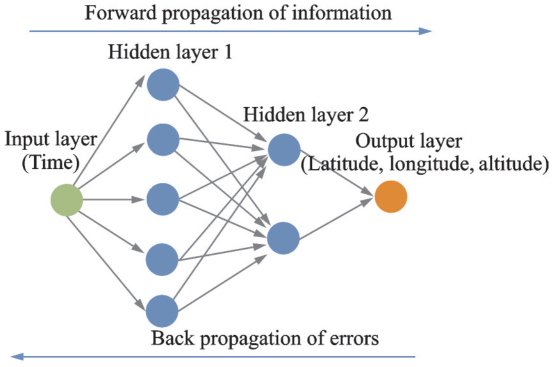

Forward propagation of information
Hidden layer 1
Hidden layer 2
Input layer
(Time)
Output layer
(Latitude, longitude, altitude)
Back propagation of errors

Fig.1 Neural network structure

The BP neural network includes input layer, hidden layer, and output layer. Given a network, there are N nodes and L layers. The activation function defined as the sigmoid function is

$$f(x)=\frac{1}{1-\mathrm{e}^{-x}} \tag{8}$$

The error of mean square function is used to describe the loss between the true value and the output value

$$E(y,y_-)=\frac{\sum_{i=1}^{n}(y-y_-)^2}{n} \tag{9}$$

where x is the input value, y the true value, and y_- the output value.

The input value of the ith neuron at the lth layer is $net_i^{(l)}$ and the output value of the ith neuron at the lth layer is $h_i^{(l)}$, the forward propagation is

$$net_i^{(l)}=\sum_{j=1}^{n}W_{ij}^{(l)}h_j^{(l-1)}+b_i^{(l)} \tag{10}$$

$$h_i^{(l)}=f(net_i^{(l)}) \tag{11}$$

where $W_{ij}{}^{(l)}$ is the layer connection value from the ith neuron of the lth to the jth neuron of the $(l+1)$th layer, $b_{\mathrm{i}}^{(l)}$ the offset of the ith neuron of the lth layer, and $f(\cdot)$ the activate function.

To update the weight and offset, the gradient descent functions of the error function are defined as

$$W_{ji}^{(l)}=W_{ji}^{(l)}+R_{\text{Learn}}\cdot\frac{\partial E(i)}{\partial W_{ji}^{(l)}} \tag{12}$$

$$b_{ji}^{(l)}=b_{ji}^{(l)}+R_{\text{Learn}}\cdot\frac{\partial E(i)}{\partial b_{ji}^{(l)}} \tag{13}$$

The parameter R_{Learn} is updated by a decay function.

$$R_{\text{Learn}}=\frac{R_{\text{Learn}}}{1+d\cdot n_{\text{step}}} \tag{14}$$

where d is the decay factor to update the learning rate. Noted that if the learning rate is too large, the model would be over-training. Otherwise, if it is too small, the optimization speed would be too slow.

The BP neural network with two hidden layers can implement any nonlinear mapping without limiting the number of hidden nodes. Therefore, four-layer neural network with two hidden layers is chosen in the proposed model in this paper.

The process of 4D trajectory data training based on the BP neural network is summarized as follows:

Step 1 Take one flight of 4D trajectory data, and detect and discard errors. Set the remained data as training dataset. Set nodes at each layer of the BP neural network.

Step 2 Set the activation function as sigmoid function. Define the value function MSE (y, y_-) representing the square sum of the output error.

Step 3 Set the learning rate. Lay down a feedback regulation.

Step 4 Choose time as input parameter. Set longitude, latitude and altitude as output separately. Feed data to train.

Step 5 Input errors of time detected in Step 1 to the trained neural network model. Collect the output which is the repaired data.

Step 6 Select the next BP neural network data, back to Step 1. Stop until all the trajectories are repaired.

1.3 Newton Interpolation with sliding window

Newton interpolation is one of the most popular interpolation methods. Data based on Newton interpolation with sliding window can reach high level accuracy, and computational efficiency can be gained as well[25].

Assume that x_0, x_1, x_2, x_3 are a set of independent variables that are not equal to each other, and $f(x)$ is a dependent variable. In this paper, the 4th-order Newton interpolation is applied to fix the frequency of points in trajectories.

$$f[x_i,x_j]=\frac{f(x_i)-f(x_j)}{x_j-x_i} \tag{15}$$

$$f[x_i,x_j,x_k]=\frac{f[x_i,x_k]-f[x_i,x_j]}{x_k-x_i} \tag{16}$$

$$f[x_i,x_j,x_k,x_l]=\frac{f[x_i,x_j,x_l]-f[x_i,x_j,x_k]}{x_l-x_i} \tag{17}$$

$$N_0=f(x_0) \tag{18}$$

$$N_1(x)=N_0+f[x_0,x_1]\cdot(x-x_0) \tag{19}$$

$$N_2(x)=N_1(x)+f[x_0,x_1,x_2]\cdot(x-x_0)\cdot(x-x_1)\cdot(x-x_2) \tag{20}$$

$$N_3(x)=N_2(x)+f[x_0,x_1,x_2,x_3]\cdot(x-x_0)\cdot(x-x_1)\cdot(x-x_2)\cdot(x-x_3) \tag{21}$$

where $f[x_i,x_j]$, $f[x_i,x_j,x_k]$ and $f[x_i,x_j,x_k,x_l]$ are the 2nd-order difference quotient, the 3rd-order difference quotient, and the 4th-order difference quotient, respectively. By following Eqs. (18)—(21), the 4th-order Newton interpolation $N_3(x)$ is obtained.

1.4 Unit cube method

Some of the 4D trajectory data is redundant. In order to save storage and not to destroy the data integrity, the process of sampling based on distance is applied. As shown in Fig. 2, a 4D trajectory includes a series of points in 3D spaces throughout the flight. There are some points that are redundant and the characteristics of the trajectory are not changed if they are removed. Therefore, the 3D space is cut with cubes in the same volume. The points are all put into their corresponding cubes. Then, select one point in each cube to represent points in it.

In Fig.2, blue and red points are the whole data. The first cube is set at the first point, and then the subsequent cubes are set one by one according to the trend of the data. Finally, all the points are filled into the respective cubes. A data point from each cube is selected, for example, the red are the points selected. The final data are the remained red points.

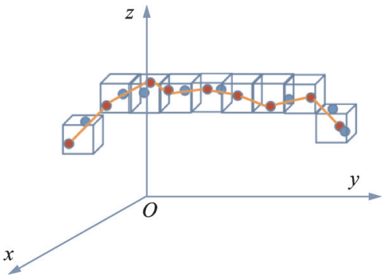


Fig.2 One demo of unit cube method

In this paper, we set the latitude to 0.5°, the longitude to 0.5°, and the altitude to 152.4 m as the cubic size. The size of the cube is set according to the requirement of data accuracy. In other words, if more precise data is required, the unit cube is supposed to be set smaller, while the size of the remained data would be larger.

1.5 Trajectory clustering

To filter noises of the trajectories or the outliers of trajectories, we apply the clustering method DBSCAN[26], which views clusters as points in the high density areas. In detail, DBSCAN clusters the points together (in ε Neighborhood with the at most *MinPts* samples). There are two parameters *pes* and *MinPts* for the clusters. The ε-neighborhood contains the points with the distance less than ε for a given point x_i in the dataset D, i. e., $N_\varepsilon(x_j)=\{x_i \in D|\mathrm{dis}(x,x_j)\leqslant\varepsilon\}$, *pes* represents the maximum distance between two samples for them which can be considered as in the same neighborhood. If the number in the neighborhood $N_\varepsilon(x_j)$ is more than *MinPts*, then a new cluster is generated. With the iteration of DBSCAN, the points are added to the clusters until all points in the dataset are labelled. Only spatial information of the trajectories is used in this section. When the trajectory is separated from other combined trajectories or consists of several sporadic points, it will be identified as noises.

The width of a victor way is 8 nm (14.8 km) according to provisions of the International Civil Aviation Organization (ICAO), and longitude and latitude are used to cluster and 0.1° is about 11 km. In our experiments, the parameter *pes* is set to 0.1 and the parameter *MinPts* to 20.

2 Results

2.1 Datasets and experimental setup

Despite their high value in aircraft surveillance, positional data streams are not error-free, particularly ADS-B messages relayed from aircraft. Spurious coordinates indicate impossible positions across a flight. Satellite transmission problems may lead to delayed or missing messages. There may be also glitches in altitude values[8]. Moreover, when aircraft is above the sea area data is lacking. As we experimentally verified errors may concern up to 0.9%, so data cleansing is a necessary step before any further processing of aircraft trajectories.

To evaluate the performance of 4D trajectory data cleansing techniques, a 4D trajectory dataset in European area is used as case studies. The dataset includes one-day 4D trajectories on 1 January, 2018. There are 5 905 137 records of 18 286 aircraft in total. We extract 4D trajectories of the aircraft 471F86 for the experimental purpose. The 4D trajectories of the flight DLH3EJ in the whole month of January, 2018 are also included to verify the performance of the clustering algorithm.

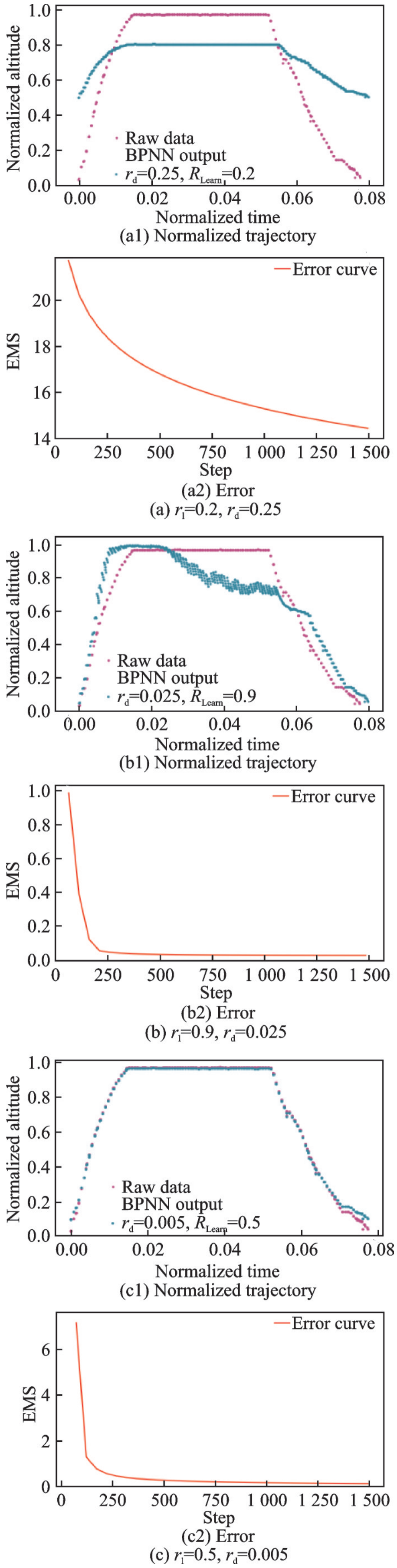
The experiments are performed on a laptop equipped with four-core i7-6300U 2.50 GHz, and 16 GB DRAM. The methods are all coded with Python 3.6.7.

2.2 Selection of parameters for BP neural network

In this section, a set of experiments are conducted to select proper parameters to build the BP neural network and the proposed training model. The trajectory data of aircraft 471F86 from Wroclaw (WRO) and Dortmund (DTM) are used. There are 575 records, and 16 errors are detected based on calculating the average speed between two adjacent records. BP neural network is applied with the data after cutting down the errors for model training.

Firstly, the performance of the BP network method is compared with different parameters of learning rate r_l and decay rate r_d. Fig. 3 shows the performance of the BP neural network with different values of learning rate and decay rate. Fig. 3(a) is the result with the learning rate set to 0.2 and the decay rate set to 0.25, which is under-trained. Fig. 3(b) is the results with the learning rate set to 0.9 and the decay rate set to 0.025, which is over-trained. Fig. 3(c) is the results with the learning rate set to 0.5 and the decay rate set to 0.005. In Fig. 3, if the learning rate is set too large and decay is small, the model would be over-trained. Otherwise, the learning rate is too small and decay is large, the final model would be under-trained and the training process would be slow. Therefore, we set the learning rate to 0.5 and the decay rate to 0.005 in the final training.

The number of nodes at each hidden layer influences the training quality and speed. Eight scenarios are tested with different combinations (n_{HL1}, n_{HL2}) of the number of nodes at each hidden layer, which are listed in Table 1, where n_{HL1} is the number of



(a1) Normalized trajectory

(a2) Error

(a) r_l=0.2, r_d=0.25

(b1) Normalized trajectory

(b2) Error

(b) r_l=0.9, r_d=0.025

(c1) Normalized trajectory

(c2) Error

(c) r_l=0.5, r_d=0.005

Fig.3 Performance of BP neural network with different learning rates and decay rates

nodes in the first hidden layer, and n_{HL2} the number of nodes in the second hidden layer. In each experiment, the learning rate is set to 0.5 and the decay rate to 0.005. Moreover, the training accuracy is set to 0.000 1 and the training process will stop if the results reach the training accuracy or the training steps are up to 3 000. The training time is also reported in Table 1. Fig. 4 shows the performance of the eight scenarios. In Table 1, setting n_{HL1} to 10 and n_{HL2} to 5 cost the shortest time to train the model. However, the performance of the combination (10, 5) shown in Fig.4(b) is not as good as that of the combination (5, 2) shown in Fig. 4(g). Therefore, in our final experiments, we set five nodes at the first hidden layer and two nodes at the second hidden layer.

Table 1 Performance of hidden layers 1, 2 with different combinations

n_{HL1}	n_{HL2}	Training time / s
20	5	7.429 3
10	5	0.550 6
5	5	1.630 2
1	5	1.158 1
10	10	2.531 3
5	10	1.810 5
5	2	1.399 9
5	1	1.447 4

2. 3 Results on real 4D trajectories

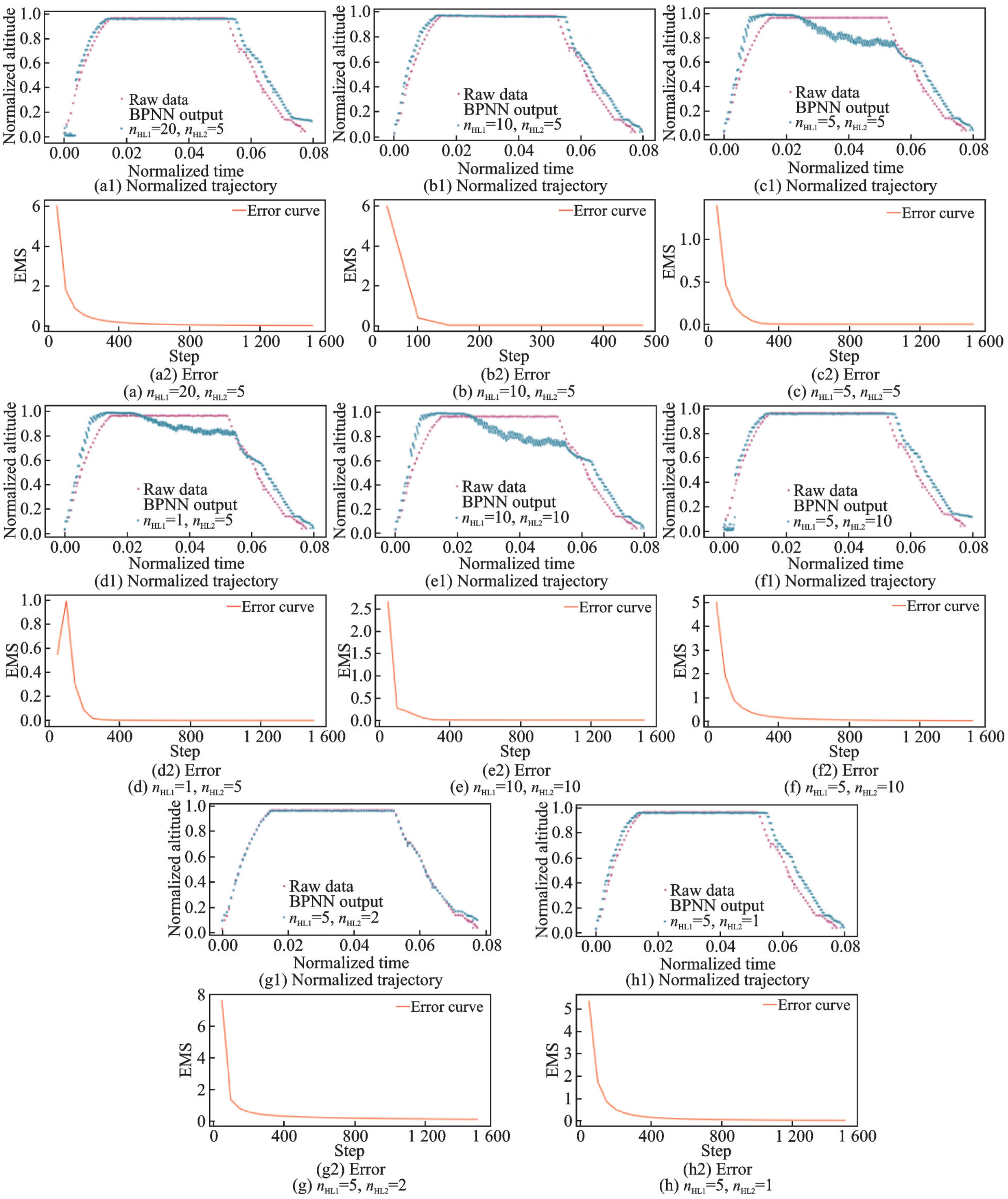
The proposed BP neural network model, Newton interpolation method and unit cube method are applied on real 4D trajectories. Fig.5 shows the results of the trajectories of the aircraft 471F86. The 471F86 aircraft finished eight flights between Wroclaw (WRO) and Dortmund (DTM), Wroclaw (WRO) and Eindhoven (EIN), Wroclaw(WRO) and Luton (LTN), Wroclaw and (WRO) Birmingham (BHX) on January 1st, 2018. There are 4 161 records in total.

Fig.5(a) shows the raw 4D trajectory data. We can see that some obvious wrong points, which drop or rise sharply as well as are far away from the major trajectory. Fig.5(b) shows the errors. The red points are the errors detected by calculating the average speed between two records. There are 50 red points which means that we detect 50 errors. Fig.5(c) shows the result after being repaired by BP neural network model. The input layer is time and the output layer is set as longitude, latitude, and altitude separately to repair errors. Fig.5(d) shows the results of interpolation data. The red points are the added points. There are 4 161 records before applying Newton interpolation method and 5 149 records after applying the method. In Fig.5(e), the green points are the final reduced points by the unit cube method. Only 900 points are selected, which shows that the unit cube method cuts down the data size vastly but keeps the trajectories. Fig.5(f) shows the trajectories after reducing records which is the finnal results of 4D trajectories of aircraft 471F86 after applying the set of data cleansing method. In Fig.5(f), the unit cube method can eliminate the glitches which cannot be detected by average speed. Comparing Fig.5(f) with the row trajectory data shown in Fig.5(a), the off-course data and glitchs are well processed, which shows the effectiveness on our data cleansing method.

2. 4 Results of comparison

Some data cleansing methods are illustrated and compared with Kalman filter. As the existing papers presented, most of data cleansing methods dealing with the trajectory data just drop the error when it is detected. This kind of process is the simplest but cannot assure the integrity of the original trajectory data. In addition, the Kalman filtering method is used in trajectory data cleansing. This kind of method can smooth glitches but some obvious glitches do still exist, which means that it is not effective enough. Although, the proposed method is relatively complex to some extent when comparing with the Kalman filtering method, the results show that our method can remove the errors utmostly, which demonstrates its effectiveness.

Here, the Kalman filter is implemented to compare with the proposed method. Fig.6 shows the results after applying Kalman filtering. In Fig.6, some errors are elimilated but some obvious glitchs still exist. Compared Fig.6 with Fig.5(f), the proposed method is more effective than the Kalman filter.



(a1) Normalized trajectory

(a2) Error

(a) n_{HL1}=20, n_{HL2}=5

(b1) Normalized trajectory

(b2) Error

(b) n_{HL1}=10, n_{HL2}=5

(c1) Normalized trajectory

(c2) Error

(c) n_{HL1}=5, n_{HL2}=5

(d1) Normalized trajectory

(d2) Error

(d) n_{HL1}=1, n_{HL2}=5

(e1) Normalized trajectory

(e2) Error

(e) n_{HL1}=10, n_{HL2}=10

(f1) Normalized trajectory

(f2) Error

(f) n_{HL1}=5, n_{HL2}=10

(g1) Normalized trajectory

(g2) Error

(g) n_{HL1}=5, n_{HL2}=2

(h1) Normalized trajectory

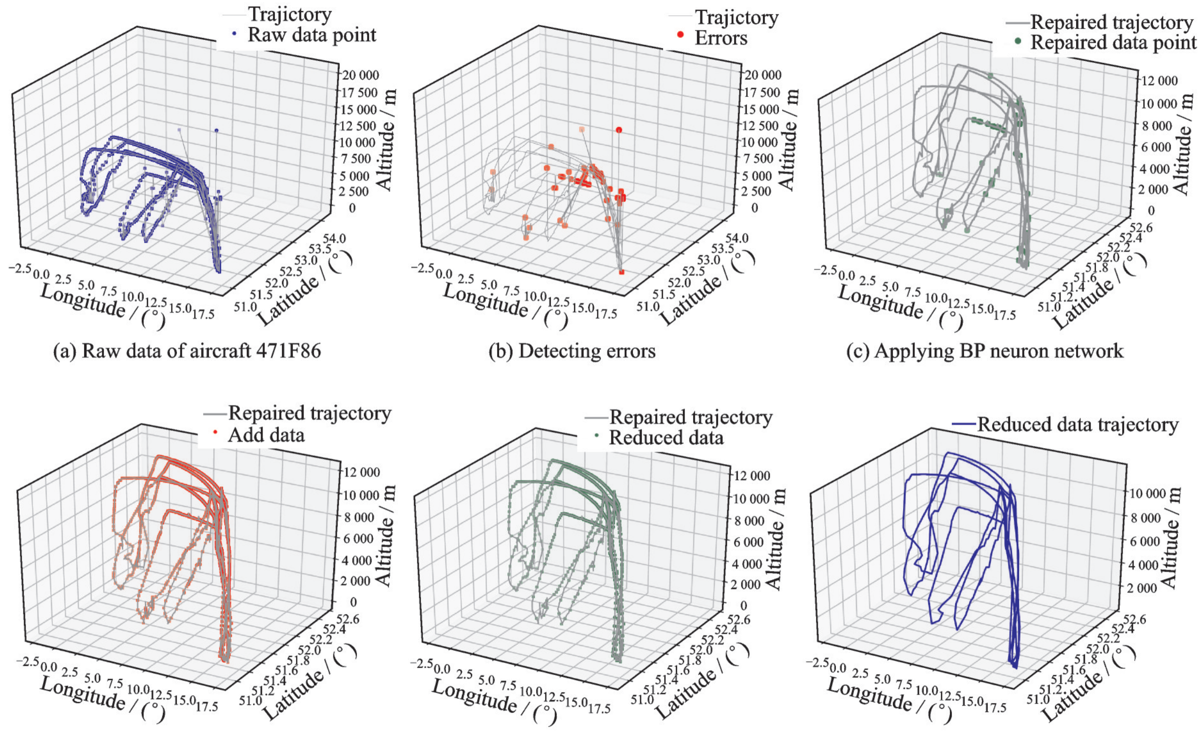
(h2) Error

(h) n_{HL1}=5, n_{HL2}=1

Fig.4 Performance of the number of nodes at hidden layers with different combinations

2.5 Clustering result analyses

The performance of DBSCAN is reported to identify the outliers of trajectories. Fig.7 shows the results of clustering the trajectories of OD pair London (LHR)-Brussels (BRU) and OD pair London (LHR)-Dusseldorf (DUS) in one day. Two additional flights of Zurich (ZRH)-Brussels (BRU) and Brussels (BRU)-Copenhagen (CPH) are also included in the test data. In Fig.7, except trajectories from LHR to BRU and LHR to DUS are clustered together, which are shown in blue._The trajectories of ZRH-BRU and BRU-CPH and noise points are clustered into other categories and displayed in different colors.



(a) Raw data of aircraft 471F86 (b) Detecting errors (c) Applying BP neuron network

(d) Applying Newton interpolation (e) Applying unit cube (f) Final trajectories

Fig.5 Data cleansing techniques on 4D trajectories of aircraft 471F86

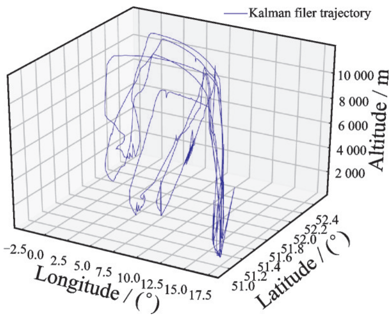


Fig.6 Kalman filter on 4D trajectories of aircraft 471F86

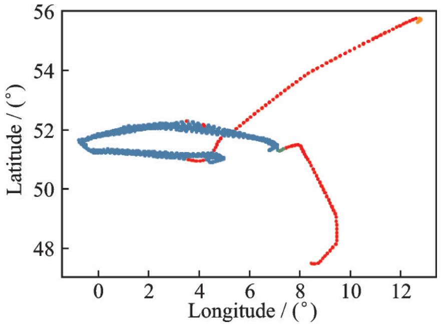


Fig.7 Clusters of trajectories of LHR- BRU and LHR-DUS

The DBSCAN algorithm is also applied to identify the trajectory noises of DLH3EJ flight, which flies from Oslo Gardermoen (OSL) to Frankfurt Int'l (FRA) in a month. Fig. 8 shows the results of DBSCAN based on flight DLH3EJ. The red points are identified as noises. In Fig.8, one can obtain the similar conclusions in Fig. 7. Therefore, the DBSCAN algorithm can be used to detect outliers of trajectories.

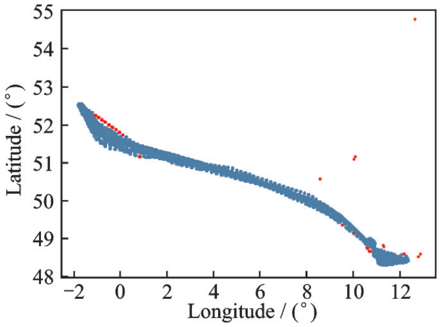


Fig.8 Clusters of trajectories of flight DLH3EJ

2. 6 Data cleansing methods on one-day trajectories

The data cleansing methods are applied to one-day 4D trajectories in the European area. There are 5 905 137 records of raw data in total. Since the data size is large, the three-dimension visualization is not a good option. The data are shown in two-dimensions with longitude and latitude.

Fig.9 shows the raw one-day 4D trajectory data. When aircraft fly above the area of sea, most of the data are lacked. If the records of one flight are less than 20, the flight is dropped out directly.

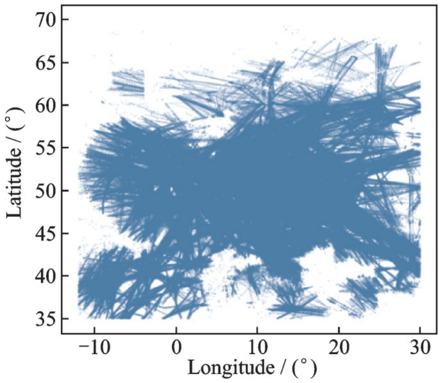
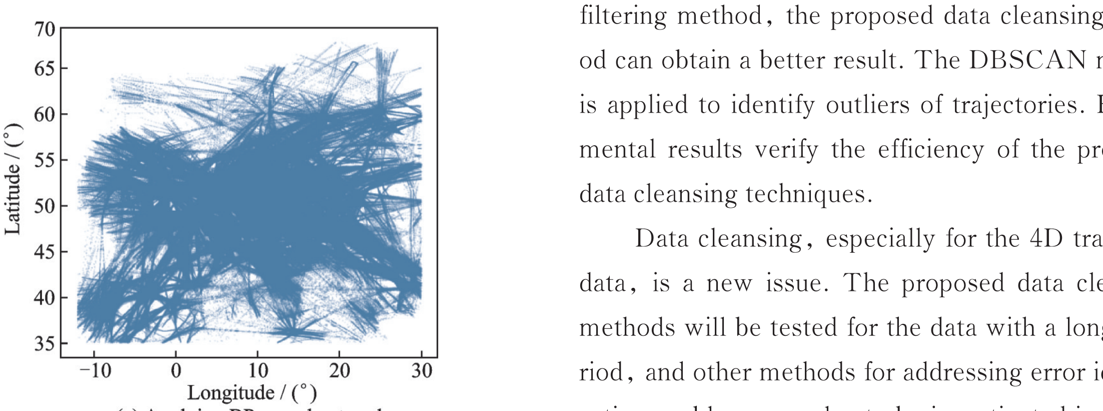


Fig.9 One-day records in Europe

Fig. 10 shows one-day trajectories after applying the data cleansing methods. Fig. 10 (a) shows the data after repairing the errors of each flight based on BP neural network. Fig. 10(b) shows the results of filling data based on Newton interpolation. Fig. 10(c) shows the results of reducing data based on unit cube. In Fig. 10(c), the off-course points are well trimmed and the points above the sea area are filled, which shows the effectiveness of the data cleansing method proposed in this paper.

There are 5 905 137 records of raw 4D trajectory data in total. After dropping out the duplicates, 5 012 518 points are left. There are 4 989 510 points after rounding the unreliable data detected by the average speed. In our experiment, 6 286 errors are detected and repaired based on the BP neural network method. There are 7 894 063 records existing after applying Newton interpolation and 1 699 110 left after cutting down data based on unit cube.



(a) Applying BP neural network

(b) Applying Newton interpolation

(c) Applying unit cube

Fig.10 Data cleansing techniques on one-day trajectories

3 Conclusions

A rich set of data cleansing techniques are presented for the 4D trajectory data. The errors are detected by the average speed, and the BP neural network is applied to deal with the errors. By using the Newton interpolation method, the frequencies of the points in the 4D trajectory data are fixed. To reduce the computational complexity while maintaining data characteristics, the unit cube sampling method is used to cut down the size of the 4D trajectory data significantly. Compared with the Kalman filtering method, the proposed data cleansing method can obtain a better result. The DBSCAN method is applied to identify outliers of trajectories. Experimental results verify the efficiency of the proposed data cleansing techniques.

Data cleansing, especially for the 4D trajectory data, is a new issue. The proposed data cleansing methods will be tested for the data with a longer period, and other methods for addressing error identification problems are also to be investigated in further research.

References

[1] RIBEIRO V F, DE ALMEIDA R H T, DE FARIA V B, et al. Conflict detection and resolution with local search algorithms for 4D-navigation in ATM[C]// Proceedings of International Conference on Intelligent Systems Design and Applications. [S. l.] : Springer, 2018: 129-139.

[2] JACKSON M R C, GONDA J, MEAD R, et al. The 4D trajectory data link (4DTRAD) service-Closing the loop for air traffic control[C]//Proceedings of 2009 Integrated Communications, Navigation and Surveillance Conference. [S.l.]: IEEE, 2009: 1-10.

[3] ROSENOW J, FRICKE H, SCHULTZ M. Air traffic simulation with 4D multi-criteria optimized trajectories[C]//Proceedings of 2017 Winter Simulation Conference (WSC). [S.l.]: IEEE, 2017: 2589-2600.

[4] MARÍNEZ-PRIETO M A, BREGON A, GARCÍA-MIRANDA I, et al. Integrating flight-related information into a (big) data lake[C]//Proceedings of 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC). [S.l.]: IEEE, 2017: 1-10.

[5] ANDRISANI D, AYOUBI M, HOSHIZAKI T. Aircraft ADS-B data integrity check[C]//Proceedings of AIAA 4th Aviation, Technology, and Operations Conf. [S.l.]: AIAA, 2004.

[6] DESELL T, CLACHAR S, HIGGINS J, et al. Evolving neural network weights for time-series prediction of general aviation flight data[C]//Proceedings of International Conference on Parallel Problem Solving. [S.l.]: Springer Nature, 2014: 771-781.

[7] ALI B S, SCHUSTER W, OCHIENG W, et al. Analysis of anomalies in ADS-B and its GPS data[J]. GPS Solutions, 2016, 20(3): 429-438.

[8] PATROUMPAS K, PELEKIS N, THEODORIDIS Y. On-the-fly mobility event detection over aircraft trajectories[C]//Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. [S. l.] : ACM, 2018: 259-268.

[9] BOHANNON P, FAN W, GEERTS F, et al. Conditional functional dependencies for data cleaning[C]// Proceedings of 2007 IEEE 23rd international conference on data engineering. [S.l.]: IEEE, 2007: 746-755.

[10] KOUDAS N, SAHA A, SRIVASTAVA D, et al. Metric functional dependencies[C]//Proceedings of 2009 IEEE 25th International Conference on Data Engineering. [S.l.]: IEEE, 2009: 1275-1278.

[11] GRZYMALA-BUSSE J W, GOODWIN L K, GRZYMALA-BUSS W J, et al. Handling missing attribute values in preterm birth data sets[C]//Proceedings of International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing. Berlin, Heidelberg: Springer, 2005: 342-351.

[12] BATISTA G, MONARD M. An analysis of four missing data treatment methods for supervised learning[J]. Applied Artificial Intelligence, 2003, 17(5/6): 519-533.

[13] SHAN Y, DENG G. Kernel PCA regression for missing data estimation in DNA microarray analysis[C]// Proceedings of IEEE International Symposium on Circuits and Systems. [S.l.]: IEEE, 2009: 1477-1480.

[14] GARCIA I, MARTINEZ-PRIETO M A, BREGÓN A, et al. Towards a scalable architecture for flight data management[C]//Proceedings of the 6th International Conference on Data Science, Technology and Applications. [S.l.]: Scite P Ress, 2017.

[15] OLIVE X, BASORA L. A python toolbox for processing air traffic data: A use case with trajectory clustering[C]//Proceedings of the 7th OpenSky Workshop. [S.l.]: Easy Chair, 2019, 67: 73-84.

[16] NI X, WANG H, CHE C. Risk index prediction of civil aviation based on deep neural network[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2019, 36(2): 313-319.

[17] LI J, YAO X, WANG X, et al. Multiscale local features learning based on BP neural network for rolling bearing intelligent fault diagnosis[J]. Measurement, 2020, 153: 107419.

[18] LI Y, LI J, HUANG J, et al. Fitting analysis and research of measured data of SAW micro-pressure sensor based on BP neural network[J]. Measurement, 2020, 155: 107533.

[19] SHI Y, LI Y, CAI M, et al. A lung sound category recognition method based on wavelet decomposition and BP neural network[J]. International Journal of Biological Sciences, 2019, 15(1): 195.

[20] HUANG X, JIN H, ZHANG Y. Risk assessment of earthquake network public opinion based on global search BP neural network[J]. PloSOne, 2019, 14 (3): e0212839.

[21] LIN S, WANG G, CHEN Y, et al. Warehouse environment parameter monitoring system and sensor error correction model based on PSO-BP[J]. Transactions of Nanjing University of Aeronautics and Astronau-

tics, 2017 (3): 333-340.

[22] ZHAO H, ZENG X, HE Z. Low-complexity nonlinear adaptive filter based on a pipelined linear recurrent neural network[J]. IEEE Transactions on Neural Networks, 2011, 22(9): 1494-1507.

[23] WANDELT S, SUN X. Efficient compression of 4D-trajectory data in air traffic management[J]. IEEE Transactions on Intelligent Transportation Systems, 2014, 16(2): 844-853.

[24] WANDELT S, SUN X, HARTMUT F. ADS-BI: Compressed indexing of ADS-B data[J]. IEEE Transactions on Intelligent Transportation Systems, 2018, 19(12): 3795-3806.

[25] WANG Q, GUAN Y, WANG A, et al. Comparison of GPS satellite orbit three-dimension coordinate interpolation algorithms[J]. Progress in Geophysics, 2014, 29(2): 573-579. (in Chinese)

[26] WANG L, PENG B. Track clustering based on LOFC time window segmentation algorithm[J]. Journal of Nanjing University of Aeronautics & Astronautics, 2018, 505: 661-665. (in Chinese)

Acknowledgements This work was supported by the National Natural Science Foundations of China (Nos. 61861136005, 61851110763, and 71731001).

Authors Ms. **TAN Xin** is a Master student at Beihang University since 2018. Her major interests are big data and machine learning.

Prof. **WANDELT Sebastian** works as a Professor at Beihang University. He received a Ph.D. degree in computer science from Hamburg University of Technology in Germany in 2011. His research interests are intelligent transportation systems and scalable data management.

Author contributions Prof. **SUN Xiaoqian** and Prof. **WANDELT Sebastian** conceptualized the study, contributed data and discussion of the study. Ms. **TAN Xin** and Ms. **ZHANG Chunxiao** designed the study, implemented the experiments, interpreted the results and wrote the manuscript. All authors commented on the manuscript draft and approved the submission.

Competing interests The authors declare no competing interests.

(Production Editor: ZHANG Tong)

高效四维航迹数据清洗技术

谭 鑫[1,2]，孙小倩[1,2]，张春晓[1,2]，Wandelt Sebastian[1,2]

(1.北京航空航天大学国家空管新航行系统技术重点实验室,北京 100191; 2.北京航空航天大学电子信息工程学院,北京 100191)

摘要:随着航空业的快速发展和新兴的众包,如Flightradar24和FlightAware等的涌现,大量空中交通数据,特别是四维(4D)航迹数据已公开于众。为了保证结果的准确性和可靠性,包括识别和减少错误在内的数据清洗,是分析4D航迹数据的第一步。为此对4D航迹数据进行如下清洗:应用反向传播神经网络算法将误差修复;用牛顿插值法对每次航行样本进行等间隔采样来获得均匀分布的4D航迹数据;进而提出一种在保持轨迹固有形状前提下的数据压缩方法,以及基于密度的有噪聚类(Density-based spatial clustering of applications with noise, DBSCAN)对样本点中的离群点进行识别。采用欧洲空域一天的4D航迹数据集进行验证,结果表明所提方法比现有方法更高效、快速。本文的数据预处理研究结果为下一阶段的4D航迹分析奠定了基础。

关键词:四维航迹;数据清洗;异常值检测;修复