Chapter 2
Preprocessing of Flight Data

Research findings indicate that the amount, accuracy, and type of data collected from airborne FDRS can hardly meet the requirements for further application and development of flight data. Therefore, it is necessary and fundamental to preprocess original flight data. The level of accuracy and reliability of preprocessing results will have a direct bearing on the quality of follow-on research. This chapter covers outlier elimination, data filling, data extension, reduction of monitorable parameters and chaotic property analysis.

2.1 Support Degree-Based Amnesic Fusion Filtering Method for Flight Data

Diverse and redundant data collected and recorded from FDRS are information sources for application of data fusion filtering technologies. In this section, amnesic fusion filtering method of flight data based on support degree is introduced, which greatly supplements and expands current outlier preprocessing algorithms. Exponential functions are used to build a matrix of support degree to conduct weighting fusion of measurement data, thus eliminating the dependence of the algorithm upon priori knowledge. Meanwhile, an amnesic control item is added to dynamically adjust the fusion weights according to data, thus preventing the "data saturation" caused by excessive data.

2.1.1 Unified Error Model of Flight Data

Every measurement result has errors. Errors exist in every scientific experiment and measurement. Webster defines error as "the difference between an observed value

J. Zhang and P. Zhang, *Time Series Analysis Methods and Applications for Flight Data*, DOI 10.1007/978-3-662-53430-4_2

and the true value of a quantity." There is no exception for FDRS. As a typical real-time data measuring and recording system, FDRS is subject to a variety of interferences during its operation. Thus, outliers are inevitable in the flight data. These abnormal data will have a direct influence upon follow-on processing. Only through sorting out the quality of these flight data can be guaranteed.

Measurement errors are determined by such factors as measuring environment, times and moment. Measurement errors fall into three categories: system errors, random errors, and gross errors. System errors can be reduced by improving the performance of equipment while random errors and gross errors can be reduced with application of appropriate mathematic methods.

As for raw flight data, the relationship between the true value and the measurement value is as follows:

$$X'(k) = X(k) + \Delta e0 + \Delta e1 + \Delta e2 + \Delta e3 + \Delta e4 \tag{2.1}$$

In the formula, $X'(k)$ stands for the measurement value of parameter X at sampling time k; $X(k)$ for the true value of parameter X at sampling time k; $\Delta e0$ for random interference error; $\Delta e1$ for sensing error of the sensor; $\Delta e2$ for sensing and quantifying error of the collector; $\Delta e3$ for recording error of the recorder; $\Delta e4$ for data deciphering error from the ground station (only valid for continuous parameters).

It can be seen that errors of FDRS are caused by various factors. Therefore, it is extremely difficult and impractical to apply traditional error compensation preprocessing methods to accurately estimate all error sources. It is especially more difficult for random interference error because of irreproducibility of live air flight.

Although errors of different types have different sources, they will be reflected in the final values. That is to say, they will present themselves in the form of deviations of actually measurement values—measurement noise. The fundamental purpose of flight data filtering preprocessing is to acquire more accurate measurement values. What is really of concern is the reflection of errors in the measurement values, not the concrete values of a certain error. To correct measurement values by estimating various errors is only an indirect method adopted to improve accuracy.

Based on the above-mentioned assumption, a unified error model of multisource flight data can be established as follows:

$$X'(k) = X(k) + V(k) \quad k = 1, 2, \ldots, n \tag{2.2}$$

In the formula, $X'(k)$ stands for the measurement value of parameter X at sampling time k; $X(k)$ for the true value of parameter X at sampling time k; $V(k)$ for the sum of error sources $\Delta e0$, $\Delta e1$, $\Delta e2$, $\Delta e3$, and $\Delta e4$ at sampling time k. That is to say, $V(k)$ is the general description of all error sources with unknown priori knowledge such as $E[V(k)]$ and $D[V(k)]$.

The mathematic description model for unified errors of flight data is a simple and practical method oriented to industrial application. It changes the traditional way to

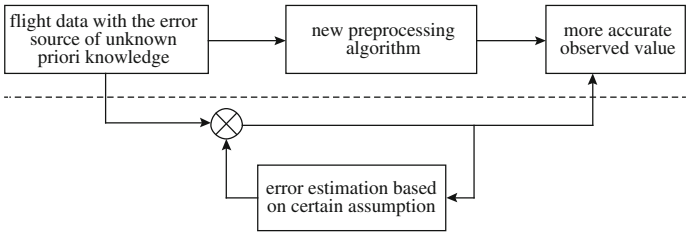


Fig. 2.1 Process contrast of two error compensation methods

analyze error compensation, eliminates many unnecessary assumptions, and avoids difficulty in estimating unknown priori errors.

Figure 2.1 shows the contrast of the different processing processes of the two error compensation methods. As shown in the figure, the block diagram below the dotted line reflects the preprocessing thinking of the traditional error compensation method. As in engineering practice, there is not such a thing as an ideal measured object, it is always the case that while using the traditional method, assumptive conditions cannot be met to guarantee the algorithm effective. The block diagram above the dotted line reflects the preprocessing thinking of the unified error compensation method. With unknown priori knowledge about the error source, the key to use the unified error compensation model to acquire more accurate measurement value lies in using filtering preprocessing algorithm independent of any priori knowledge.

2.1.2 Support Degree-Based Amnesic Fusion Filtering Algorithm

1. Problem Description

Current filtering algorithms for flight data preprocessing focus on a specific parameter, and thus have the following deficiencies:

(1) In the process of parameter filtering, judgment or weighting is fixed (as for an artificial neural network, the connecting weight is also fixed when training finishes). Thus its universality is poor;
(2) Once the parameter changes, it is complex, time-consuming, and inefficient to rebuild a model;
(3) Although the weight moving average filtering method takes into consideration the application of a larger filtering weight for information, lack of specific materialization foundation leads to insufficient utilization of such features as diversity and redundancy of FDRS-collected data, thus unable to validate the filtering data.

Compared with using one sensor, more reliable results can be acquired by utilizing multiple sensors for status parameters and using applicable fusing algorithm for them. And in this way, the measured object can also be more accurately identified. In recent years, parameter estimation based on multisensor data fusing technologies has aroused wide research interest. Diversity and redundancy of FDRS-collected and recorded data guarantee the information source for utilizing data fusing technologies. There are various recording media in modern FDRS such as primary recorder, secondary recorder, accident recorder, and QAR. Different recording media can provide multiple measured and recorded values for parameters with certain feature at the same time. Moreover, different information sources of the same recording media can also provide multiple measured and recorded values for parameters with certain feature at the same time, including pressure altitude and radio altitude, absolute altitude and vertical speed, heading provided by the attitude heading system and the inertial navigation system, etc.

At present, the common fusing algorithms can be classified into two types. The first type includes fusing methods based on priori knowledge(noise intensity, priori probability distribution, associated probability distribution, etc.), such as Bayes method based on statistics theory, optimal weight allocation methods based on planning, fusion methods based on Kalman Filters and its extended version, etc. As priori knowledge of measurement values is used for estimation, fusion works well. However, in engineering practices, these algorithms often fail because of inability to meet assumptive conditions. The second type includes fusion algorithms independent of priori knowledge, such as fusion methods based on relationship matrix, fusion methods based on relative distance and confidence distance, fusion methods based on closest measured distance, and consistency fusion based on support degree. These algorithms are oriented to engineering practices, applicable for occasions when priori knowledge of measurement values is unknown. However, with increasing measurement times, the increasing quantity of data greatly impacts the execution efficiency of these algorithms; meanwhile, the information provided by new data is submerged in the old data "ocean", and thus recursive algorithms lose corrective ability and cannot be updated when deviation from the true value occurs.

2. Measurement of Support Degree

For a sensor array consisting of n sensors, direct measurement is adopted to check static or gradually changing parameter X, i.e., $z_i(k) = X + v_i(k)$, $k = 1, 2, \ldots, n$. In this formula, $z_i(k)$ stands for the measurement value of sensor i at time k; X for the true value; $v_i(k)$ for measured noise at time k, with unknown priori knowledge of $E[V(k)]$ and $D[V(k)]$. Great difference between $z_i(k)$ and $z_j(k)$ means low mutual support degree of the measurement values of two sensors; otherwise means high. Exponential attenuation function is used to form a support degree matrix to express support degrees of these sensors at the same measurement time.

Definition 2.1 Support degree $\alpha_{ij}(k)$ of the measurement values of sensors i and j at time k is given by

$$a_{ij}(k) = \exp\left\{-a\left[z_i(k) - z_j(k)\right]^2\right\} \tag{2.3}$$

In the formula, a is an adjustable parameter. Using exponential attenuation function to express support degrees of measurement values of these sensors, the absolute values of 0 or 1 for support degrees in traditional methods are avoided. Therefore, the support degree of any two sensors at time k can be expressed by the support degree matrix SD(k):

$$\mathrm{SD}(k) = \begin{bmatrix} 1 & a_{12}(k) & \cdots & a_{1n}(k) \\ a_{21}(k) & 1 & \cdots & a_{2n}(k) \\ \vdots & \vdots & & \vdots \\ a_{n1}(k) & a_{n2}(k) & \cdots & 1 \end{bmatrix} \tag{2.4}$$

Obviously, for element i in support degree matrix $SD(k)$, if $\sum_{j=1}^{n} a_{ij}(k)$ has a relative large value, the measurement values of sensor i are more consistent with those of other sensors; otherwise, they deviate from those of other sensors.

3. Amnesic Factor

Definition 2.2 The function that determines the weight for new information and older information in the multistreaming time series, whose value varies with time, is called an amnesic function $A(t)$.

It is obvious that an amnesic function has monotonic property. That is, if $M \leq N, M, N \in R$, then $A(M) \leq A(N)$. This property is based on the fact that we would always believe that new information is more useful than older information, so the weight for the value of last measured data should be larger than that for the old one.

It should be noted that any monotonic nondecreasing function can be called amnesic function $A(t)$. However, the choice of $A(t)$ depends on the engineering issue itself. Here are some common amnesic functions.

(1) **Linear amnesic function**:

$$A(t) = at + b, a, b > 0$$

(2) **Exponential amnesic function**:

$$A(t) = e^{-at}, a > 1$$

(3) **Piecewise constant amnesic function**:

$$A(t) = \begin{cases} C_1, & 0 \leq t < M \\ C_2, & M \leq t \leq N \end{cases}$$

In the formula, $C_1 > 0, C_2 > 0, C_1 \leq C_2$ are constants and cannot be 1 at the same time; N represents current time, M is a parameter for amnesic depth. Obviously, M continuously increases with the ongoing measurements, but $N - M$ will remain constant. Piecewise constant amnesic function is simple in form and easy for engineering application. For the convenience of calculation, piecewise constant amnesic function is adopted for the applications in this section, with $C_1 = 0, C_2 = 1$.

4. Algorithm Descriptions

Definition 2.3 Consistency parameter of measurement values of sensor i with those of the other sensors at time k is given by

$$r_i(k) = \frac{\sum_{j=1}^{n} a_{ij}(k)}{n} \tag{2.5}$$

Obviously, $0 < r_i(k) \leq 1$.

Definition 2.1 reflects the approximation level of the measurement values of the two sensors at time k. Definition 2.3 reflects the approximation degree of the measurement values of sensor i with those of the other sensors (sensor i included) at time k.

Although $r_i(k)$ is large at one time, it does not mean that sensor i is reliable at any time of the whole observation range. The reliability of the sensor can be expressed by the consistency metrics at all measurement time. For example, $r_i(k)$ is very large at one time, and becomes very small at another, which indicates that consistency of sensor measurements is unstable. That is to say, the reliability of the sensor is poor during the whole measuring process. The intuition behind this is that the sensor with large mean value and steady variance of $r_i(k)$ time series should be assigned with bigger weight. Therefore, the two statistical concepts of mean value and variance value are used to study the reliability information contained in $r_i(k)$ series at different times.

Mean value of $r_i(k)$ series of sensor i at time k is

$$\overline{r_i(k)} = \frac{1}{k}\sum_{t=1}^{k} r_i(t) \tag{2.6}$$

Variance value of $r_i(k)$ series at time k is

$$\sigma_i^2(k) = \frac{1}{k}\sum_{t=1}^{k} [\overline{r_i(k)} - r_i(t)]^2 \tag{2.7}$$

Definition 2.4 $q_i(k)$ denotes the weight of the measurement value of sensor i at time k.

In the actual fusion process, measurement information of sensors with large mean value and small variance value (i.e., with high consistency and reliability) should be fully utilized. Therefore, weight $q_i(k)$ of the measurement value of sensor i should be in plus correlation with $\overline{r_i(k)}$ and minus correlation with $\sigma_i^2(k)$. To avoid $q_i(k)$ from being minus, linear function may be used to measure the final weight, i.e., $q_i(k) = [1 - \lambda\sigma_i^2(k)]\overline{r_i(k)}$, where λ is an adjustable parameter. By adjusting λ, the influence of $\sigma_i^2(k)$ on $q_i(k)$ will be changed accordingly.

So, the fusion estimation based on support degree can be expressed in this way:

$$\hat{\boldsymbol{X}}(k) = \frac{\sum_{i=1}^{n} q_i(k) z_i(k)}{\sum_{i=1}^{n} q_i(k)} \tag{2.8}$$

This formula is a consistent fusion algorithm. Amnesic function control item is further introduced and amnesic fusion algorithm based on support degree is denoted as the following formula:

$$\hat{\boldsymbol{X}}(k) = \frac{\sum_{i=1}^{n} w_i(k) z_i(k)}{\sum_{i=1}^{n} w_i(k)} \tag{2.9}$$

In this formula, $w_i(k) = [1 - \lambda\sigma_i^2(k)]\overline{r_i(k)}$ stands for amnesic fusion weight and can be calculated with the following formulas:

$$\overline{r_i(k)} = \begin{cases} \frac{\sum_{l=1}^{k} r_i(l)}{k} (k \le m) \\ \frac{\sum_{l=k-m+1}^{k} r_i(l)}{m} (k > m) \end{cases} \tag{2.10}$$

$$\sigma_i^2(k) = \begin{cases} \frac{1}{k} \sum_{l=1}^{k} [\overline{r_i(k)} - r_i(l)]^2 (k \le m) \\ \frac{1}{m} \sum_{l=k-m+1}^{k} [\overline{r_i(k)} - r_i(l)]^2 (k > m) \end{cases} \tag{2.11}$$

$$m = N - M$$

2.1.3 Application and Conclusion

Some flight data are chosen as samples. Sixteen groups of data about measurement records of "engine exhaust gas temperature" parameters are extracted from the FDMS primary recorder, QAR, and accident recorder. As shown in Table 2.1, the recorders are noted as "Sensor A, Sensor B, Sensor C," respectively, in accordance with the descriptions in data fusion field.

Table 2.1 Sensor measurement values

Measurement time	Measurement values (target value 900, unit: °C)		
	Sensor A	Sensor B	Sensor C
1	908.7	201.1	903.6
2	909.1	352.2	905.3
3	890.5	401.5	905.0
4	889.7	903.1	900.8
5	892.1	900.8	896.3
6	892.5	898.9	897.0
7	908.5	898.3	896.8
8	910.0	901.5	905.2
9	907.9	902.1	904.9
10	889.2	903.1	904.3
11	910.0	903.4	905.0
12	910.0	899.0	901.2
13	908.3	901.5	895.9
14	889.3	898.6	896.5
15	908.5	902.1	896.7
16	889.2	901.7	903.2

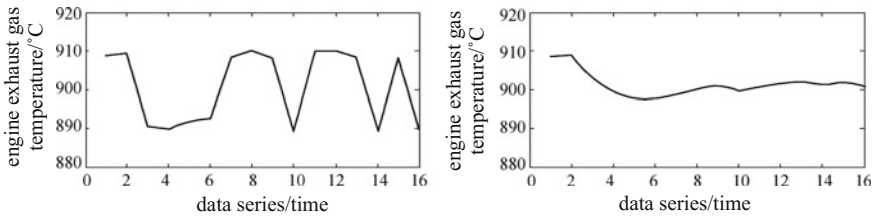


Fig. 2.2 Different effects of the original series and the filter series of Sensor A

Mean value fusion algorithm $X_N = \frac{1}{N}\sum_{i=1}^{N} x_i$ is used to preprocess the three sets of measurement data in Table 2.1, with $n = 16$. The different effects of the original series and the filter series of Sensor A and Sensor C are shown in Figs. 2.2 and 2.3, respectively.

As shown in Figs. 2.2 and 2.3, mean value fusion algorithm can be used to filter and smooth the data. The different effects of the original series and the filter series of Sensor B are shown in Fig. 2.4.

As shown in Fig. 2.4, the function of the arithmetic mean filtering method in preprocessing data is gradually degrading. There is an obvious deviation from the target value in the fourth filtering because of the following major reasons: during the iterative process, outliers are not eliminated in time and redundant information provided by multiple sensors is not efficiently utilized.

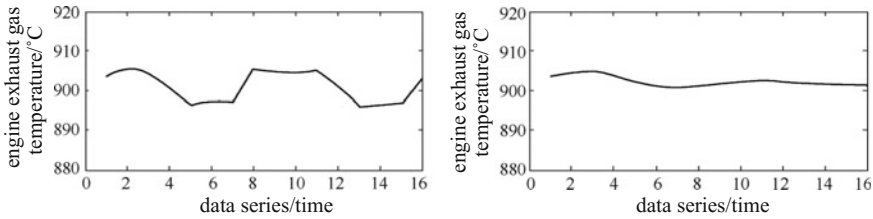


Fig. 2.3 Different effects of the original series and the filter series of Sensor C

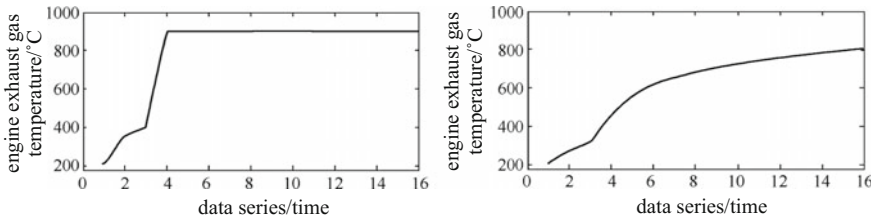
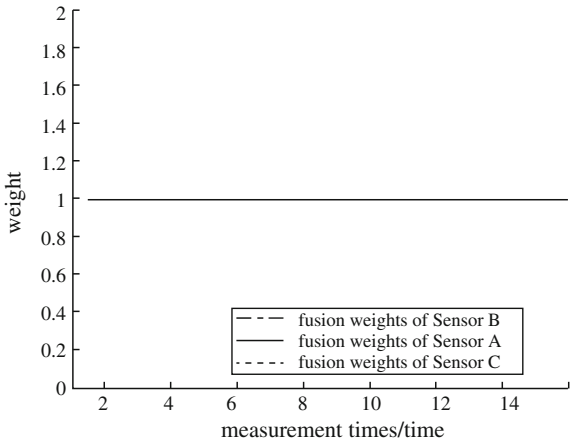


Fig. 2.4 Different effects of the original series and the filter series of Sensor B

Fig. 2.5 Changes of the weights of three sensors for each measurement time based on mean value fusion



In the following section, mean value fusion algorithm, amnesic fusion algorithm based on support degree ("memory fusion" for short), and consistency fusion algorithm are used to filter and estimate the measurement values in Table 2.1.

For memory fusion and consistency fusion algorithms, the same weights used include measurement weight $\alpha = 0.85$, weight $\lambda = 0.35$, and memory weight $m = 5$. The changes of the weights of the three sensors for each measurement time are shown in Figs. 2.5, 2.6, and 2.7.

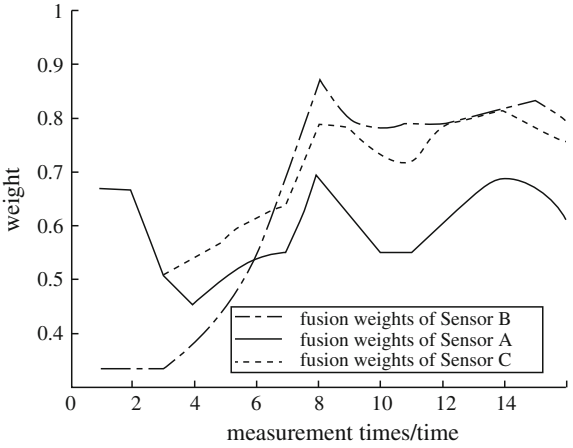


Fig. 2.6 Changes of the weights of three sensors for each measurement time based on memory fusion

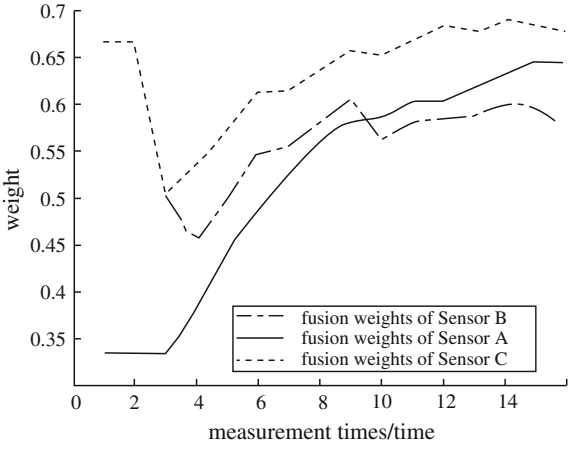


Fig. 2.7 Changes of the weights of three sensors for each measurement time based on consistency fusion

Obviously, for mean value fusion, a same weight (1 for all of the three sensors) is adopted to fuse measurement values; for consistency fusion, reliability information of the sensors provided by support degree matrix can be effectively used to allocate weights. But because of the influence of interferences, the weight of Sensor B cannot be effectively reallocated, still smaller than the weight of Sensor C in the end of measuring process; for limited memory fusion, based on consistency fusion, limited memory control item is also introduced. When the measurement value is updated for the eighth time, influence of interferences upon weight allocation is eliminated in time, and thus the weights of the three sensors are effectively and properly allocated again.

The fusion filtering effects generated by the three methods are shown in Figs. 2.8 and 2.9, where vertical coordinates refer to absolute error (°C), and horizontal coordinates refer to measurement times(time).

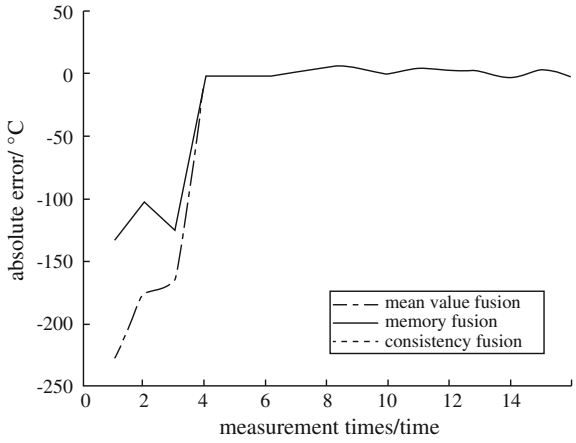


Fig. 2.8 Comparison of fusion filtering effects

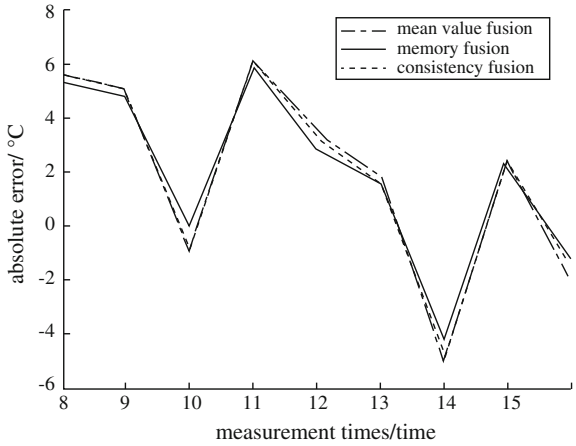


Fig. 2.9 Comparison of partially magnified fusion effects of measurement times 8–16

Figure 2.9 illustrates that as the measurement proceeds, amnesic control item plays an important role in adjusting the proper allocation of weights, thus further improving fusion accuracy. Judging from the changing process of absolute errors, with the application of amnesic fusion algorithm based on support degree, the filtering absolute errors are fewer than those in the other two methods as a whole, and they are steady and convergent.

2.2 Missing Flight Data Filling Method Based on Comprehensive Weighting Optimization

Because the operation of FDRS may be interfered by various factors, there may be missing values for recorded flight data, i.e., "missing records". Just like the processing of noise data, before further developing the flight data, effective data filling must be finished to ensure effective application of flight data. By combining comprehensive neural network and least square fitting, a method to fill the missing flight data under α condition is introduced.

2.2.1 Modified Neural Network Model Based on Mixed Algorithm

It is known that there are two problems in executing the algorithm of BP neural network: many partial minimal points and platform problem; and step size selecting problem. To solve these problems, the following methods are often adopted: mended BP (MBP), random optimization (RO), etc. Based on steepest descent optimization algorithm, MBP adopts directional searching method to choose the step size. Thus, low-quality partial minimal points may be converged. RO is not constrained by steepest descent, thus the searching direction can change randomly.

Modified neural network based on mixed algorithm adopts two-layer single-output network. In the network, measurement data is used as inputs, Sigmoid function and linear function are chosen, respectively, from the neuron at the hidden layer and the output layer, and corresponding fitting results are outputs. The learning algorithm of neural network combines the process of MBP and RO which are conducted by turns. When MBP result is judged to be a low-quality minimal point, RO is used; when RO result is judged to move close to a high-quality minimal point, MBP comes back.

1. Judgment of MBP Algorithm Running into a Partial Minimal Point

In the MBP algorithm, supposing $|\Delta R_{\text{emp}}(\xi(k))| = \delta(k)$, it gradually decreases with the increase of k. Supposing ε is a small plus, when k increases to a certain threshold, if $\left[\delta(k')/R_{\text{emp}}(\xi(k'))\right] < \varepsilon$, and $R_{\text{emp}}(\xi(k'))$ is still large, judgment of the algorithm running into a low-quality minimal point can be made.

2. Judgment of RO Moving from a Partial Minimal Point Close to a High-Quality Partial Minimal Point

In the RO algorithm, $\xi(k')$ is taken as initial value, and iterative computing is started from $k = 0$. Supposing $\delta(k)$ is the decreasing amount of experience risk for each iterative step, when k increases to a certain threshold, if the inequality $\sum_{k=0}^{k''} \delta(k) > \lambda R_{\text{emp}}(\xi(k'))$ exists, the judgment of moving is true. The left part of

the inequality is the cumulative risk decreasing amount in the RO algorithm, and coefficient λ is a proper plus, $\lambda \in [0.1, 0.2]$.

2.2.2 Polynomial Fitting Model Based on Least Square Method

Supposing measurement data is $y(t)$, an n-order polynomial of a time variant is used during a certain period of time to describe:

$$y(t) = a_1 t^n + a_2 t^{n-1} + \cdots + a_n t + a_{n+1} \tag{2.12}$$

Supposing the measurement data for $N+1$ equal time intervals are $y_k, y_{k+1}, \ldots, y_{k+N}$ in turn, T refers to the interval, and $t_k = 0$ refers to the time of the initial measurement point, thus $t_{k+lT} = lT$. The following formula can be inferred:

$$y_{k+l} = a_1 (lT)^n + a_2 (lT)^{n-1} + \cdots + a_n (lT) + a_{n+1} \tag{2.13}$$

Suppose

$$a'_1 = a_1 T^n,\ a'_2 = a_2 T^{n-1},\ a'_3 = a_3 T^{n-2}, \ldots,\ a'_{n+1} = a_{n+1}$$

Then,

$$\begin{bmatrix} y_k \\ y_{k+1} \\ \cdot \\ \cdot \\ y_{k+N} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdot & \cdot & 0 & 1 \\ 1 & 1 & \cdot & \cdot & 1 & 1 \\ \cdot & \cdot & & & \cdot & \cdot \\ \cdot & \cdot & & & \cdot & \cdot \\ N^n & N^{n-1} & \cdot & \cdot & N & 1 \end{bmatrix} \begin{bmatrix} a'_1 \\ a'_2 \\ \cdot \\ \cdot \\ a'_{n+1} \end{bmatrix} \tag{2.14}$$

Suppose

$$V = \begin{bmatrix} 0 & 0 & \cdot & \cdot & 0 & 1 \\ 1 & 1 & \cdot & \cdot & 1 & 1 \\ \cdot & \cdot & & & \cdot & \cdot \\ \cdot & \cdot & & & \cdot & \cdot \\ N^n & N^{n-1} & & & N & 1 \end{bmatrix}$$

$$Y = [y_k, y_{k+1}, \ldots, y_{k+N}]^T$$

$$X = \left[a'_1, \ldots, a'_{n+1}\right]$$

Substitute them into the formula (2.14) to get

$$Y = VX \tag{2.15}$$

Obviously, this is a contradictory equation group and it can be solved with least square method:

$$\hat{X} = \left[V^T V\right]^{-1}\left[V^T Y\right] \tag{2.16}$$

then,

$$\hat{Y} = V[V^T V]^{-1}[V^T Y]$$

Thus, it is easy to get the algorithm to calculate $\hat{y}_{k+l}(l = 0, \ldots, N)$, the estimated value of the n-order polynomial with $N+1$ points at time l:

$$\hat{y}_{k+l} = [\,1 \quad l \quad l^2 \quad \cdot \quad \cdot \quad \cdot \quad l^n\,]\{[V^T V]^{-1}[V^T Y]\} \tag{2.17}$$

When $l > N$, the formula (2.17) is converted to use current measurement data to predict the estimated data at an unmeasured point. When $l = N+1, N+2, N+3$, the values are status values for the follow-on first, second, and third time points. With the increasing distance between the predicted point and current measured data segment, prediction error will gradually increase. Thus, generally speaking, only the follow-on three points are predicted.

When $l = 0, 1, \cdots, N$, the formula (2.17) can be used to predict the estimated value for each measurement point. If the polynomial fitting algorithm is used for the whole data extent, the order of the polynomial would be too high, leading to instability of the solution of formula (2.16). Therefore, the polynomial moving fitting algorithm is adopted, where a small data segment is selected as a window with $(2M+1)$ odd points.

Low-order polynomial fitting algorithm, a widely used central smoothing technique, is adopted within the window to estimate the status value of the central point of the window. With the moving of the estimating point, the window moves. For the initial and last M points, the window does not move, and the initial $(2M+1)$ points and the last $(2M+1)$ points are used, respectively, for polynomial fitting calculation.

If the order of the polynomial model is too low, the fitting would be rough; on the contrary, the data noise would be incorporated into the model. According to statistics, if χ^2 of the estimated parameters is close to its freedom level, the order is considered proper. When $y_k, y_{k+1}, \ldots, y_{k+N}$ of $N+1$ measurement value is used to conduct n-order polynomial fitting, the estimated data is substituted into the following formula to calculate χ^2:

$$\chi^2 = \sum_{i=1}^{N} \left[\frac{y_i - (a_1 x_i^n + a_2 x_i^{n-1} + \cdots + a_n x_i + a_{n+1})}{\Delta y_i} \right]^2 \tag{2.18}$$

For χ^2 of a proper order, the degree of freedom $(N-n)$ should be taken as expected value. Therefore, start incrementally from $n=2$ to calculate χ^2 of different orders. Judge the approximation level of $1-p(\chi^2<(N-n))$ to 0.5. In the actual calculation, F test is adopted to test whether the two matrix variances are equal. According to statistics, $F_n = \frac{(\chi^2_{n-1}-\chi^2_n)}{\chi^2_n/(N-n)}$ complies with F distribution with freedom level as $(1, N-n)$, where n is current order, $N+1$ is the number of fitting data, and χ^2_n is χ^2 with n orders.

With the significant level α, determine the denial field of $F(1, N-n)$. If $F_n \le F_{\frac{\alpha}{2}}(1, N-n)$ or $F_n \ge F_{1-\frac{\alpha}{2}}(1, N-n)$, where $F_{\frac{\alpha}{2}}$ satisfies $P\{F \le F_{\frac{\alpha}{2}}\} = \frac{\alpha}{2}$, when F_n value is in the denial field, increase the order by 1, or current order is considered optimal.

2.2.3 Comprehensive Weighting Method to Fill the Missing Values

The fitting results from the above-mentioned two methods are weighted with the following formula:

$$X_i = \alpha X_{1i} + \beta X_{2i} \tag{2.19}$$

In the formula, X_i stands for the data after fitting at time i; α, β for weights; X_{1i}, X_{2i} for fitting values of BP neural network and least square method at time i, respectively.

Weights are chosen based on the following principles: when the data are too long and many data are missing, the fitting data weight of BP neural network is relatively large, i.e., $\alpha > \beta$; on the contrary, the fitting data weight of least square method is relatively large, i.e., $\alpha < \beta$. Weights can be chosen based on the results of massive data training.

This data filling method can take advantage of not only the excellent data fitting ability of the neural network, but also the least square method to make up for the deficiencies of the fitting results of a neural network with relatively few data, thus making the fitting data more reasonable and reliable.

2.2.4 Simulated Analysis and Conclusion

The missing values for pitch angle in the flight data are simulated, and the flight data curve including missing values is shown in Fig. 2.10. According to the figure, there are numerous missing values in the pitch angle data within 500 s, which has severe influence upon the data continuity and interferes with the follow-on data processing.

Figure 2.11 shows the effects of filling missing data through comprehensive weighting, neural network and least square polynomial fitting. It is shown that compared with the processing results of least square polynomial fitting and neural

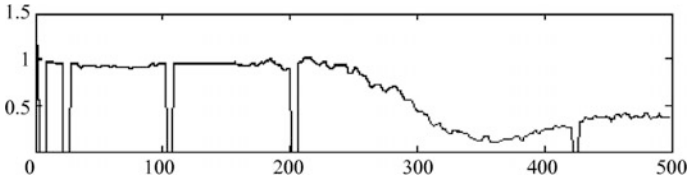


Fig. 2.10 Data curve generated by missing value simulation

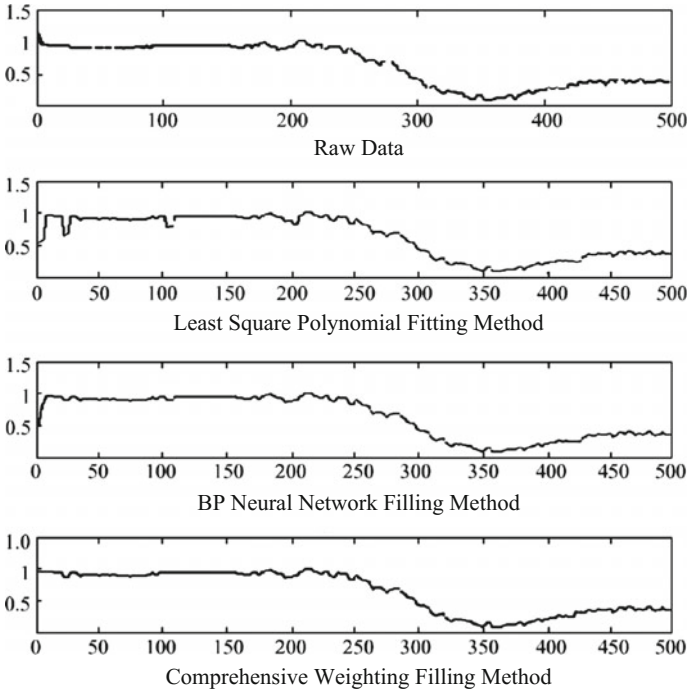


Fig. 2.11 Contrast of data filling effects

network, the data curve generated by comprehensive weighting data filling is more continuous and conformable to the raw and pure data. It reflects relatively true conditions of flight data and can effectively make up for the deficiencies of FDR data while guaranteeing data integrity, thus improving accuracy and reliability of the data.

2.3 Flight Data Extension Method Based on Virtual Sensor Technologies

For certain aircraft, the types and quantity of the parameters such as continuous and discrete data which can be collected by FDRS are relatively few and these parameters are far from enough to establish a relatively complete monitoring model. Moreover, when the analog sensor fails or drifts obviously, data cannot be acquired from this information source or even if data can be acquired, they cannot be used as valid monitoring data. Considering these problems, this section introduces the method to extend the flight data based on virtual sensor technology.

2.3.1 Overview of Virtual Sensor Technologies

A tolerance system is usually designed with redundant hardware, which means the need for additional hardware, such as computer, actuator, and sensor, so that the system can recover from partial damage. Additional sensors are often used to produce redundant output to conduct comparison in consistency. Although hardware redundancy is an important means for a tolerance system, it costs heavily in the aspects of funding, software, space for additional equipment, and maintenance. Virtual sensor-based software can partially replace some sensor hardware, thus reducing the requirements for cost and space. Moreover, it can reconstruct missing or hard-to-acquire measurement data with available data. As virtual sensors can be used to replace some hardware devices to reduce requirements for space and cost and meanwhile they have good performance, they have been widely applied in various fields such as communications, trouble diagnosis of sensor systems, robot systems, biochemical systems, target tracking, industrial manufacturing, medical care, noise control, and so on.

Virtual sensing, a signal processing technology, is used to estimate the response of a system's inability or difficulty to accommodate a physical sensor. Parameter estimation commonly starts from $Q = \{m_j : j = 1, 2, \ldots, n\}$, the set of measurement vectors m_j of $(m \times 1)$ to estimate the parameter vector f of $(t \times 1)$. The measurement vectors are the actual outputs of the physical sensor while n stands for the number of measurements. To implement the virtual sensor technology, it is assumed that $f = g(Q_s)$, mapping from Q_s, any subset of the measurement set Q, to

the parameter vector f, is known. Thus, the virtual sensor technology divides the estimation problem into two separate parts. At first, calculate the parameter vector f (known as virtual sensor measurement) and the corresponding variance matrix according to the mapping $f = g(Q_s)$; then, calculate the parameter with smallest variance based on all the sensor measurements.

2.3.2 *Virtual Flight Data Extension Based on Mathematical Model*

The values of the virtual sensor come from the self-corrective real-time model of the engine and its theory is shown in Fig. 2.12. Suppose the engine model is a real-time component-level model with relatively high stability and dynamic precision, the status estimator conducts online estimation of the engine status by controlling input, based on the error between the output of the real engine and that of the engine model. Then the estimator takes corrected status value as benchmark for iterative calculation and thermal parameter calculation of the model.

While designing a virtual sensor, initialize the engine model first to make it consistent with the operational state of the engine. The input of the model includes the operational environment parameters (such as height, Mach number) of the engine and control volume (such as fuel), and meanwhile, the optimal estimated value of the status volume received by the model from the status estimator and the corrected initial values serve as benchmark to calculate the parameters of each section of the engine. In other words, the optimal estimated value serves as calculating benchmark for a component-level model. According to Huang Xianghua's research and calculation, after only one-step calculation, the component-level model can get thermal parameters close to those of a real engine, and thus the model can track a real engine.

Kalman algorithm is used for status estimation to linearize the component-level model first and then to establish a state space model of the engine status:

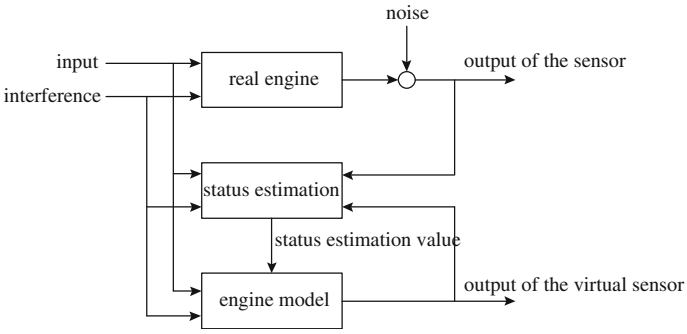


Fig. 2.12 Virtual flight data extension model based on the engine model

$$\delta_x = A\delta_x + B\delta_u + w \tag{2.20}$$

$$y = C\delta_x + D\delta_u + v \tag{2.21}$$

In the formula, w and v stand, respectively, for dynamic noise vector and measurement noise vector, which are irrelevant Gaussian white noise with the mean value as zero. Similar normalized parameters are adopted for status value, input value, and output value, which means a linear model with small deviance is established with the stable operating point of the engine as benchmark.

The discrete optimal filtering estimation algorithm is as follows:

$$\hat{\delta}_x(k+1) = \hat{\delta}_x(k+1,k) + K(k)(\delta_y(k) - \hat{\delta}_y(k)) \tag{2.22}$$

Under real-time control, it takes a long time to calculate the optimal filtering estimation value, even longer than the permitted sampling time. While a constant gain suboptimal filter is used for filtering, i.e., $K(k) = K_d$ (K_d stands for the stable constant gain matrix for optimal filtering), this filtering is suboptimal before it gets stable state. As Kalman filter itself is robust in nature, the constant gain suboptimal filter can relatively effectively estimate the status value.

Mathematic model is an important tool for modern aeroengine manufacturing, testing, and operation. Precise mathematic model guarantees the precise extension of the virtual flight data based on the engine model. However, because of technological security, it is rather difficult to thoroughly master the precise mathematic model that can fully reflect the operational state and process of an engine, giving rise to the difficulty in fault analysis in the process of teaching, research, and application. At present, the appearance of identification methods such as least square of an engine model based on flight data and Volterra series model provides a foundation for the research of precision engine modeling. However, these methods are not practical enough and must be aided greatly by subject experts. This is one of the important topics that need to be studied.

2.3.3 Virtual Flight Data Extension Based on BP Network

1. Problem Description

Usually, there is certain mapping relationship among various parameters which describe the aircraft condition. So long as there is a precise system mathematical model, the required virtual flight data can be estimated with known data of one or more parameters. However, as the mathematic models of each subsystem aboard the aircraft are usually complex nonlinear models, it is relatively complicated to use traditional modeling method to establish a model and the accuracy of estimated data will be influenced.

BP neural network, a parallel computation model, has many unique advantages: excellent nonlinear mapping ability; few requirements for empirical knowledge about the modeling object; no need for knowledge about the structure, parameters and dynamic features of the modeling object; and only with input and output data of the object, the relationship between the input and output of the network can be acquired through the learning function of the network itself. Given the superb advantage of the nonlinear feature of the neural network, the method to estimate virtual flight data based on BP neural network will be introduced in the following paragraphs.

2. Algorithm to Acquire Virtual Data Based on BP Network

The basic tenet to acquire virtual data based on neural network is: with data of all parameters except those to be estimated as input of the network, and data of the parameters to be estimated as output, utilize the known data training network in the system and when the network meets requirements, input into the network the parameters to be estimated and other parameters to acquire estimated values of the virtual data.

1) Data Normalization

BP neural network adjusts the connecting weight according to gradient descent method to minimize the error function. During the process of adjusting the connecting weight, when the input x_i is relatively large, the input of next-layer node transferring function is relatively large. The output becomes saturated, close to 1. To meet the requirements of BP neural network, the input data must be normalized. Fuzzy membership normalization method is adopted. The following formula is often used for data normalization:

$$\mu_i(x) = \frac{k(x_{\mathrm{i}} - \alpha_i)^2}{1 + k(x_i - \alpha_i)^2} \tag{2.23}$$

In the formula, x_i stands for input sample value, α_i for the maximum of the input sample value, k for proportional coefficient.

While the formula is used to normalize the flight data, only the degree of difference between the recorded data and the estimated data is taken into consideration, and the deviation direction is not considered. Therefore, some information will be lost after normalization processing. To reduce lose of information, the formula is modified as follows:

$$\mu_i(x) = \frac{k(x_i - \alpha_i)|x_i - \alpha_i|}{1 + k(x_i - \alpha_i)^2} \tag{2.24}$$

In the formula, k is a parameter determined by the environment. The flight data can be normalized into the segment [−1,1] by selecting proper value of k.

2) Virtual Data Acquisition

A three-layer BP neural network is adopted. Suppose *m, n,* and *u* stand for the node number of the input layer, output layer, and hidden layer, respectively. The node output function of the output layer is

$$c_j = f\left(\sum_{r=1}^{n} V_{rj} \cdot b_r + \theta_j\right) \ (j = 1, \ldots, n) \tag{2.25}$$

In the formula, V_{rj} stands for the connecting weight between the hidden layer and the output layer, b_r for the node value of the hidden layer, θ_j for the node threshold of the output layer, and c_j for the node value of the output layer.

The node output function of the hidden layer is

$$b_r = f\left(m\sum_{i=1}^{m} W_{ir} \cdot a_i + T_r\right) \quad (r = 1, \ldots, u) \tag{2.26}$$

In the formula, W_{ir} stands for the connecting weight between the input layer and the hidden layer, α_i for the node value of the input layer, and T_r for the node threshold of the hidden layer. The action function $f(x)$ usually takes the form of the sigmoid function $f(x) = (1 + e^{-x})^{-1}$.

The learning process of the neural network is: While the neural network is learning, if the error between the value of the output layer and the expected value is larger than the permitted error, the network will adjust the connecting weight and the threshold of the nodes in the connecting layer. In this way, the error of the node in the output layer will be reversely propagation for the input layer to distribute the error to each connecting node, thus figuring out the error reference of each connecting node. Then adjust them correspondingly according to the weight and threshold of each connecting node to enable the network to meet the output requirements, thus realizing the mapping of $A^{(k)} \rightarrow C^{(k)} (k = 1, 2, \ldots m)$, where $A^{(k)} = (a_1^{(k)}, a_2^{(k)}, \ldots, a_m^{(k)})$, $C^{(k)} = (c_1^{(k)}, c_2^{(k)}, \ldots, c_n^{(k)})$, $a_i^{(k)} \in R$,$c_j^{(k)} \in R$, (R is real number field.)

The concrete calculation includes the following steps:

Step 1: set randomly a small value between 0 and 1 for $W_{ir,}$ $T_{r,}$ $V_{rj,}$ and $\theta_{j.}$

Step 2: input the value of $\alpha_i^{(k)}$ into the node of the input layer and calculate forward b_r and c_j in turn.

Step 3: calculate the error between the node output value and the expected output value of the input layer $d_j : d_j = c_j \cdot (1 - c_j) \cdot (c_j^{(k)} - c_j)$.

Step 4: back error propagation for the nodes of the hidden layer e_r: $e_r = b_r \cdot (1 - b_r) \cdot \left(\sum_{j=1}^{n} V_{rj} \cdot d_j\right)$.

Step 5: to reduce vibration during the learning process, Rumelbart with inertial impulse technology is adopted for adjusting the weight and threshold.

Inertial impulse is added to filter the high-frequency vibration during the learning process, thus getting the maximal learning rate to accelerate learning. Here, the general form for adjusting the weight is

$$\Delta V_{rj}(t+1) = -\lambda \frac{\partial E}{\partial V_{rj}} + \eta \Delta V_{rj}(t) = \sum_{k=1}^{m} (\lambda b_r d_j) + \eta \Delta V_{rj}(t) \quad (2.27)$$

$$\Delta W_{ir}(t+1) = -\beta \frac{\partial E}{\partial V_{rj}} + \delta \Delta W_{ir}(t) = \sum_{k=1}^{m} (\beta a_i e_r) + \delta \Delta W_{ir}(t) \quad (2.28)$$

The general form for adjusting the threshold is

$$\Delta \theta_j(t+1) = \sum_{i=1}^{m} \lambda d_j + \eta \Delta \theta_j(t) \quad (2.29)$$

$$\Delta T_r(t+1) = \sum_{k=1}^{m} \beta e_j + \delta \Delta T_r(t) \quad (2.30)$$

In the formula, λ and β stand for learning rates, usually between 0 and 1; η and δ for dynamic factor; E for the sum of squared error in the whole training set, i.e.,

$$E = \frac{1}{2} \sum_{k=1}^{m} \sum_{j=1}^{n} (C_j^k - C_j)^2, \quad E = \sum_{k=1}^{m} E_k \quad (2.31)$$

Step 6: repeat Steps 2–5 and terminate training when the error d_j meets the requirements or becomes 0.
Select the flight data from the same sortie as training samples. With the data of known parameters related to missing parameters as input and the data of missing parameters to be estimated as output, conduct the above-mentioned iterative process repeatedly until the requirements are met. In this way, a matching neural network model can be acquired.

Step 7: input the data of relevant parameters at the same moment of the to-be-estimated data into the trained network. The output value is the estimated value.

3. Experiment and Result Analysis

Take an engine system for example, adopt the given three-layer BP neural network model to simulate, with 6500 s data recorded by the FDRS as data samples, where the first 6000 s data as training samples, and the remaining 500 s data as test samples; the throttle angle and rotating speed of the engine as input value of

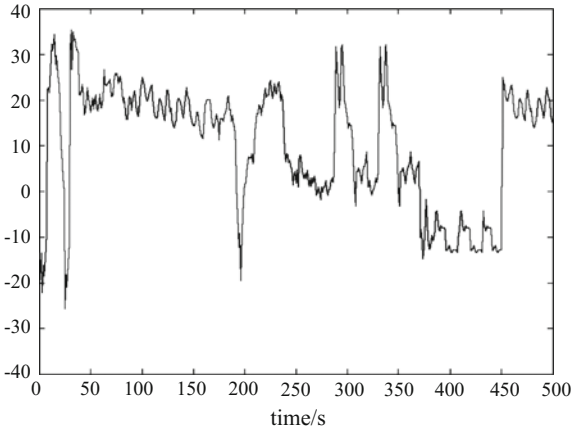


Fig. 2.13 Curve of the error between the estimated value and the actual value of the virtual data

the model, exhaust gas temperature as output value; 2 input nodes, 25 hidden nodes, and 1 output node.

Standardize the data sample with $k = 0.00001$, $\alpha_1 = 110$, $\alpha_2 = 8000$, and $\alpha_3 = 900$. Then, input the first 6000 s standardized data into the above-mentioned network model for training, with 0.0001 for E, the sum of squared error, learning rate $\lambda = \beta = 0.01$, dynamic factor $\eta = \delta = 0.75$. After 682 iterations, the network meets the requirements. Input the test samples into the trained network to estimate the virtual parameters. Figure 2.13 shows the curve of the error between the estimated value and the actual value of the 500 s virtual data with maximum relative error as 0.038%.

The experiment shows that when the sensor for exhaust gas temperature has trouble, resulting in invalid data, it is feasible to acquire the virtual data based on BP network to meet application requirements.

2.4 Self-expanding Genetic Algorithm for Feature Selection in Monitoring Flight Data Capacity

It is an issue of classifying or clustering model identifications in nature to utilize flight data to monitor the aircraft equipment condition or diagnose a trouble. The key of the issue lies in the establishment of mapping from the equipment data to trouble symptom space. The condition can be better studied by selecting the features at different conditions represented by the flight data. Therefore, the key feature parameters distinguishing different patterns are critical input. Generally speaking, as flight data are multivariable high-dimensional time series, numerous feature parameters are associated with each other. Too many input values used for model identification will reduce operability and effectiveness of the results. Therefore, an important preprocessing task is to optimize representative feature parameter sets.

The self-expanding genetic algorithm for feature selection in monitoring flight data capacity is introduced in this section.

2.4.1 Feature Selection and Genetic Algorithm

1. Research Domain of Feature Selection

Objective data can be indicated by vectors, and these vector sets may take the form of certain linear or nonlinear manifold structure. Because of the strong association caused by the model and other potential factors, there are always greatly redundant observing vectors. To explore linear or nonlinear structure from limited discrete dot sets (possibly with certain noise interference) to classify, clustering and data visualization becomes a challenging problem in supervised learning, that is, feature dimension reduction problem which needs to be solved for statistical pattern identification. Dimension reduction methods are divided into two categories: feature selection and feature extraction. Feature selection is subdivided into feature filtering and feature checking while feature extraction includes linear and nonlinear feature extraction. The optimal methods for feature filtering include exhaustive method, branch and bound algorithm, feature selection based on genetic algorithm, etc. Linear feature extraction methods include principal component analysis, Fisher identification analysis, factor analysis, multidimensional scaling, etc. Nonlinear feature extraction methods consist of principal component analysis based on nucleus and identification analysis based on nucleus.

Optimal subset needs to be established for feature selection by starting from a null set to establish d features incrementally, or starting from measured full-value set to delete redundant features gradually. The methods can be divided into two categories: filter and wrapper, i.e., filter and checking, as shown in Fig. 2.14.

Based on the optimal feature subset acquired by identification function, feature filtering only depends on the statistical feature of training samples, independent of the learning algorithm of the classification device. However, feature checking estimates the strong and weak points of each feature subset for actual classification effects based on the learning algorithm of the classification device. It is not only

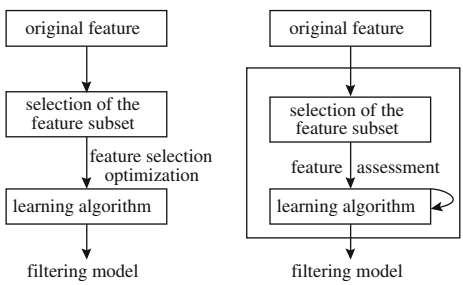


Fig. 2.14 Two feature selection models

related with the statistical feature of training samples, but also closely related with the statistical feature of test sample and learning algorithm.

Feature selection can be described mathematically: given measurement values for p variants, find the optimal subset of d. Optimization is conducted based on the set of subset X_d of d, that is to find subset $\overline{X}_d$.

$$J(\overline{X}_d) = \max_{X \in X_d} J(X) \tag{2.32}$$

J rule function is usually based on measurement of distance or distribution difference. Formula 2.33 is the possible number of subsets. To select 10 features from 25 variants means that 3,268,760 subsets should be considered and optimal rule should be calculated for each subset.

$$n_d = \frac{p!}{(p-d)!d!} \tag{2.33}$$

While monitoring the aircraft condition, there are some special points for reducing the dimensions of feature parameters of flight data. First, with limited trouble samples from flight data, to design classifier with numerous features does not conform to actual requirements. Second, sometimes, intuitional physical meaning cannot be acquired through feature selection methods. Therefore, it is difficult to figure out the influence of each feature upon the classification. Third, as the number of the recorded parameters in the flight data is not the minimal dimension by which pattern cannot be distinguished, eliminating redundant or interference features helps to identify the aircraft condition model accurately. Given the above-mentioned factors, feature selection becomes the primary means to reduce the dimensions of feature parameters of flight data.

2. Feature Selection Based on Genetic Algorithm

Genetic algorithm (GA) is a highly parallel, random, and self-adaptive search algorithm based on natural selection and natural heredity, which stimulates the principle of survival of the fittest during the biologic evolution process to conduct simultaneous optimization of multiple parameters and populations. During the research process of evolution calculation, Holland proposed bit string coding technology which not only applies to mutation operation but also to crossover operation, emphasizing that crossover should be used as the primary genetic operation. After applied to the research of self-adaptive performance of natural and artificial systems, the algorithm, known as genetic algorithm, has been successfully applied to such aspects as optimization and learning. The algorithm includes genetic algorithm, evolution strategy, evolution planning, and genetic program design.

Genetic algorithm is widely used in feature selection. However, with new samples, a standard algorithm is usually adopted in feature selection for all samples. With relatively great interference of new samples, the original speed and results of feature selection will be influenced. Therefore, self-expanding genetic algorithm

based on increasing of samples is put forward. When there are new samples, excellent partial features are reselected based on the original feature selection to avoid repeated feature selection for all samples, thus the fitness level is increased and iteration times are reduced greatly.

2.4.2 Self-expanding Genetic Algorithm

Selection of features indicating aircraft troubles based on flight data has the following characteristics: first, current flight data may not include all types of trouble states; second, new trouble modes included in the trouble data samples will influence the results of original feature selection. With accession of new data in the flight database, there must be many new positive and negative samples. Repeated optimization will lead to complicated calculation and the evolution features of the genetic algorithm will not be fully utilized.

1. Structure of Self-Expanding Genetic Algorithm

Self-expanding genetic algorithm can be used to solve the above-mentioned problems effectively. The basic principle of the algorithm is to use acquired features to classify new samples. If classification accuracy decreases, select a certain proportion of excellent individuals according to fitness level of the individuals from the population acquired after the previous execution of the genetic algorithm. These selected individuals will be used as the initial individuals for this execution while other individuals in the initial population will be generated randomly as shown in Fig. 2.15. The self-expanding genetic algorithm mainly includes the following steps:

(1) Initialize the control parameters such as population number, threshold, mutation probability.
(2) Generate the initial population G randomly, calculate the fitness level f of each individual in the population and the average fitness level of the population.
(3) Use selection, crossover, and mutation operators to generate filial generation and then evaluate the filial generation, i.e., calculate the fitness level of the filial generation. If the fitness of the newly acquired filial generation is better than the parent's, use the filial generation to substitute the parent, otherwise accept the filial generation.
(4) Judge the converging condition—If the value difference of the average fitness level between neighboring generations is smaller than given threshold, searching is terminated. Sort the results according to fitness values and select the result with the highest fitness value as the optimal solution.
(5) Incorporate the newly acquired samples into the original sample set and classify them with acquired features. If classification accuracy does not decrease, current features are valid; whereas, select a certain proportion of excellent individuals according to the fitness level of the individuals from the population

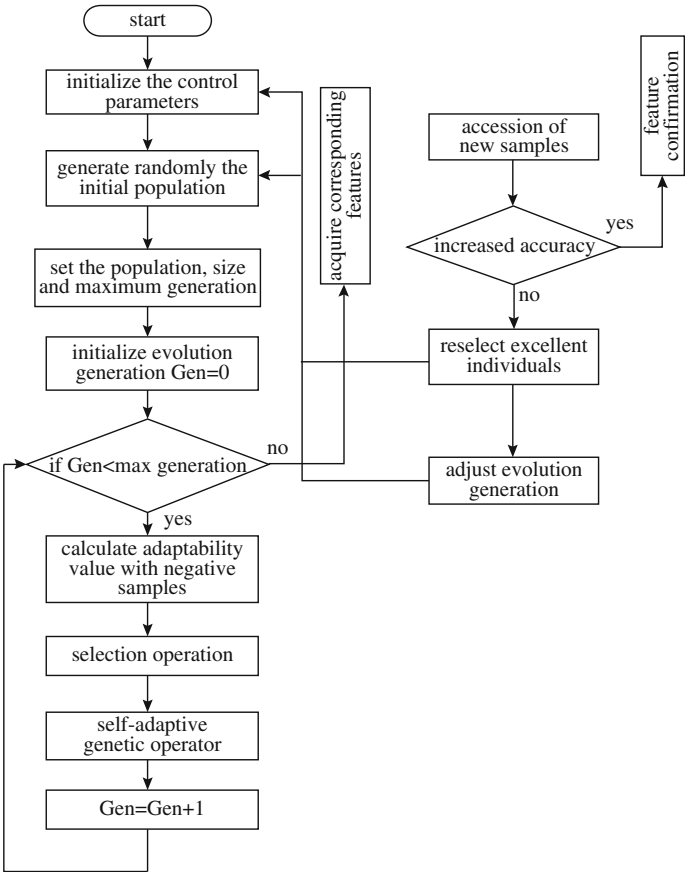


Fig. 2.15 Process of self-expanding genetic algorithm

acquired after the previous execution of the genetic algorithm. These selected individuals will be used as the initial individuals for this execution while other individuals in the initial population will be generated randomly.

(6) Adjust the maximum generation $n_1 = n_0 * (s_1/s_0)$, where n_0 stands for the original maximum generation, s_1 for current sample number, and s_0 for original sample number. Do not terminate the procedure until one of the following conditions is satisfied: first, the evolution generation reaches the maximum given evolution generation; second, the maximum fitness level is no less than the fitness level of the original rule and the maximum fitness level will not change during the given N-generation evolution process.

2. Coding, Evaluating, and Operator Designing of Self-Expanding Genetic Algorithm

The operation, representation, and evaluation of self-expanding genetic algorithm include the following aspects:

(1) Coding. Binary coding, a coding method most widely used in the genetic algorithm, is relatively simple and easy to understand. Although binary coded string will be very long while solving high-dimensional optimization problem, the length of flight data trouble sample itself is relatively limited and cannot be expanded infinitely. Therefore, binary strings can be used for coding.
(2) Generation of the initial population. The initial population can be generated in this way: generate randomly a binary string (individual) with the length D, with d as the number of "1", and continuously produce M individuals of this sample. M stands for the population size, which influences the final result and execution efficiency of the genetic algorithm. Too small M may easily lead to partial optimization, thus influencing its optimization performance; otherwise, the computation will be very complicated. Suppose $M = tn$, where t is a real number between 1 and 2, then at least n samples are needed to distribute the initial population in the problem space rather evenly.
(3) Confirmation of evaluation functions. During the early phase of the algorithm operation, the fitness levels differ greatly, possibly resulting in prematuration in the selection process. But during the late phase of the algorithm operation, the individual fitness levels may be very close, and the competitiveness of most individuals has no bearing on the optimal individual, thus possibly entering into random selection process. Linear scaling can be adopted to adjust individual fitness level in different operation phases.
(4) Chromosome selection operator. Proportional selection is a random playback sampling method in which the probability for selecting each individual is directly proportional to its fitness level. Because of random operation, this method has relatively large selection error, and sometimes even the individual with high fitness level cannot be selected. Therefore, optimal saving tactics is adopted to ensure that the optimal individuals acquired so far cannot be destroyed by genetic calculation such as crossover and mutation. This tactics is important to guarantee the convergence of the genetic algorithm. When the fitness level of the worst individual in current population is lower than the overall fitness level of the optimal individuals, the best individual is used to replace the worst individual in current population.
(5) Designing of chromosome crossover operator. The global search function of the genetic algorithm is directly dependent upon crossover algorithm, and LOX crossover operator should be adopted. The operator can be constructed in the following way: first, select a pair of parent individuals randomly; then, select a feature randomly, with which to conduct m operations for each parent individual to get a new string. Move leftward or rightward for L positions (L is determined randomly), but it is necessary to keep stable the distance between the operations corresponding to selected work. Use the operations corresponding to the features not selected in the second parent individual to replace

the original individuals one by one according to their original relative sequence to generate a descendant individual.

(6) Crossover probability and mutation probability. Self-adaptive crossover and mutation probability selection method is adopted. $f_{\max}$ stands for the maximum fitness level of the population, $\bar{f}$ for the average fitness level of the population, f_c for the higher fitness level in the two strings participating in the mutation, and f_{m} for the fitness level of the mutating string. The self-adaptive adjustment of p_c and p_m is inversely proportional to the convergence of the algorithm, and can effectively prevent the convergence from being partially minimal. The higher the fitness level of the individual is, the smaller the corresponding p_c and p_m are, and thus good evolution results are kept. The corresponding p_c and p_m can be acquired with the following formula in which $k_i = (i = 1, 2, 3, 4)$ stands for adjustable parameter which can be set with reference to typical genetic algorithm.

$$\begin{aligned} p_c &= \begin{cases} k_1(f_{\max} - f_c)/(f_{\max} - \bar{f}_c), & f_c \geq \bar{f} \\ k_2, & f_c < \bar{f} \end{cases} \\ p_m &= \begin{cases} k_3(f_{\max} - f_m)/(f_{\max} - \bar{f}_m), & f_m \geq \bar{f} \\ k_4, & f_m < \bar{f} \end{cases} \end{aligned} \tag{2.34}$$

2.4.3 Case Verification and Assessment

With the engine of a type of aircraft as the research object, continuous flight data include speed of the high-pressure rotor speed, throttle position, vibration speed of the aft pod, exhaust gas temperature; discrete type flight data include metal element concentration of lubricating oil. With the typical trouble of abnormal vibration value of the engine for example, verify the accuracy of self-expanding genetic algorithm. Data samples are taken from abnormal engine vibration values. Severe bearing attrition results in engine replacement ahead of schedule. Figure 2.16 shows a group of flight data samples under the stable airborne condition. The data samples are speed of the high-pressure rotor of the engine, throttle position, vibration speed of the aft pod, and exhaust gas temperature in sequence. The unit for horizontal coordinates is second (s).

Select 26 normal positive samples and 76 negative samples of the engine with the length of chromosome lchrom = 12, the size of the population popsize = 150, maximum generation Max(Gen) = 300. Figure 2.17 shows the curve of the evolution process with severe engine bearing attrition, where horizontal coordinates stand for evolution generation and vertical coordinates for the fitness level. The fitness level of the optimal individual is acquired: fitness = 68.3653258 with presentation string of the chromosome as “011011000100”. According to the

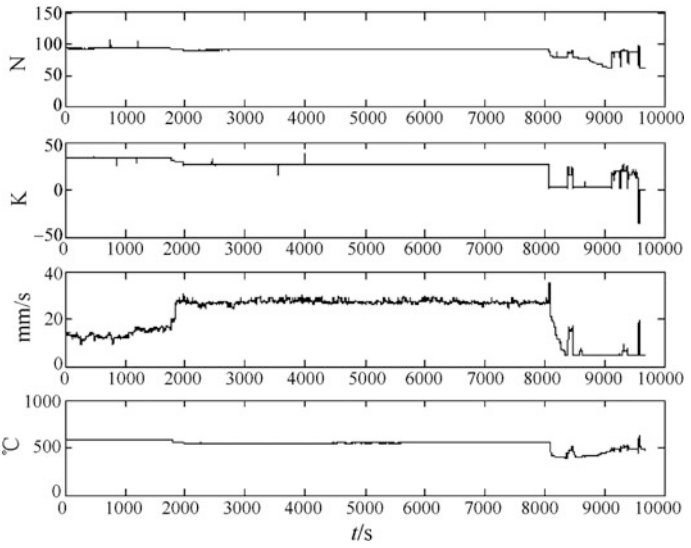


Fig. 2.16 Flight data samples

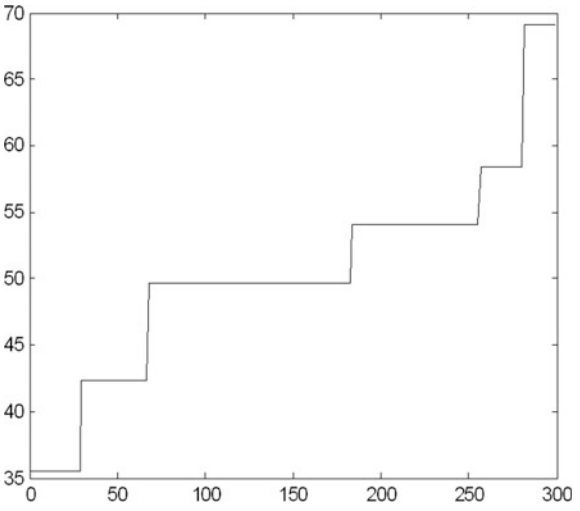


Fig. 2.17 Actual evolution results

classification generation rule, the above rule can be explained in this way: ($C_3 < 521.97 \&\& C_5 < 58.8115 \&\& C_8 > 44.8745$). Based on corresponding expertise, the physical meaning can be interpreted in this way: Under the stable airborne condition with rotor speed larger than 97.153% or smaller than 99.234%, percentage of vibration speed change per second larger than 24.454%, percentage of exhaust gas temperature change per second larger than 12.235% and smaller than

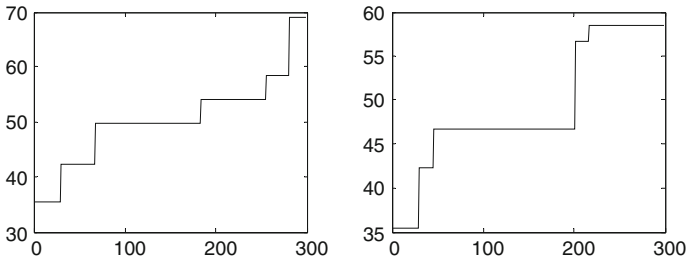


Fig. 2.18 Contrast between standard genetic algorithm and self-expanding algorithm

34.564%, the features acquired are rotor speed, vibration speed, and exhaust gas temperature, corresponding to the severe engine bearing attrition phenomena. The feature selection results completely conform to the knowledge of frontline experts.

Add 10 positive examples and 5 negative examples, thus increasing the number of positive and negative examples to 36 and 81, respectively, with other parameters unchanged. Figure 2.18 shows the contrast between self-expanding algorithm and standard genetic algorithm. It is obvious that self-expanding algorithm conducts efficient feature selection based on the original feature selection, and thus the starting point of the fitness level in the evolution process becomes higher, the iteration times greatly decrease and interference of noise data is eliminated.

Use a binary-class support vector machine (SVM) to conduct classification with the three acquired features. The samples are taken from the flight data of a type of aircraft accumulated in recent 2 years and each sortie corresponds to a group of samples. Table 2.2 shows the classification results, where SGA stands for self-expanding genetic algorithm, GA for standard genetic algorithm, and SVM for binary-class support vector machine. The table shows that because of the influence of interference parameters, when the samples increase to certain number, the classification accuracy decreases with the adoption of only support vector machine (SVM). The classification accuracy of feature sets after feature selection of genetic algorithm increases, which indicates that feature selection greatly reduces the influence of interference factors and thus classification accuracy is enhanced. While using the given self-expanding genetic algorithm for feature selection, the number of features further decreases, but the classification accuracy is greatly enhanced.

2.5 Chaotic Property Analysis of Flight Data

Phase space reconstruction usually uses a sequence to analyze the dynamic property of the original system and after selecting effective features, analyzes the property of the system from which the data comes, including chaotic prediction, estimation of dynamic invariants, characterizes discrimination of chaotic signals, etc. Actually, the time series generated by the flight data often reflects the dynamic property of the

Table 2.2 Contrast of classification accuracy with or without adoption of feature selection

Method	Feature	Classification accuracy with 800 groups of samples	Classification accuracy with 1500 groups of samples	Classification accuracy with 2000 groups of samples	Classification accuracy with 3000 groups of samples
SVM	12	78.52	81.38	83.35	85.57
GA+SVM	8	83.82	85.25	87.49	91.12
SGA+SVM	7	93.17	94.26	95.37	96.54

original system. Phase space reconstruction method can be used to analyze the features of the flight data qualitatively and quantitatively. In this section, based on phase space reconstruction of the original flight data, typical flight data are analyzed with such methods as power spectrum graphs and Maximum Lyapunov Exponent. It is proven that they have typical chaotic property, thus exploring a new way to further study flight data processing technologies.

2.5.1 Mathematical Description of Phase Space Reconstruction of the Chaotic Series

1. Phase Space of the Chaotic Series

Chaos, a kind of aperiodic, macroscopic, and spatiotemporal behavior, is generated by a nonlinear dynamic process inherent in a system. It integrates skillfully the apparent disorder and inherent regularity and reflects the inherent randomness of a nonlinear dynamic system. It is a seeming irregularity and random-like process existing in a deterministic system. A system with both sensitivity to the original values and aperiodic movement is a chaotic system.

The state of a system at certain moment is known as phase, and the geometric space determining the state is known as phase space. Theoretically speaking, the phase space of a nonlinear system may have a very high, even infinite dimension. Actually, a sequence is often acquired: $x_1, x_2, \ldots, x_n$. Formula 2.35 stands for a time series outputted from a continuous dynamic system: $x(1), x(2), \ldots, x(M)$, where M refers to the length of the time series. While reconstructing the phase space, appropriate embedding dimension m and time delay τ should be selected.

$$\begin{aligned} X(1) &= [x(1), x(1+\tau), x(1+2\tau), \ldots, x(1+(m-1)\tau)], \\ X(2) &= [x(2), x(2+\tau), x(2+2\tau), \ldots, x(2+(m-1)\tau)], \\ &\cdots\cdots \\ X(M) &= [x(M), x(M+\tau), x(M+2\tau), \ldots, x(M+(m-1)\tau)] \end{aligned} \tag{2.35}$$

Evaluating the effects of phase space reconstruction is to verify whether some invariants of the original phase space can remain unchanged. A sequence can be used directly to analyze the system. However, the nonlinear time series is

comprehensive reflection of many interacting physical factors and contains the locus of all variants participating in the movement. Therefore, this time series must be expanded into three- or more-dimensional phase space to fully present the time series information. This is phase space reconstruction of the time series.

In 1981, Takens F. put forward a delay embedding theorem which gives the conditions under which a chaotic dynamic system can be reconstructed from a sequence of observations of the state of a dynamic system. This method has become the primary and most fundamental method for phase space reconstruction. For example, as the chaotic property of voice is mainly generated by incentive airflow, linear prediction error signals are equivalent to incentive signals. Linear prediction method eliminates the influence of vocal tract resonance during the process of voicing. The research findings include: (1) Different voice has different chaotic attractor; (2) the attractor of voiced sound has a closed torus while unvoiced sound has an irregular curve, as shown in Fig. 2.19.

2. Takens Reconstruction Theorem

Suppose M is an m-dimensional manifold, for transform pair (ϕ, y), $\phi : M \to M$ is a smooth diffeomorphism and y is a smooth function in M, then $\Phi_{(\phi,y)} : M \to R^{2m+1}$ is an embedding. Here, $\Phi_{(\phi,y)} = \left\{y(x), y[\phi(x)], \ldots, y\left[\phi^{2m}(x)\right]\right\}$, where ϕ' is a stream of x, ϕ corresponds to the dynamic relationship of a dynamic system, M to the attractor of the system, and y to the function relationship between the system state and the measurement data. Embed a chaotic time series of a single variant $x_1, x_2, \ldots\ldots x_n$ in m-dimensional space to get the phase space locus for N phase points:

$$Y_i = (x_i, x_{i+\tau}, \ldots, x_{i+(m-1)\tau})^T \tag{2.36}$$

In the formula, $i = 1, 2, \ldots, N$; $N = n - (m-1)\tau$; Y_i stands for the phase space vector after reconstruction; τ for delay time; m for embedding dimension; n for points of the original time series; N for the number of phase space vectors after reconstruction. Therefore, the phase space matrix is acquired:

Fig. 2.19 Phase space locus of chaotic attractors for unvoiced and voiced sound

$$\begin{cases} Y_1 = (x_1, x_{1+\tau}, \ldots, x_{1+(m-1)\tau})^T \\ Y_2 = (x_2, x_{2+\tau}, \ldots, x_{2+(m-1)\tau})^T \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ Y_N = (x_N, x_{N+\tau}, \ldots, x_{N+(m-1)\tau})^T \end{cases} \tag{2.37}$$

According to Takens Theorem, with proper τ and m selected, the dynamic form of the original system can be restored with topological equivalence to identify the basic property of the original dynamic system. Moreover, during the evolution process of a dynamic system, all variants are inherently correlated and the evolution of any variant in the system is dependent upon its interaction with other variants in the system. Therefore, evolution information of the system is contained during the evolution process of every variant.

To get geometric structure of the phase space of a dynamic system from a sequence, Packard et al. adopted a time-delaying technology to reconstruct phase space. To reconstruct an equivalent state space, only the change of one variant in the system needs to be considered. Delay phase diagram method is used to process the data with fixed delay as new unidimensional information, i.e., map the data of a single variant on a vector point in multidimensional space. Phase space can be reconstructed from single variants and the vector points in the reconstructed phase space have the same property as those in the original true space.

3. Defining Embedding Dimension of the Phase Space and Selection of Optimal Time Delay

To reconstruct phase space of signals based on chaotic theorem, embedding dimension and delay time should be determined at first. In 1983, to extract feature exponents of time series, Grassberger et al. presented the G-P algorithm to calculate fractal dimension of strange attractors. In 1985, Wolf raised an algorithm to calculate Lyapunov exponent of time series. Phase space reconstruction is an important foundation for exponent calculation. Without effective phase space reconstruction of the original system, the above-mentioned calculation cannot be executed. Packard et al. suggested using the delay coordinate of certain variant in the original system to reconstruct phase space. Takens demonstrated that a proper embedding dimension can be found. That means, if the dimension of the delay coordinate $m \geq 2d + 1$(d stands for the dimension of the original dynamic system), regular locus (attractors) in the embedding space can be recovered. However, according to Takens Theorem, the precondition for reconstruction is to get pure data without length and accuracy limits for certain variant in the original dynamic system, which cannot be satisfied in an actual system. Therefore, research must be conducted to determine how long the actual data should be to meet actual reconstruction requirements. In 1988, Smith stated a harsh condition: up to 42^m (m stands for embedding dimension) data points are needed, which was proven unnecessary by Tsonis. In 1990, Ramsey stated that limited data points can be used to get reliable reconstructed phase space; Nvrengerg and Essex presented the minimal data length necessary for reconstruction, for example, for a system with $d < 4$, at

least 4000 data points are needed for reliable reconstruction. In 1993, M.T. Rosenstein et al. put forward a method to use a small amount of data to calculate the maximum Lyapunov exponent. For a discrete dynamic system, this method does need a small amount of data. For Henon mapping, Logistic mapping, etc., only 500 data points (maybe fewer) are needed to accurately figure out the maximum Lyapunov exponent. But for continuous dynamic systems such as Lorenz system and Rossler system, the required amount of data is roughly equivalent to the amount of data stated by Nvrengerg.

While reconstructing phase space, selection of proper time delay τ and embedding dimension m has a direct bearing on the accuracy of variants which describe the feature of strange attractors after phase space reconstruction. For selection of τ and m, there are mainly two viewpoints:

According to the first viewpoint, τ and m are independent of each other, and their selection can be conducted independently (Takens proved that for the time series without length limits and noise interference, τ and m are independent of each other). At present, selection of time delay and embedding dimension is based on three rules: (1) sequence correlation methods such as autocorrelation method, mutual information method, and high order correlation method; (2) phase space expansion methods such as factor filling method, vibration measurement method, average displacement method, and SVF method; (3) complex autocorrelation and unbiased complex autocorrelation method.

According to the second viewpoint, τ and m are correlated because the actual time series has limited length and is subject to the influence of various noises. Numerous experiments demonstrated that the relationship between τ and m is closely related to the time window (tw) for phase space reconstruction. For particular time series, tw is relatively stable. The improper matching of τ with m will have a direct influence upon the equivalent relationship between the structure of reconstructed phase space and the original space. Therefore, the combined algorithms of τ and m, such as time window method, C2C method, and automatic algorithm of embedding dimension and time delay, are generated correspondingly.

Most researchers think the second viewpoint is more practical and rational in engineering practice. At present, research of combined algorithms of embedding dimension and time delay is still one of the hot topics for analysis of chaotic time series.

4. Lyapunov Exponent Based on Phase Space Reconstruction

The basic feature of chaotic movement is the extreme sensitivity of movement to the initial condition. With a tiny change of the initial condition, the evolution locus of the system based on time will separate from the original locus at exponential speed and completely cover the actual state of the system after a certain amount of time, which indicates unpredictability of the long-term performance of the system. For unidimensional dynamic system F:

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \left| \frac{dF(x)}{dx} \right|_{x=x_i} \tag{2.38}$$

λ refers to Lyapunov exponent, which describes the separation degree of the exponent for each of the multiple iterations on average. When $\lambda \leq 0$, viewed as a whole

$$\left| \frac{dF(x)}{dx} \right| \leq 1 \tag{2.39}$$

Therefore, neighboring points will eventually get close and merge into one point, corresponding to stable fixed point and periodic point; $\lambda > 0$ indicates instability of the movement orbit and the orbit of neighboring points separate in exponential way, thus entering into chaos. The maximum Lyapunov exponent can be used to measure the sensitivity of the chaotic system to the initial value. The larger the exponent value is and the stronger the chaotic property is, the higher the sensitivity is; otherwise, the lower the sensitivity is. Lyapunov exponent indicates the separation level of two neighboring points which evolve with time in the reconstructed phase space. Lyapunov exponent is a pedigree $\lambda_i (i = 1, 2, \ldots, p-1)$, where λ_i indicates the separation level of neighboring orbits which evolve with time in each dimension of the m-dimensional space. If only the maximum exponent in the pedigree is positive, the system is a unidimensional chaotic system; if two or more Lyapunov exponents are positive, the system is a multidimensional chaotic or super chaotic system; or the system is not chaotic. Wolf et al. proposed that the method based on phase locus evolution be used to estimate the maximum Lyapunov exponent $\lambda_{\max}$ with the following steps:

(1) Reconstruct phase space.
(2) With the initial phase point Y_{t_0} as starting point, suppose Y'_{t_0} is the point closest to Y_{t_0}, construct the initial vector V_0, and calculate the length L_{t_0}. Suppose V_0 evolves forward along the locus in a proper period of time τ to get a new vector V_1, calculate the length L'_{t_0}; find $Y'(t_1)$ at $t_1 = t_0 + \tau$, and track and calculate L_{t_1} and L'_{t_1}; repeat the process for m times to get the calculation of maximum Lyapunov exponent λ:

$$\lambda = \frac{1}{N_m} \sum_{k=1}^{M} \ln \frac{L'_{t_k+1}}{L_{t_k}} \tag{2.40}$$

In the formula, N_m stands for the total points of the time series in the m-dimensional space; M for the groups formed by (L_{t_k}, L'_{t_k+1}); L_{t_k} for the distance between the point $X_{t_k} = (x_{t_k}, x_{t_k-1}, \ldots x_{t_k-m+1})$ and the point closest to it; L'_{t_k+1} for

the length of L_{t_k} at $t_k + 1$. When the estimated value of the exponent remains stable with m, the acquired calculation result is $\lambda_{\max}$, the maximum Lyapunov exponent.

5. Defining Step Size for Multistep Prediction

As chaotic behavior is resulted from the presence of strange attractors, its short-term behavior is predictable. However, because of its extreme sensitivity to the initial condition, its long-term behavior cannot be predicted. For a time series without noise, the longer it is, the higher the prediction accuracy is and more slowly the prediction accuracy decreases with the step size. When the time series is long enough, higher prediction accuracy can be achieved even for multistep prediction. For the time series with noise, the prediction accuracy decreases at exponential speed with the increase of step size and the length of the time series can hardly influence the prediction time. But the higher the noise level is, the shorter the prediction time is. Many scholars define the maximum prediction time length of the chaotic series as the reciprocal of the maximum Lyapunov exponent λ, or define the average prediction time ($T_{\max} = 1/k$) according to Kolmogorov entropy. Researches indicate that it is inadvisable to define the maximum prediction time length with the reciprocal of the maximum Lyapunov exponent λ without considering the influence of such factors as specific object, series length, and noise level upon the prediction time. Therefore, in this book, study object and series length are taken into consideration and improved maximum prediction time length is adopted after experimental demonstration:

$$T_{\max} = 1/\lambda + 1 \tag{2.41}$$

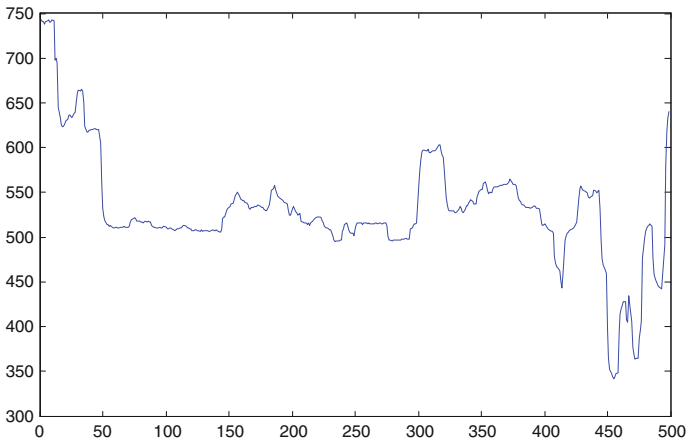


Fig. 2.20 *T*4 temperature

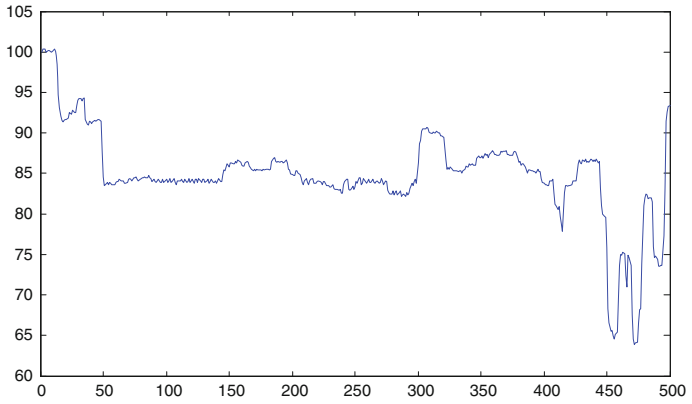


Fig. 2.21 $N1$, low-pressure rotor speed

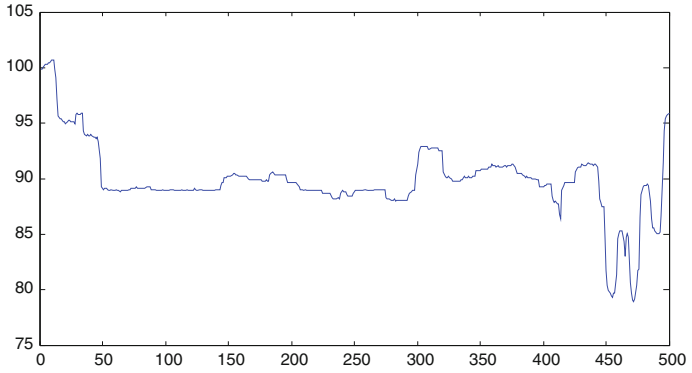


Fig. 2.22 $N2$, high-pressure rotor speed

2.5.2 *Analysis and Verification of Chaotic Property*

To reveal the in-depth change feature of flight data, chaotic analysis theory is used to determine whether the flight data series has chaotic property. At first, define the embedding dimension m and time delay t for phase space reconstruction of flight data $\{x(t), t = 1, 2, \ldots, n\}$; then, use m and t to reconstruct phase space, construct learning sample and calculate the maximum Lyapunov exponent. Here, typical engine state data are adopted, including $T4$ temperature, $N1$ and $N2$ for low-pressure and high-pressure rotor speed, respectively, with series length n as 500, as shown in Figs. 2.20, 2.21, and 2.22, where horizontal coordinates indicate the number of data.

1. Phase Space Reconstruction of Flight Data

As flight data are correlated in various ways, in the analysis of chaotic time series, the first step for chaotic prediction and differentiation of the features of chaotic signals is to reconstruct phase space of the chaotic signals. Therefore, it is necessary to define proper embedding dimension m and delay time τ.

1) Selection of Delay Time

Selection of proper delay time can guarantee the independence and weak correlation of embedding coordinates. One of the relatively simple and practical ways to determine delay time τ is to adopt series autocorrelation function. Make 1–200 s delay for three series with autocorrelation function method. Figures 2.23, 2.24, and 2.25 (the horizontal coordinate axis X refer to delay time(s), the vertical coordinate axis Y refer to autocorrelation function value) show the autocorrelation function curves of delay time τ with temperature $T4$ and high-pressure/low-pressure rotor speed, respectively. The figures show that the autocorrelation function curves heave and set at $s = 0$ with increasing of delay time. In general, the time when the autocorrelation value becomes zero (or approximate to zero) for the first time is selected as delay time to be defined.

As shown in Fig. 2.23, with the autocorrelation functions of $T4$ temperature: $k(1) = 1.0679$, $k(2) = 1.2496$, $k(3) = 2.1963$, and $k(4) = -2.8357$, $k(1)$ is the value when the autocorrelation function of $T4$ temperature is approximate to zero for the first time, and thus the optimal delay time of $T4$ temperature can be determined as $\tau = 1$.

As shown in Fig. 2.24, with the autocorrelation functions of low-pressure speed $N1$: $k(1) = 1.0142$, $k(2) = 1.0917$, $k(3) = 1.2888$, and $k(4) = -5.2480$, $k(1)$ is the value when the autocorrelation function is approximate to zero for the first time, and thus the optimal delay time of $N1$ can be determined as $\tau = 1$.

As shown in Fig. 2.25, with the autocorrelation functions of high-pressure speed $N2$: $k(1) = 1.0202$, $k(2) = 1.0413$, $k(3) = 1.0861$, $k(4) = 1.3471$, $k(5) = 3.7498$, and $k(6) = -1.9008$, $k(1)$ is the value when the autocorrelation function is approximate to zero for the first time, and thus the optimal delay time of $N2$ can be determined as $\tau = 1$.

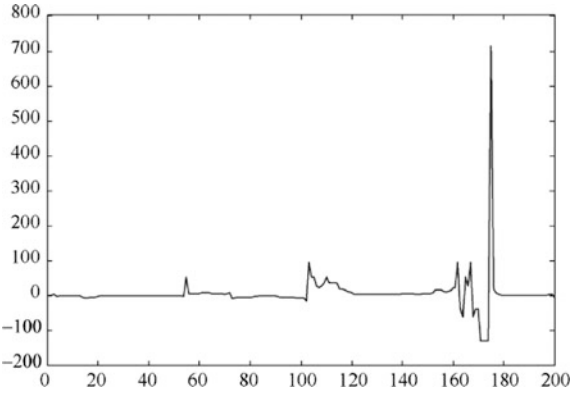


Fig. 2.23 Delay time curve of $T4$ temperature

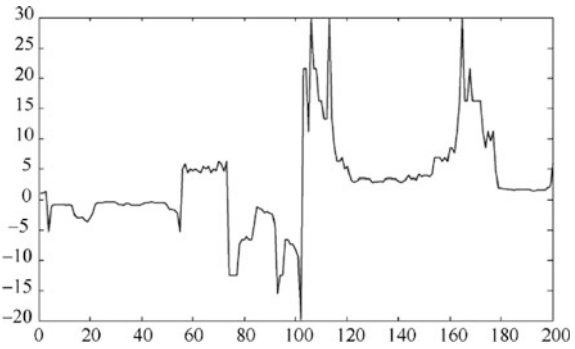


Fig. 2.24 Delay time curve of low-pressure speed $N1$

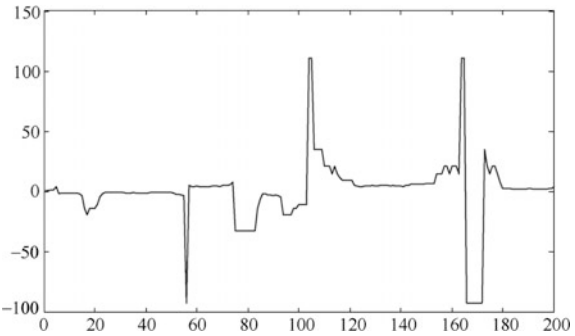


Fig. 2.25 Delay time curve of high-pressure speed $N2$

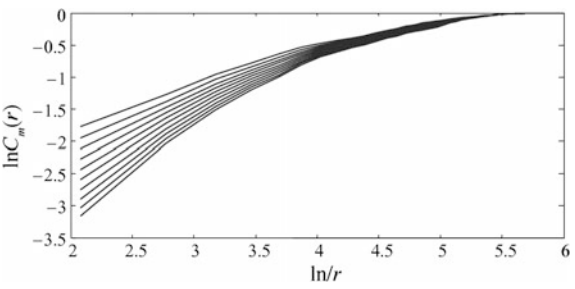


Fig. 2.26 $\ln C_m(r) - \ln r$ curve of $T4$ temperature

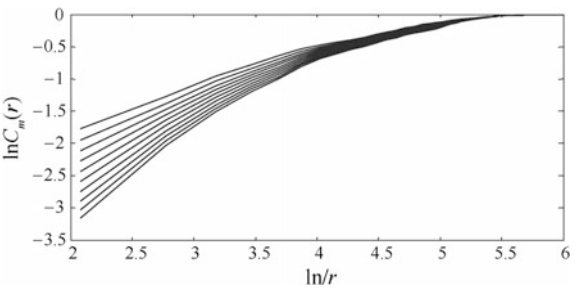


Fig. 2.27 $\ln C_m(r) - \ln r$ curve of low-pressure speed $N1$

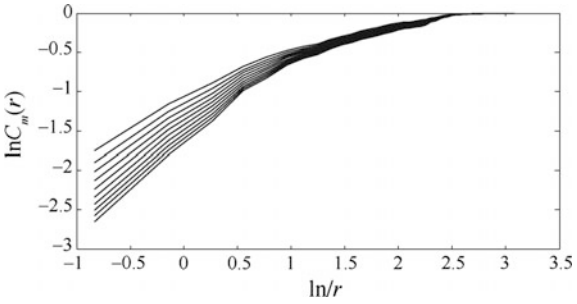


Fig. 2.28 $\ln C_m(r) - \ln r$ curve of high-pressure speed $N2$

2) Selection of Embedding Dimension

Saturated correlation dimension algorithm (briefly known as G-P algorithm) is a common method used to determine embedding dimension. Correlation dimension will converge gradually with the rising of the embedding dimension and eventually result in a convergence value. The dimension in the process of converging is the saturated embedding dimension. According to G-P algorithm, when embedding dimension m is the value from 1 to 10, the $\ln C_m(r) - \ln r$ curves of $T4$ temperature and high-pressure/low-pressure speed are shown in Figs. 2.26, 2.27, and 2.28.

As shown in Figs. 2.26, 2.27, and 2.28, with the rising of embedding dimension m, the line segments in the middle become parallel gradually. Eliminate the line segments with slope as 0 and ∞ and determine the optimum fitting straight line. The slope of the straight line is the correlation dimension. The curves of the correlation dimension D of $T4$ temperature and high-pressure/low-pressure speed changing with the embedding dimension m are shown in Figs. 2.29, 2.30, and 2.31.

As Fig. 2.29 shows, with the rising of the embedding dimension, the correlation dimension of $T4$ temperature with $m = 6$ becomes saturated, and the corresponding correlation dimension $D(6) = 2.6$. As Fig. 2.30 shows, the correlation dimension of low-pressure speed $N1$ becomes saturated when $m = 4$, and the corresponding correlation dimension $D(4) = 1.71$. As Fig. 2.31 shows, the correlation dimension of high-pressure speed $N2$ becomes saturated when $m = 5$, and the corresponding correlation dimension $D(5) = 1.9$. Therefore, 6, 4, and 5 are selected for the embedding dimension of $T4$ temperature and high-pressure/low-pressure speed, respectively.

For selection of saturated correlation dimension, a very large number of samples are required. Smith states that $N_{\min} \geq 42^m$, the minimum sample number is required. Thus, when $m = 3$, $N_{\min} \geq 74088$. This requirement is relatively demanding and hard to be met. Eckman and Ruelle derived $N_{\min} \geq \sqrt{10}^D$, where D stands for the correlation dimension. When $D = 4$, $N_{\min} \geq 100$. This requirement is less demanding. Hong Shizhong and Hong Shiming analyzed the derivation of these two conclusions and found some problems. So, they summarized a new relationship formula, $N_{\min} \geq \sqrt{2}(\sqrt{27.5})^D$. With this method, when $D = 3$, $N_{\min} \geq 204$. Therefore, this method is relatively appropriate. Lin Zhenshan believes that a large

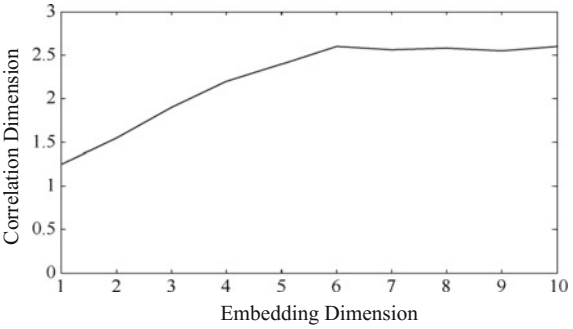


Fig. 2.29 Relationship between the embedding dimension and the correlation dimension of $T4$

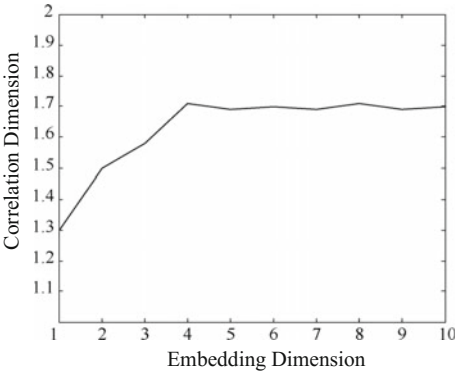


Fig. 2.30 Relationship between the embedding dimension and the correlation dimension of $N1$

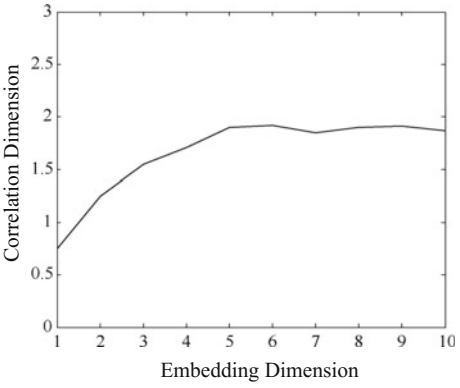


Fig. 2.31 Relationship between the embedding dimension and the correlation dimension of $N2$

amount of data is unnecessary for phase space reconstruction aimed at prediction and only enough data are needed for experiments to realize $N_{\min} \geq 260$. The data adopted in this book are 500, which meets the requirement.

2. Qualitative Analysis of Chaotic Property

Qualitative analysis of chaotic property is to roughly analyze the property of measured series in time or frequency domain. The common methods include phase diagram method, power spectrum, etc.

Phase diagram can describe the changes of the system state during the whole time period and reflect the spatial structure of system attractors. The phase space locus of a chaotic system usually manifests as iterative, aperiodic, and never intersecting movement resulted from its stretching and folding in limited space. It differs from irregular and random movement and it is not a repeated movement of a periodic function.

Power spectrum can be used to differentiate the regular (fixed point, periodic, quasi-periodic) and irregular forms (chaos, noise) of the time series. The power spectrum for periodic movement is discrete, only including basic frequency and its harmonic wave or frequency division. The power spectrum for random white noise and chaos is continuous while the power spectrum for chaotic series is continuous with broad peak. Power spectrum method is adopted to analyze the above data and Figs. 2.32, 2.33, and 2.34 show the power spectrum of *T*4 temperature and high-pressure/low-pressure speed, respectively. As the figures show, the curves are all continuous with broad peak. Therefore, the three series are identified initially as chaotic series.

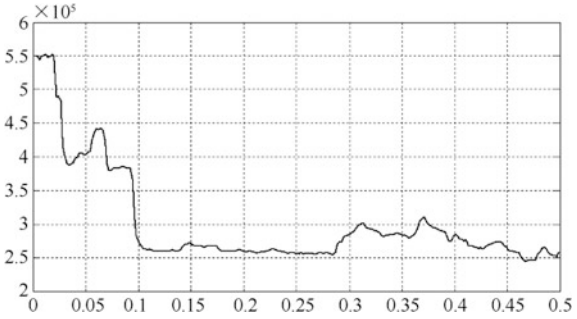


Fig. 2.32 Power spectrum of *T*4 temperature

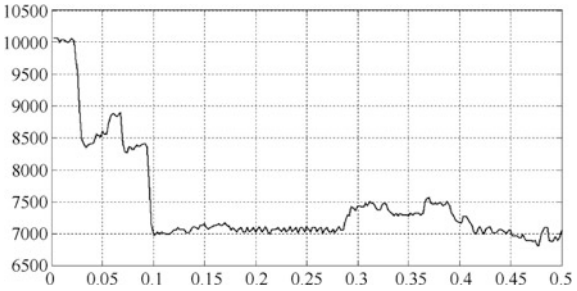


Fig. 2.33 Power spectrum of low-pressure speed *N*1

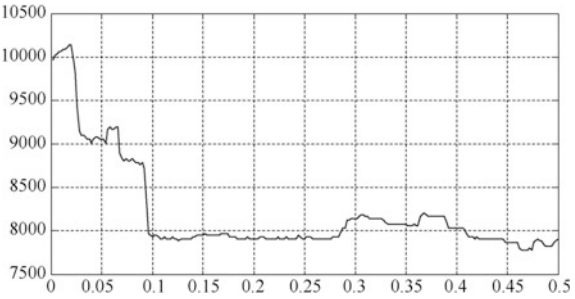


Fig. 2.34 Power spectrum of high-pressure speed *N*2

Table 2.3 Maximum Lyapunov exponent for *T*4 temperature

Embedding dimension of *T*4	2	3	4	5	6	7	8
Lyapunov exponent for *T*4 temperature	0.1156	0.1607	0.1231	0.1087	0.1008	0.1411	0.0584

Table 2.4 Maximum Lyapunov exponent for low-pressure speed *N*1

Embedding dimension of low-pressure speed *N*1	2	3	4	5	6	7	8
Lyapunov exponent for low-pressure SPEED *N*1	0.1831	0.1136	0.1178	0.1012	0.0876	0.0994	0.1086

Table 2.5 Maximum Lyapunov exponent for high-pressure speed *N*2

Embedding dimension of high-pressure speed *N*2	2	3	4	5	6	7	8
Lyapunov exponent for high-pressure speed *N*2	0.0514	0.0734	0.0675	0.0408	0.0345	0.0481	0.0326

3. Quantitative Analysis of Chaotic Property

From the above-mentioned qualitative analysis, rough knowledge about the chaotic property of engine data can be acquired. But it is not enough to prove that

the series is chaotic. Therefore, it is necessary to use quantitative analysis to verify it. Methods for quantitative analysis include maximum Lyapunov exponent, Kolmogorov Entropy, etc.

Wolf method is adopted to estimate λ, the maximum Lyapunov exponent in the flight data series. To testify the stability of the algorithm and study the influence of different embedding dimensions m upon λ, calculate λ, the maximum Lyapunov exponent with embedding dimension m from 2 to 8, respectively. Tables 2.3, 2.4, and 2.5 describe the results, which show that with embedding dimension m from 2 to 8, the maximum Pyapunov exponents for $T4$ temperature and high-pressure/low-pressure speed are different, but λ is greater than zero. For $T4$ temperature, when the embedding dimension $m = 3$, the maximum Lyapunov exponent $\lambda = 0.1607$. For low-pressure speed $N1$, when the embedding dimension $m = 2$, the maximum Lyapunov exponent $\lambda = 0.1831$. For high-pressure speed $N2$, when the embedding dimension $m = 3$, the maximum Lyapunov exponent $\lambda = 0.0734$. The conclusion is that data for $T4$ temperature and high-pressure/low-pressure speed have obviously chaotic property. Therefore, the methods involved can provide a feasible thought and approach to identify whether other types of flight data have chaotic property.

http://www.springer.com/978-3-662-53428-1

Time Series Analysis Methods and Applications for Flight Data

Zhang, J.; Zhang, P.

2017, X, 240 p. 161 illus., 35 illus. in color., Hardcover
ISBN: 978-3-662-53428-1