Chapter 3
Evolutionary Algorithms in SVR's Parameter Determination

As mentio.ned in Chap. 2, the traditional determination of three parameters does not guarantee to improve forecasting accuracy level, because of its inability to set up more suitable initial values of parameters σ, C, and ε in the initial step and inability to simultaneously consider the interaction effects among three parameters to efficiently find out the near optimal solution for large scale data set. Therefore, it is feasible to employ evolutionary algorithms to conduct intelligent searching around the solution range to determine suitable parameter combination by minimizing the objective function describing the structural risk of a SVR model. This chapter introduces several representative evolutionary algorithms used in a SVR forecasting model to determine suitable parameter combination to receive improved forecasting accuracy level.

3.1 Data Set and Forecasting Comparison Statistical Tests

3.1.1 Data Set

To be based on the same comparison conditions, this book uses historical monthly electric load data of Northeast China to compare the forecasting performance among the proposed SVR-based models hybridizing with seven evolutionary algorithms, ARIMA, and TF-ε-SVR-SA models proposed by Wang et al. [1]. Table 3.1 lists the data set used in this chapter. Figure 3.1 illustrates the tendency of this data set. Totally, there are 64 data (from January 2004 to April 2009) of Northeastern China monthly electric load. However, based on Wang et al.'s [1] support vectors computation, only 53 months data (from December 2004 to April 2009) are suggested. Therefore, for the same comparison condition, the employed data are divided into three data sets: the training data set (32 months, December

W.-C. Hong, *Intelligent Energy Demand Forecasting*, Lecture Notes in Energy 10,
DOI 10.1007/978-1-4471-4968-2_3,

Table 3.1 Monthly electric load in Northeastern China (from Jan. 2004 to Apr. 2009) (unit, hundred million kWh)

Time	Electric load	Time	Electric load	Time	Electric load
Jan. 2004	129.08	Nov. 2005	150.84	Sep. 2007	175.41
Feb. 2004	127.24	Dec. 2005	165.27	Oct. 2007	179.64
Mar. 2004	136.95	Jan. 2006	155.31	Nov. 2007	188.89
Apr. 2004	125.34	Feb. 2006	138.5	Dec. 2007	197.62
May 2004	126.86	Mar. 2006	133.27	Jan. 2008	200.35
Jun. 2004	129.34	Apr. 2006	151.41	Feb. 2008	169.24
Jul. 2004	131.91	May 2006	155.63	Mar. 2008	196.97
Aug. 2004	136.22	Jun. 2006	155.7	Apr. 2008	186.15
Sep. 2004	131.56	Jul. 2006	162.98	May 2008	188.485
Oct. 2004	134.62	Aug. 2006	163.41	Jun. 2008	190.82
Nov. 2004	144.62	Sep. 2006	157.57	Jul. 2008	196.53
Dec. 2004	154.62	Oct. 2006	160.15	Aug. 2008	197.67
Jan. 2005	151.48	Nov. 2006	168.13	Sep. 2008	183.77
Feb. 2005	126.74	Dec. 2006	180.71	Oct. 2008	181.07
Mar. 2005	148.57	Jan. 2007	179.94	Nov. 2008	180.56
Apr. 2005	136.6	Feb. 2007	147.29	Dec. 2008	189.03
May 2005	138.83	Mar. 2007	172.45	Jan. 2009	182.07
Jun. 2005	136.6	Apr. 2007	169.98	Feb. 2009	167.35
Jul. 2005	146.21	May 2007	173.21	Mar. 2009	189.3
Aug. 2005	146.09	Jun. 2007	177.43	Apr. 2009	175.84
Sep. 2005	140.04	Jul. 2007	184.29		
Oct. 2005	142.02	Aug. 2007	183.53		

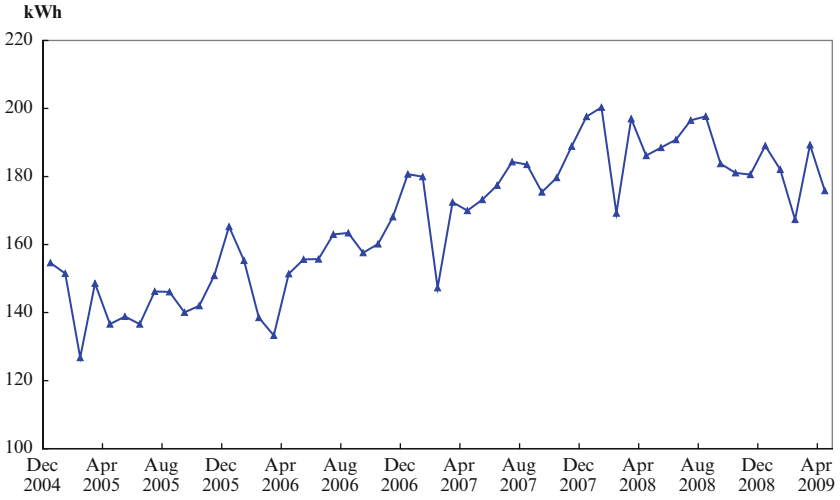

Fig. 3.1 Illustration of the monthly electric load in Northeastern China (from Jan. 2004 to Apr. 2009)

Table 3.2 Training, validation, and testing data sets of the proposed models

Data sets	SVR-based models	TF-ε-SVR-SA model [1]
Training data	Dec. 2004–Jul. 2007	Dec. 2004–Sep. 2008
Validation data	Aug. 2007–Sep. 2008	
Testing data	Oct. 2008–Apr. 2009	Oct. 2008–Apr. 2009

2004 to July 2007), the validation data set (14 months, August 2007 to September 2008), and the testing data set (7 months, from October 2008 to April 2009), as shown in Table 3.2.

3.1.2 Forecasting Comparison Statistical Tests

3.1.2.1 Wilcoxon Signed-Rank Test

Wilcoxon signed-rank test is used to detect the significance of a difference in central tendency of two data series when the size of two data series is equal [2]. The statistic W is represented as Eq. (3.1):

$$W = \min\{S^+, S^-\}, \tag{3.1}$$

where

$$S^+ = \sum_{i=1}^{n} I^+(d_i), \tag{3.2}$$

$$S^- = \sum_{i=1}^{n} I^-(d_i), \tag{3.3}$$

$$I^+(d_i) = \begin{cases} 1 & \text{if } d_i > 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3.4}$$

$$I^-(d_i) = \begin{cases} 1 & \text{if } d_i < 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3.5}$$

and

$$d_i = (\text{data series I})_i - (\text{data series II})_i. \tag{3.6}$$

3.1.2.2 Asymptotic Test

Diebold and Mariano [3] suggest the asymptotic test (S_1), which is applicable because one or more assumptions are violated to be suitable for the simple F test, the Morgan–Granger–Newbold test [4, 5], and the Meese–Rogoff test [6]. Therefore, the S_1 statistic, as Eq. (3.7), is employed in this study:

$$S_1 = \frac{\bar{d}}{\sqrt{\frac{2\pi\hat{f}_d(0)}{T}}}, \tag{3.7}$$

where d_i is the loss-differential series of two compared electric forecasting models, as shown in Eq. (3.8),

$$d_i = e_{1i}^2 - e_{2i}^2, \tag{3.8}$$

where e_1 and e_2 denoted the errors of the two compared electric load forecasting models, respectively.

$2\pi\hat{f}_d(0)$ is the weighted sum of the available sample autocovariances as Eq. (3.9):

$$2\pi\hat{f}_d(0) = \sum_{\tau=-(T-1)}^{T-1} 1 * \left(\frac{\tau}{S(T)}\right)\hat{\gamma}_d(T), \tag{3.9}$$

where T is the sample size, $\hat{\gamma}_d(T)$ is defined as Eq. (3.10),

$$\hat{\gamma}_d(T) = \frac{1}{T}\sum_{t=|\tau|+1}^{T}(d_t - \bar{d})(d_{t-|\tau|} - \bar{d}), \tag{3.10}$$

and $1 * \left(\frac{\tau}{S(T)}\right)$ is the lag window, defined as Eq. (3.11),

$$1 * \left(\frac{\tau}{S(T)}\right) = \begin{cases} 1 & \text{if } \left|\frac{\tau}{S(T)}\right| \le 1 \\ 0 & \text{otherwise} \end{cases}. \tag{3.11}$$

Obviously, $S(T) = k$, where k denotes as the number of forecasting ahead, the forecasts in this book are one-step-ahead; thus, k is set as 1.

The test will be performed at the 0.05 and 0.10 significance levels in two-tail tests (with normal distribution) under the null hypothesis of equal forecast accuracy for the two compared electric load forecasting models.

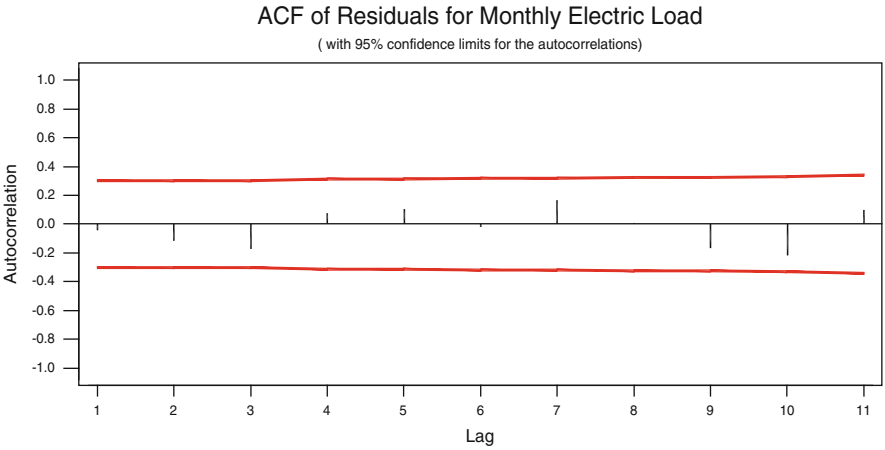

Fig. 3.2 Estimated residual ACF

3.2 Modeling and Forecasting Results of Alternative Models

3.2.1 ARIMA Model

For ARIMA models, the statistical package identified the most suitable model for the training data as ARIMA(1,1,1) with constant term. The ARIMA(1,1,1) model can be expressed as Eq. (3.12):

$$(1 - 0.0641B^1)\nabla y_t = 1.2652 + (1 + 0.9715B^1)\varepsilon_t. \tag{3.12}$$

After determining the suitable parameters of the ARIMA model, it is important to examine how closely the proposed model fits a given time series. The autocorrelation function (ACF) is calculated to verify the parameters. Figure 3.2 plots the estimated residual ACF and indicates that the residuals are not autocorrelated. PACF, the partial autocorrelation function, displayed in Fig. 3.3, is also used to check the residuals and indicates that the residuals are not correlated. The forecasting results are shown in the third column of Table 3.3.

3.2.2 Holt–Winters Model

For the Holt–Winters (HW) method, by Minitab 14 statistic software, the α value and β value are determined as 0.5618 and 0.0472, respectively. The forecasting results are shown in the fourth column of Table 3.3.

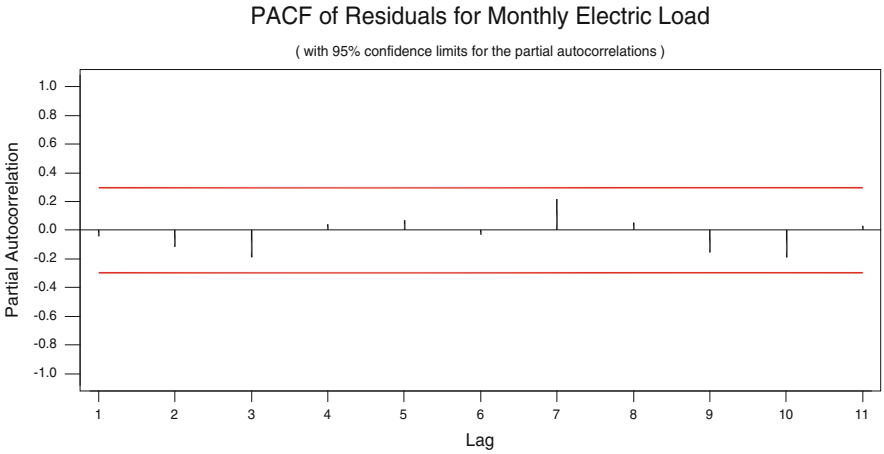

Fig. 3.3 Estimated residual PACF

Table 3.3 Forecasting results of ARIMA, HW, GRNN, and BPNN models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA (1,1,1)	HW (0.5618, 0.0472)	GRNN ($\sigma = 3.33$)	BPNN
Oct. 2008	181.07	192.932	191.049	191.131	172.084
Nov. 2008	180.56	191.127	192.015	187.827	172.597
Dec. 2008	189.03	189.916	192.981	184.999	176.614
Jan. 2009	182.07	191.995	193.947	185.613	177.641
Feb. 2009	167.35	189.940	194.913	184.397	180.343
Mar. 2009	189.30	183.988	195.879	178.988	183.830
Apr. 2009	175.84	189.348	196.846	181.395	187.104
MAPE (%)		6.044	7.480	4.636	5.062

3.2.3 GRNN Model

For the GRNN model, Fig. 3.4 shows the MAPE values of the GRNN with various σ. Clearly, while σ exceeds over 3.33, the value of MAPE will subsequently increase. Therefore, the limit of σ is 3.33. In this book, the value of σ is set at 0.04. The forecasting results are shown in the fifth column of Table 3.3.

3.2.4 BPNN Model

For the BPNN model, Matlab 6.5 computing software is employed to implement the forecasting procedure. The number of nodes in the hidden layer is used as a validation parameter of the BPNN model. The most suitable number of hidden

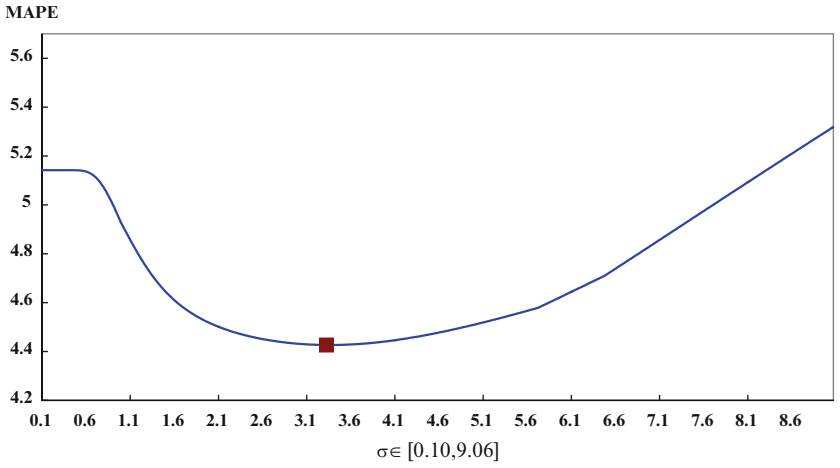

Fig. 3.4 MAPE with various σ values of GRNN model

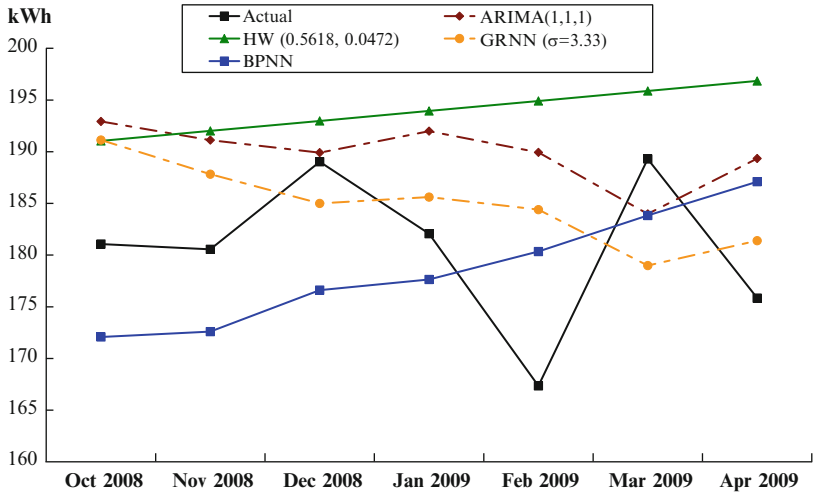

Fig. 3.5 Forecasting results of ARIMA, HW, GRNN, and BPNN models

nodes of the BPNN model is set as 3. The forecasting results are shown in the last column of Table 3.3.

Figure 3.5 is provided to illustrate the forecasting accuracy among different models. Obviously, these four models are not fitting well particularly for HW and BPNN models. Therefore, it is desired to look for more robust technology to overcome these shortcomings from the traditional models and ANN-based models. In the following sections, ARIMA(1,1,1) and GRNN($\sigma = 3.33$) are selected to be compared with SVR-based models.

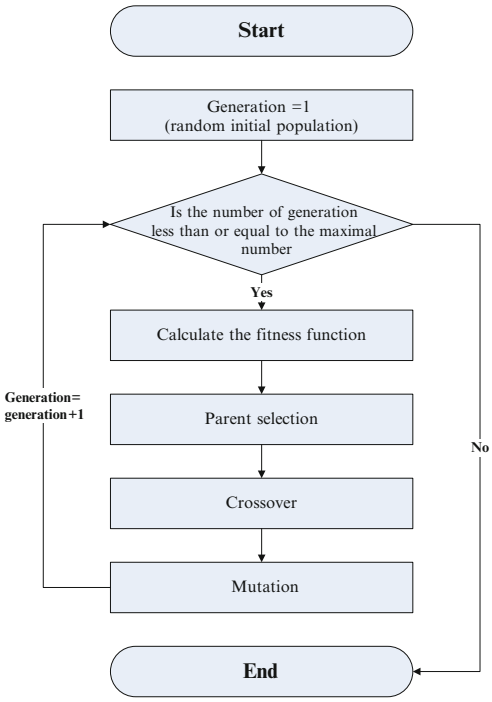

Fig. 3.6 The architecture of the genetic algorithm

3.3 Genetic Algorithm in SVR's Parameter Determination

3.3.1 Operation Procedure of GA

Proposed by Holland [7], genetic algorithm (GA) is an organized random search technique by imitating the biological evolution process. Such algorithm is based on the principle of the survival of the fittest, which attempts to retain genetic information from generation to generation. GA is also auto-adaptive stochastic search technique [7]; it generates new individuals with selection, crossover, and mutation operators. GA starts with a coding of the parameter set of all types of objective functions; thus, it has the ability to solve those traditional algorithms that are not easy to solve. The major advantages of GA are the capabilities for finding optimal or near optimal solutions with relatively modest computational requirements. Figure 3.6 depicts the operation of a GA, which is described below.

Step 1: Initialization. Generate randomly an initial population of chromosomes. The three free parameters, σ, C, and ε, are encoded in a binary format and represented by a chromosome.

Step 2: Evaluating fitness. Evaluate the fitness of each chromosome. In this book, a negative mean absolute percentage error (−MAPE) is used as the fitness function. The MAPE is as Eq. (3.13):

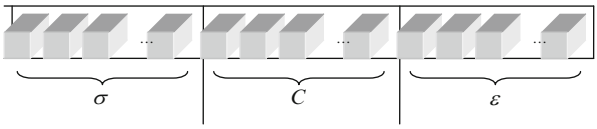

Fig. 3.7 The binary encoding of a chromosome

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\text{a}_i - \text{f}_i}{\text{a}_i}\right| \times 100\,\%, \tag{3.13}$$

where a_i and f_i represent the actual electric load and forecast electric load, and N is the number of forecasting periods.

Step 3: Selection. Based on fitness functions, chromosomes with higher fitness values are more likely to yield offspring in the next generation. The roulette wheel selection principle [7] is applied to choose chromosomes for reproduction.

Step 4: Crossover and mutation. Create new offspring by crossover and mutation operations. Mutations are performed randomly by converting a "1" bit into a "0" bit or a "0" bit into a "1" bit. The single-point-crossover principle is employed. Segments of paired chromosomes between two determined breakpoints are swapped. The rates of crossover and mutation are probabilistically determined. In this book, the probabilities of crossover and mutation are set to 0.5 and 0.1, respectively.

Step 5: Next generation. Form a population for the next generation.

Step 6: Stop condition. If the number of generation is equal to a given scale, then the best chromosomes are presented as a solution; otherwise, go back to step 2.

GA is used to yield a smaller MAPE by searching for better combinations of three parameters in SVR. In this book, binary encoding is specified in GA. Three free parameters, σ, C and ε, are represented by a chromosome that consists of three genes in the form of binary numbers (Fig. 3.7). The size of the population is set to 200 herein. Each gene contains 40 bits. If each gene contains 40 bits, for example, then a chromosome contains 120 bits. More bits in a gene correspond to a finer partition of the searched space. Parent selection is a procedure in which two chromosomes from the parent population are chosen according to the fitness functions. Fitter chromosomes are more likely to generate offspring to the next generation. For simplicity, suppose a gene has four bits. A chromosome contains 12 bits (Fig. 3.8). Before crossover is performed, the values of the three parameters in #1 parent are 1.5, 1.25, and 0.34375. For #2 parent, the three values are 0.625, 8.75, and 0.15625. After crossover, for #1 offspring, the three values are 1.625, 3.75, and 0.40625. For #2 offspring, the three values are 0.5, 6.25, and 0.09375. Mutations are performed randomly by converting a "1" bit into a "0" bit or a "0" bit into a "1" bit. The rates of crossover and mutation are probabilistically determined. In this study, the probabilities of crossover and mutation are set to 0.5 and 0.1, respectively.

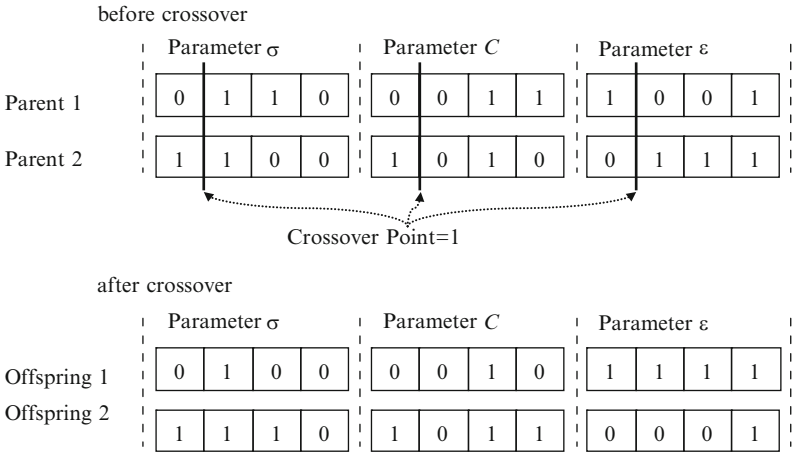

Fig. 3.8 A simplified example of parameter representation

3.3.2 GA for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with GA), namely, SVRGA model.

In the training stage, the rolling-based forecasting procedure (see Fig. 3.9) is conducted, which is dividing training data into two subsets, namely, fed-in subset (e.g., 25 load data) and fed-out subset (7 load data), respectively. Firstly, the primary 25 load data of fed-in subset are feeding into the proposed model (the structural risk minimization principle is employed to minimize the training error) and then obtain one-step-ahead forecasting load, namely, the 26th forecasting load. Secondly, the next 25 load data, including 24 of the fed-in subset data (from 2nd to 25th) pulsing the 26th data in the fed-out subset, are similarly again fed into the proposed model (the structural risk minimization principle is also employed to minimize the training error) and then obtain one-step-ahead forecasting load, namely, the 27th forecasting load. Repeat the rolling-based forecasting procedure till the 32nd forecasting load is obtained. Meanwhile, training error in this training stage is also obtained.

For the first training rolling, 25 months (from Dec. 2004 to Dec. 2006), electric load data are feeding into SVRGA to compute the forecasting electric load of Jan. 2007, then, for the second training rolling, the next 25 months (from Jan. 2005 to Jan. 2007), electric load data are similarly feeding into SVRGA to compute the forecasting electric load of Feb. 2007, and repeat previous procedures to compute the forecasting electric load of Mar. 2007,. . ., Jul. 2007, respectively. Finally, compute the MAPE of training stage between the forecasting electric load (from Jan. 2007 to Jul. 2007) and the original load (from Jan. 2007 to Jul. 2007, namely, fed-out subset).

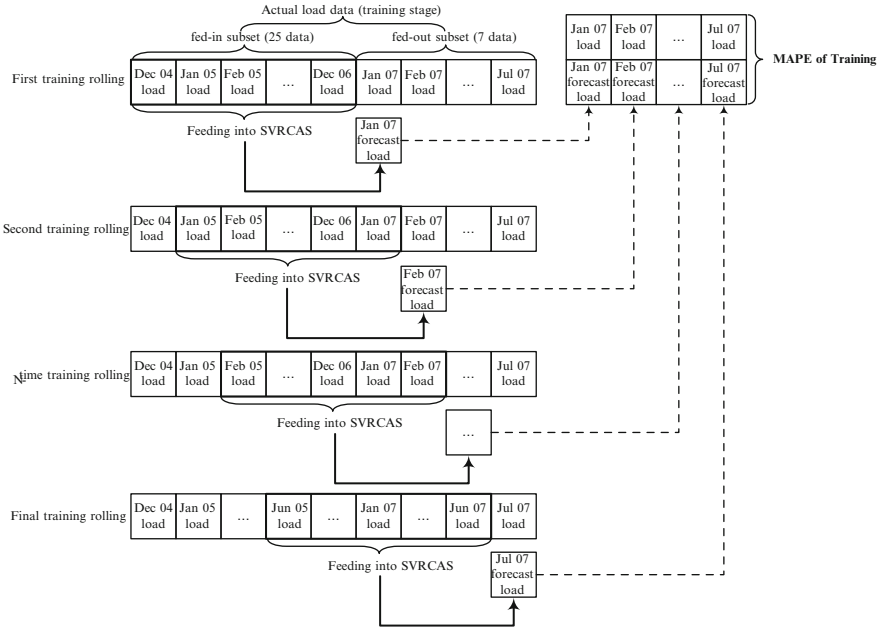

Fig. 3.9 The rolling-base forecasting procedure (training stage)

Table 3.4 Parameter determination of SVRGA model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	76.84	297.20	0.4298	4.354
10	4.46	143.54	1.6705	3.763
15	4.67	70.23	3.9921	3.719
20	233.56	2,911.70	11.2340	3.974
25	**686.16**	**5,048.40**	**19.3170**	**3.676**

While training error improvement occurs, the three kernel parameters, σ, C, and ε, of the SVRGA model adjusted by GA are employed to calculate the validation error. Then, the adjusted parameters with minimum validation error are selected as the most appropriate parameters. Finally, a one-step-ahead policy is employed to forecast electric load. Note that the testing data set is not used for modeling but for examining the accuracy of the forecasting model. The forecasting results and the suitable parameters for the SVRGA model are illustrated in Table 3.4, in which it is indicated that these two models all perform the best when 25 fed-in data are used.

Table 3.5 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN($\sigma = 3.33$), TF-ε-SVR-SA, and SVRGA. The MAPE values are calculated to compare fairly the proposed models with other alternative models. The proposed SVRGA model has smaller MAPE values than

Table 3.5 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, and SVRGA models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA(1,1,1)	GRNN($\sigma = 3.33$)	TF-ε-SVR-SA	SVRGA
Oct. 2008	181.07	192.932	191.131	184.504	178.326
Nov. 2008	180.56	191.127	187.827	190.361	178.355
Dec. 2008	189.03	189.916	184.999	202.980	178.355
Jan. 2009	182.07	191.995	185.613	195.753	178.356
Feb. 2009	167.35	189.940	184.397	167.580	178.357
Mar. 2009	189.30	183.988	178.988	185.936	178.358
Apr. 2009	175.84	189.348	181.395	180.165	181.033
MAPE (%)		6.044	4.636	3.799	3.676

Table 3.6 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRGA vs. ARIMA(1,1,1)	0[a]	0[a]
SVRGA vs. GRNN($\sigma = 3.33$)	0[a]	0[a]
SVRGA vs. TF-ε-SVR-SA	2[a]	2[a]

[a]Denotes that SVRGA model significantly outperforms other alternative models

Table 3.7 Asymptotic test

	Asymptotic (S_1) test	
Compared models	$\alpha = 0.05$	$\alpha = 0.10$
SVRGA vs. ARIMA(1,1,1)	H_0: $e_1 = e_2$ $S_1 = -11.546$; p=0.000 (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -11.546$; $p = 0.000$ (reject H_0)
SVRGA vs. GRNN($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -2.100$; $p = 0.0179$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.100$; $p = 0.0179$ (reject H_0)
SVRGA vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -2.344$; $p = 0.00954$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.344$; $p = 0.00954$ (reject H_0)

ARIMA, GRNN, and TF-ε-SVR-SA models. Furthermore, to verify the significance of accuracy improvement of SVRGA model comparing with ARIMA(1,1,1), GRNN ($\sigma = 3.33$), and TF-ε-SVR-SA models, the statistical test, namely, Wilcoxon signed-rank test, is conducted at the 0.025 and 0.05 significance levels in one-tail tests.

The test results are shown in Tables 3.6 and 3.7, respectively. Clearly, the SVRGA model is significantly superior to ARIMA(1,1,1), GRNN($\sigma = 3.33$), and TF-ε-SVR-SA. Figure 3.10 is provided to illustrate the forecasting accuracy among different models.

The superior performance of the SVRGA model has several causes: first, for a SVR-based model, it has nonlinear mapping capabilities and can more easily

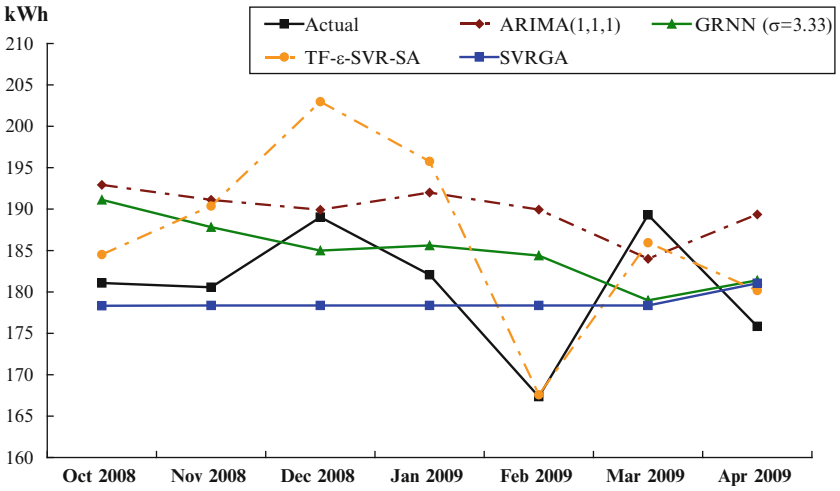

Fig. 3.10 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, and SVRGA models

capture electric load data patterns than can the ARIMA and GRNN models. Second, the parameter selection in a SVR model significantly influences their forecasting performance. Improper determining of these three parameters will cause either over-fitting or under-fitting of a SVR model. In this section, the GA can determine suitable free parameters for forecasting electric load. Finally, the SVR-based model performs structural risk minimization rather than minimizing the training errors. Minimizing the upper bound on the generalization error improves the generalization performance compared to the ARIMA and GRNN models. However, it is clear that SVRGA is not fitting the actual electric loads very well even it has smaller MAPE values and passed the Wilcoxon test than other alternatives. Therefore, it still requires hybridizing other novel techniques to improve this shortcoming.

3.4 Simulated Annealing Algorithm in SVR's Parameter Determination

3.4.1 Operation Procedure of SA Algorithm

The simulated annealing (SA) algorithm is an optimization technique, analogous to the annealing process of material physics. Boltzmann [8] pointed out if the system is in thermal equilibrium at a temperature T, then the probability $P_T(s)$ of the system being in a given state s is given by the Boltzmann distribution, shown as Eq. (3.14):

$$P_T(s) = \frac{\exp(-E(s)/kT)}{\sum_{w \in S} \exp(-E(w)/kT)}, \tag{3.14}$$

where $E(s)$ denotes the energy of state s (the state is defined as the SVR forecasting error in each iteration here), k represents the Boltzmann constant, and S is the set of all possible states. However, Eq. (3.14) does not contain information on how a fluid reaches thermal equilibrium at a given temperature. Metropolis et al. [9] develop an algorithm that simulates the process of Boltzmann. The Metropolis algorithm is summarized as follows. When the system is in original state s_{old} with energy $E(s_{\text{old}})$, a randomly selected atom is perturbed, resulting in a state s_{new} with energy $E(s_{\text{new}})$. This new state is either accepted or rejected depending on the Metropolis criterion: if $E(s_{\text{new}}) \leq E(s_{\text{old}})$, then the new state is automatically accepted; in contrast, if $E(s_{\text{new}}) > E(s_{\text{old}})$, then the probability of accepting the new state is given by the following probability function, Eq. (3.15):

$$P(\text{accept } s_{\text{new}}) = \exp\left(-\frac{E(s_{\text{old}}) - E(s_{\text{new}})}{kT}\right). \tag{3.15}$$

According to the studies of Boltzmann and Metropolis, Kirkpatrick et al. [10] claim that the Metropolis approach is conducted for each temperature on the annealing schedule until thermal equilibrium is reached. Additionally, a prerequisite for applying SA is that a given set of the multiple variables defines a unique system state, for which the objective function can be calculated. The procedure of SA algorithm is described as follows and the flowchart is shown as Fig. 3.11.

Step 1: Initialization. Set upper bounds of the three SVR parameters, σ, C, and ε. Then, generate and feed the initial values of the three parameters into the SVR model. The forecasting error is defined as the system state (E). Here, the initial state (E_0) is obtained.

Step 2: Provisional state. Make a random move to change the existing system state to a provisional state. Another set of three positive parameters is generated in this stage.

Step 3: Acceptance tests. The following equation is employed to determine the acceptance or rejection of the provisional state [9]:

$$\begin{cases} \text{Accept the provisional state,} & \text{if } E(s_{\text{new}}) > E(s_{\text{old}}), \\ \quad \text{and } p < P(\text{accept } s_{\text{new}}),\ 0 \leq p \leq 1 & \\ \text{Accept the provisional state,} & \text{if } E(s_{\text{new}}) \leq E(s_{\text{old}}) \\ \text{Reject the provisional state,} & \text{otherwise.} \end{cases} \tag{3.16}$$

In Eq. (3.16), the p is a random number for determining the acceptance of the provisional state. If the provisional state is accepted, then set the provisional state as the current state.

Step 4: Incumbent solutions. If the provisional state is not accepted, then return to step 2. Furthermore, if the current state is not superior to the system state, then repeat steps 2 and 3 until the current state is superior to the system state, and set the current state as the new system state. Previous studies [10, 11] indicate that the

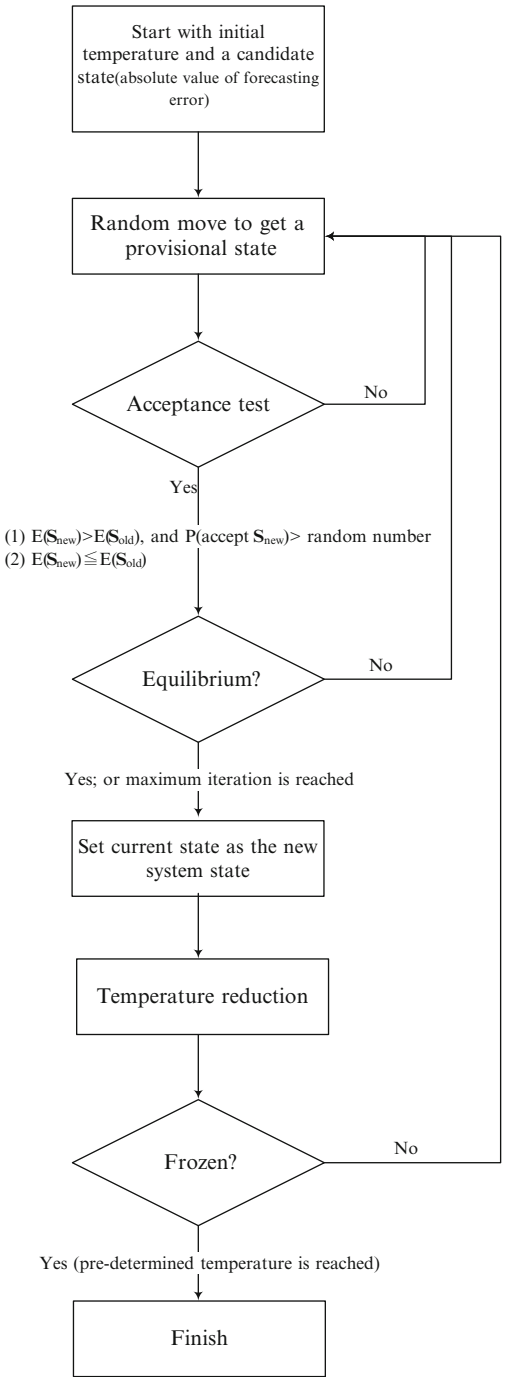

Fig. 3.11 The architecture of the SA algorithm

Table 3.8 Parameter determination of SVRSA model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	464.06	399.70	0.6891	4.289
10	3.72	176.14	0.6089	4.161
15	3.53	165.38	7.3935	3.941
20	3.02	1,336.70	9.8374	3.871
25	**94.998**	**9,435.20**	**12.6570**	**3.801**

maximum number of loops (N_{sa}) is $100d$ to avoid infinitely repeated loops, where d denotes the problem dimension. In this investigation, three parameters (σ, C, and ε) are used to determine the system states. Therefore, N_{sa} is set to be 300.

Step 5: Temperature reduction. After the new system state is obtained, reduce the temperature. The new temperature reduction is obtained by the Eq. (3.17),

$$\text{New temperature} = \text{Current temperature} \times \rho, \tag{3.17}$$

where $0 < \rho < 1$. The ρ is set to be 0.9 in this book [12]. If the predetermined temperature is reached, then stop the algorithm and the latest state is an approximate optimal solution. Otherwise, go to step 2.

Similarly, the value of the mean absolute percent error (MAPE), shown as Eq. (3.13), also serves as the criterion for identifying suitable parameters for use in the SVRSA model. The SA algorithm is used to seek a better combination of the three parameters in a SVR model, so that a smaller MAPE is obtained in each iteration.

3.4.2 SA Algorithm for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with SA), namely, SVRSA model.

Similarly, in the training stage, the rolling-based forecasting procedure is also employed to obtain the forecasting load and receive training error in the training stage. Then, if training error improvement occurs, the three kernel parameters, σ, C, and ε, of the SVRSA model adjusted by SA algorithm are employed to calculate the validation error. The adjusted parameters with minimum validation error are also selected as the most appropriate parameters. The forecasting results and the suitable parameters for the SVRSA model are illustrated in Table 3.8, in which it is also indicated that these two models all perform the best when 25 fed-in data are used.

Table 3.9 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN($\sigma = 3.33$), TF-ε-SVR-SA, and SVRSA. The MAPE values are calculated to compare fairly the proposed models with other alternative models. The proposed SVRSA model only has smaller MAPE values than ARIMA and GRNN models, but TF-ε-SVR-SA model. Furthermore, to verify the

Table 3.9 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, and SVRSA models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA(1,1,1)	GRNN($\sigma = 3.33$)	TF-ε-SVR-SA	SVRSA
Oct. 2008	181.07	192.932	191.131	184.504	184.584
Nov. 2008	180.56	191.127	187.827	190.361	185.412
Dec. 2008	189.03	189.916	184.999	202.980	185.557
Jan. 2009	182.07	191.995	185.613	195.753	185.593
Feb. 2009	167.35	189.940	184.397	167.580	185.737
Mar. 2009	189.30	183.988	178.988	185.936	184.835
Apr. 2009	175.84	189.348	181.395	180.165	184.390
MAPE (%)		6.044	4.636	3.799	3.801

Table 3.10 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRSA vs. ARIMA(1,1,1)	1[a]	1[a]
SVRSA vs. GRNN($\sigma = 3.33$)	2[a]	2[a]
SVRSA vs. TF-ε-SVR-SA	3	3[b]

[a]Denotes that SVRSA model significantly outperforms other alternative models
[b]Denotes that SVRSA model is significantly outperformed by other alternative models

Table 3.11 Asymptotic test

	Asymptotic (S_1) test	
Compared models	$\alpha = 0.05$	$\alpha = 0.10$
SVRSA vs. ARIMA(1,1,1)	H_0: $e_1 = e_2$ $S_1 = -9.790$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -9.790$; $p = 0.000$ (reject H_0)
SVRSA vs. GRNN($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -1.210$; $p = 0.1131$ (not reject H_0)	H_0: $e_1 = e_2$ $S_1 = -1.210$; $p = 0.1131$ (not reject H_0)
SVRSA vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -0.969$; $p = 0.1663$ (not reject H_0)	H_0: $e_1 = e_2$ $S_1 = -0.969$; $p = 0.1663$ (not reject H_0)

significance of accuracy improvement of SVRSA model comparing with ARIMA (1,1,1), GRNN($\sigma = 3.33$), and TF-ε-SVR-SA models, the Wilcoxon signed-rank test and asymptotic test, as mentioned, are also conducted. The test results are shown in Tables 3.10 and 3.11, respectively. Clearly, the SVRSA model is significantly superior to ARIMA(1,1,1) model and minor significantly superior to GRNN ($\sigma = 3.33$) model (only receives significance with both levels in Wilcoxon test, but all fails with both levels in asymptotic test). And, based on Tables 3.9, 3.10, and 3.11, TF-ε-SVR- SA model has smaller MAPE value but not complete significantly (only receives significance with $\alpha = 0.05$ level in Wilcoxon test, but all fails with both levels in asymptotic test) outperforms SVRSA model. Figure 3.12 is provided to illustrate the forecasting accuracy among different models.

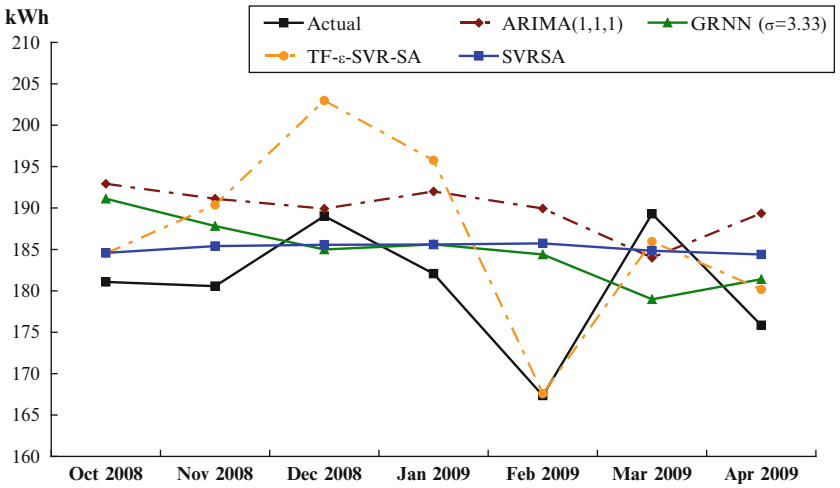

Fig. 3.12 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, and SVRSA models

The superior performance of the SVRSA model not only caused from several similar causes of SVRGA model, such as SVR-based model with nonlinear mapping capabilities, minimizing the structural risks rather than the training errors, but also caused from the searching mechanism of SA algorithm itself. In this section, the SA algorithm can successfully escape from some critical local minimum (forecasting error) of the three-parameter combination for electric load forecasting. However, it is also clear that SVRSA model is not fitting the actual electric loads very well even if it has significant smaller MAPE values than other alternatives. Therefore, it also still requires hybridizing other novel techniques to improve this shortcoming.

3.5 Hybrid GA with SA in SVR's Parameter Determination

3.5.1 Shortcomings of GA and SA

GA is auto-adaptive stochastic search technique [7] that is based on the Darwinian survival-of-the-fittest philosophy and generates new individuals with selection, crossover, and mutation operators. GA starts with a coding of the parameter set of all types of objective functions; thus, GA has the ability to solve those traditional algorithms that are not easy to solve. GA is able to reserve a few best fitted members of the whole population for the next generation in the operation process; however, after some generations, GA may lead to a premature convergence to a local optimum in searching the suitable parameters of a SVR model.

SA is a stochastic based general search tool that mimics the annealing process of material physics [10]. When the system in the original state is greater than that of the new generated state, this new state is automatically accepted. In contrast, the new state is accepted by Metropolis criterion with a probability function. The performance of SA is dependent on the cooling schedule. Thus, SA has some institution to be able to escape from local minima and reach to the global minimum [13]. However, SA costs more computation time. To ensure the efficiency of SA, a proper temperature cooling rate (stop criterion) should be considered.

To overcome these drawbacks from GA and SA, it is necessary to find some effective approach and improvement to avoid misleading to the local optimum and to search optimum objective function efficiently. Genetic algorithm–simulated annealing (GA–SA) hybrid algorithm is a novel trial in dealing with the challenges mentioned above. The GA–SA can firstly employ the superiority of SA algorithm to escape from local minima and approximate to the global minimum, and secondly apply the mutation process of GA to improve searching ability in the range of values. So, the hybrid algorithm has been applied to the fields of system design [14], system and network optimization [15, 16], query to information retrieval system [17], continuous-time production planning [18, 19], and electrical power districting problem [20]. However, there is little application of the GA–SA to SVR's parameter determination. This investigation presented in this book is motivated by a desire to solve the problem of maintaining the premature convergence to a local optimum of GA and the efficiency of SA mentioned above in determining the three free parameters in a SVR electric load forecasting model, namely, SVRGASA.

3.5.2 Operation Procedure of GA–SA Algorithm

To overcome the drawbacks from GA and SA, this study proposes a hybrid GA–SA algorithm by applying the superiority of SA to escape from local minima and approximate to the global minimum, in addition, by using the mutation process of GA to improve searching ability in the range of values. On the other hand, to avoid computation executing time consuming, only the optimal individual of GA population will be delivered to the SA for further improving. The proposed GA–SA algorithm consists of the GA part and the SA part. GA evaluates the initial population and operates on the population using three basic genetic operators to produce new population (best individual); then, for each generation of GA, it will be delivered to SA for further processing. After finishing all the processes of SA, the modified individual will be sent back to GA for the next generation. These computing iterations will be never stopped till the termination condition of the algorithm is reached. The proposed procedure of GA–SA algorithm is illustrated as follow and the flowchart is shown as Fig. 3.13.

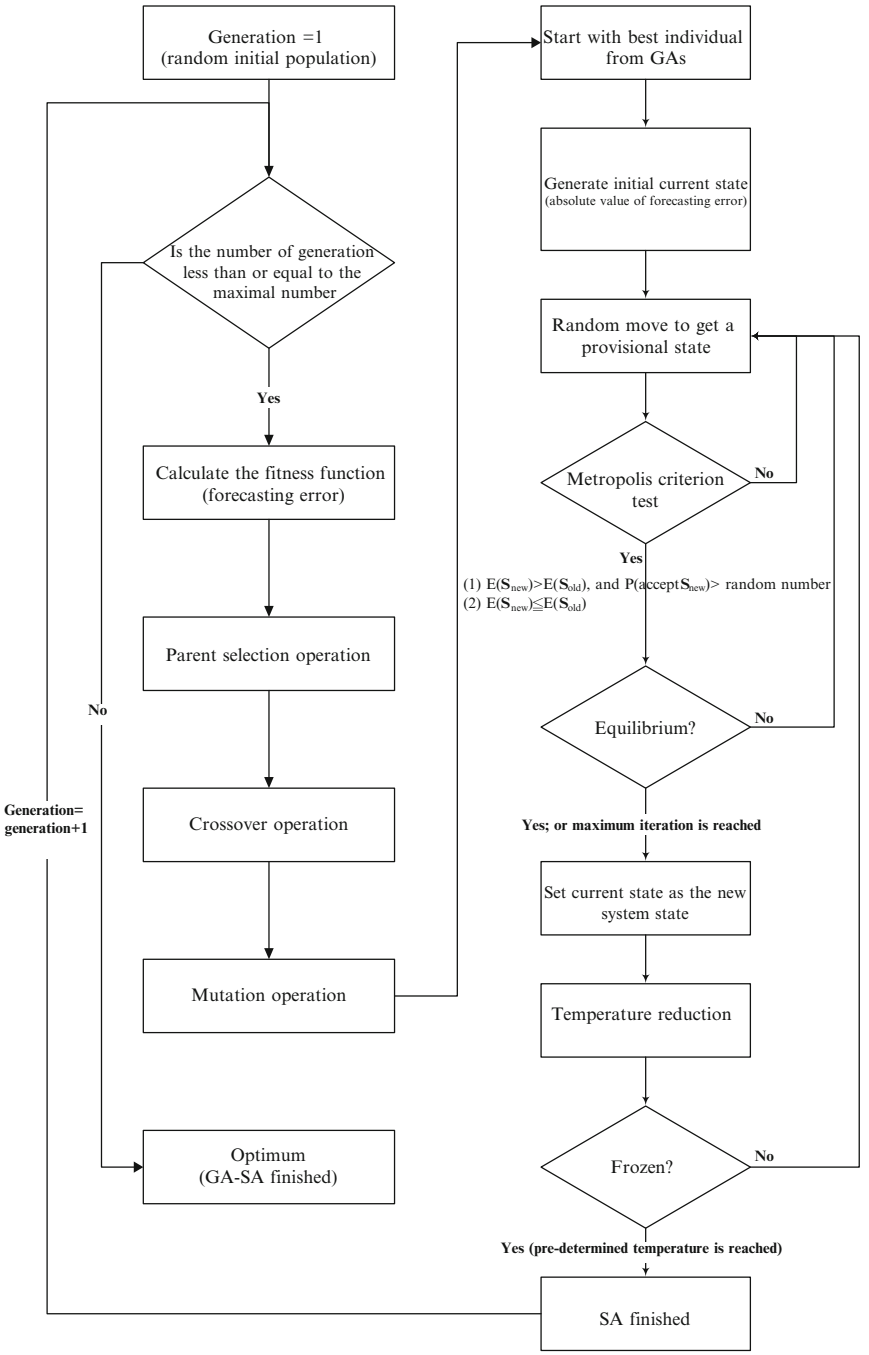

Fig. 3.13 The architecture of GA–SA algorithm

3.5.2.1 The Procedure of the GA Part

Step 1: Initialization. Construct randomly the initial population of chromosomes. The three parameters, C, σ, and ε, in a SVR model in the ith generation are encoded into a binary format and represented by a chromosome that is composed of "genes" of binary numbers (Fig. 3.2). Each chromosome has three genes, which represent three parameters. Each gene has 40 bits. For instance, if each gene contains 40 bits, a chromosome contains 120 bits. More bits in a gene correspond to finer partition of the search space.

Step 2: Evaluating fitness. Evaluate the fitness (forecasting errors) of each chromosome. In this book, a negative mean absolute percentage error (−MAPE) is used as the fitness function. The MAPE is as Eq. (3.13).

Step 3: Selection operation. Based on fitness functions, chromosomes with higher fitness values are more likely to yield offspring in the next generation. The roulette wheel selection principle [7] is also applied to choose chromosomes for reproduction.

Step 4: Crossover operation and mutation operation. Mutations are performed randomly by converting a "1" bit into a "0" bit or a "0" bit into a "1" bit. In crossover operation, chromosomes are paired randomly. The single-point-crossover principle is employed herein. Segments of paired chromosomes between two determined breakpoints are swapped. Finally, decode the crossover three parameters in a decimal format.

Step 5: Stop condition. If the number of generation is equal to a given scale, then the best chromosomes are presented as a solution; otherwise, go to the step 1 of the SA part.

In the proposed GA–SA algorithm process, GA will deliver its best individual to SA for further processing. After the optimal individual of GA being improved, SA sends it back to GA for the next generation. These computing iterations will be never stopped till the termination condition of the algorithm is reached.

3.5.2.2 The Procedure of the SA Part

Step 1: Generate initial current state. Receive values of the three parameters from GA. The value of forecasting error, MAPE, shown as Eq. (3.12), is defined as the system state (E). Here, the initial state (E_0) is obtained.

Step 2: Provisional state. Make a random move to change the existing system state to a provisional state. Another set of three positive parameters are generated in this stage.

Step 3: Metropolis criterion tests. Equation (3.16) is also employed to determine the acceptance or rejection of provisional state [9]. If the provisional state is accepted, then set the provisional state as the current state.

Step 4: Incumbent solutions. If the provisional state is not accepted, then return to step 2. Furthermore, if the current state is not superior to the system state, then repeat

Table 3.12 Parameter determination of SVRGASA model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	96.06	469.09	4.2588	5.049
10	22.45	99.97	0.9677	4.383
15	5.14	146.91	9.8969	3.951
20	788.75	6,587.20	9.2529	3.853
25	**92.09**	**2,449.50**	**13.639**	**3.530**

steps 2 and 3 until the current state is superior to the system state, and set the current state as new system state. The maximum number of loops (N_{sa}) is also set to 300.

Step 5: Temperature reduction. After the new system state is obtained, reduce the temperature. The new temperature reduction is obtained by the Eq. (3.17). If the predetermined temperature is reached, then stop the algorithm and the latest state is an approximate optimal solution. Otherwise, go to step 2.

3.5.3 GA–SA Algorithm for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with GA–SA), namely, SVRGASA model.

Similarly, in the training stage, the rolling-based forecasting procedure is also employed to obtain the forecasting load and receive training error in the training stage. Then, if training error improvement occurs, then the three kernel parameters, σ, C, and ε, of the SVRGASA model adjusted by GA–SA are employed to calculate the validation error. The adjusted parameters with minimum validation error are also selected as the most appropriate parameters. The forecasting results and the suitable parameters for the SVRGASA model are illustrated in Table 3.12, in which it is also indicated that these two models all perform the best when 25 fed-in data are used.

Table 3.13 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN($\sigma = 3.33$), TF-ε-SVR-SA, SVRGA, SVRSA, and SVRGASA. The MAPE values are calculated to compare fairly the proposed models with other alternative models. The proposed SVRGASA model has smaller MAPE values than other alternative models. Furthermore, to verify the significance of accuracy improvement of SVRGASA model comparing with other alternative models, the Wilcoxon signed-rank test and asymptotic test, as mentioned, are also conducted. The test results are shown in Tables 3.14 and 3.15, respectively. Clearly, the SVRGASA model is significantly superior to other alternative models. Figure 3.14 is provided to illustrate the forecasting accuracy among different models.

Table 3.13 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRGA, SVRSA, and SVRGASA models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA (1,1,1)	GRNN ($\sigma = 3.33$)	TF-ε-SVR-SA	SVRGA	SVRSA	SVRGASA
Oct. 2008	181.07	192.932	191.131	184.504	178.326	184.584	183.563
Nov. 2008	180.56	191.127	187.827	190.362	178.355	185.412	183.898
Dec. 2008	189.03	189.916	184.999	202.980	178.355	185.557	183.808
Jan. 2009	182.07	191.995	185.613	195.753	178.356	185.593	184.128
Feb. 2009	167.35	189.940	184.397	167.580	178.357	185.737	184.152
Mar. 2009	189.30	183.988	178.988	185.936	178.358	184.835	183.387
Apr. 2009	175.84	189.348	181.395	180.165	181.033	184.390	183.625
MAPE (%)		6.044	4.636	3.799	3.676	3.810	3.530

Table 3.14 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRGASA vs. ARIMA(1,1,1)	0[a]	0[a]
SVRGASA vs. GRNN($\sigma = 3.33$)	2[a]	2[a]
SVRGASA vs. TF-ε-SVR-SA	1[a]	1[a]
SVRGASA vs. SVRGA	0[a]	0[a]
SVRGASA vs. SVRSA	0[a]	0[a]

[a]Denotes that SVRGASA model significantly outperforms other alternative models

Table 3.15 Asymptotic test

	Asymptotic (S_1) test	
Compared models	$\alpha = 0.05$	$\alpha = 0.10$
SVRGASA vs. ARIMA (1,1,1)	H_0: $e_1 = e_2$ $S_1 = -10.965$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -10.965$; $p = 0.000$ (reject H_0)
SVRGASA vs. GRNN ($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -1.879$; $p = 0.03016$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -1.879$; $p = 0.03016$ (reject H_0)
SVRGASA vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -2.432$; $p = 0.00751$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.432$; $p = 0.00751$ (reject H_0)
SVRGASA vs. SVRGA	H_0: $e_1 = e_2$ $S_1 = 4.426$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = 4.426$; $p = 0.000$ (reject H_0)
SVRGASA vs. SVRSA	H_0: $e_1 = e_2$ $S_1 = -17.370$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -17.370$; $p = 0.000$ (reject H_0)

In this section, the hybrid GA–SA algorithm helps to avoid trapping into local minimum than GA and SA algorithms do, thus, outperforming the SVRGA and SVRSA models. For example, in Tables 3.4, 3.8, and 3.12, the GA–SA algorithm is

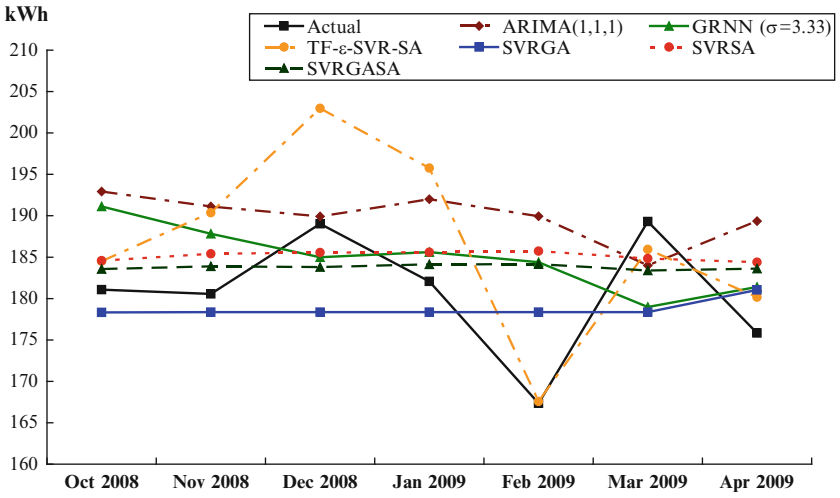

Fig. 3.14 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRGA, SVRSA, and SVRGASA models

excellently to shift the local solution of SVRGA and SVRSA models by 25 fed-in data rolling type, $(\sigma, C, \varepsilon) = (686.16, 5048.4, 19.317$ and $94.998, 9435.20, 12.6570$, respectively) with local optimal forecasting errors, in terms of MAPE (3.676 % and 3.810 %, respectively), to be improved by GA–SA algorithm to another better solution, $(\sigma, C, \varepsilon) = (92.807, 2449.50, 13.639)$ to be the appropriate local optimal forecasting error in terms of MAPE (3.530 %). Thus, it once again reveals that GA–SA algorithm is much appropriate than GA and SA algorithms in parameter adjustments to achieve forecasting accuracy improvement by integrated into the SVR model. However, it is also clear that SVRSA model is not fitting the actual electric loads very well even it has significant smaller MAPE values than other alternatives. Therefore, it also still requires hybridizing other novel techniques to improve this shortcoming.

3.6 Particle Swarm Optimization Algorithm in SVR's Parameter Determination

3.6.1 Operation Procedure of PSO Algorithm

In the previous sections, although both SVRGA and SVRSA are superior to other competitive forecasting models (ARIMA, HW, GRNN, and BPNN models), however, the drawbacks of GA and SA algorithms, as mentioned, are lacking knowledge memory or storage functions, while previous knowledge of the problem is destroyed once the population (GA) or the temperature changes (SA algorithm).

Thus, these drawbacks of GA and SA algorithms would lead to time-consuming and inefficient in searching the suitable parameters of a SVR model. Recently, inspired by the social behavior of organisms such as fish schooling and bird flocking, Kennedy and Eberhart [21] first introduced particle swarm optimization (PSO) algorithm, in which it is also initialized with a population of random solutions. Each individual, namely, particle, is assigned with a randomized velocity flown through hyperspace to look for the optimal position to land. Compared with GA and SA algorithm, PSO algorithm has memory to store the knowledge of good solutions by all particles; in addition, particles in the swarm share information with each other. Therefore, due to the simple concept, easy implementation, and quick convergence, nowadays PSO algorithm has gained much attention and wide applications in solving continuous nonlinear optimization problems [22]. However, the performance of PSO algorithm greatly depends on its parameters, and similar to GA and SA algorithm, it often suffers from being trapped in local optimum [23, 24].

In a PSO system, it starts with the random initialization of a population (swarm) in the search space, where multiple solution-candidates coexisted. Each solution, namely, particle, flies in the space looking for the optimal *position* to land. Eventually, the global best position of the system could be found out by adjusting the direction of each particle towards its own best location and towards the best particle of the swarm at each generation. The direction of each particle is adjusted by dynamically altering the *velocity* of each particle, according to its own flying experience as well as the experience of neighboring particles. During the searching process, tracking and memorizing the best position encountered could cumulate each particle's experience. Thus, the PSO system essentially has the capability of memory; each particle remembers the best position it reaches during the past, and then the PSO system combines local search method (via self experience) with global search methods (via neighboring experience).

The position, the velocity, and own best position of the ith particle pair, due to the three parameters in a SVR model, in the n-dimensional space can be represented as Eqs. (3.18)–(3.20), respectively,

$$X_{(k)i} = \left[x_{(k)i,1}, x_{(k)i,2}, \ldots, x_{(k)i,n}\right], \tag{3.18}$$

$$V_{(k)i} = \left[v_{(k)i,1}, v_{(k)i,2}, \ldots, v_{(k)i,n}\right], \tag{3.19}$$

$$P_{(k)i} = \left[p_{(k)i,1}, p_{(k)i,2}, \ldots, p_{(k)i,n}\right], \tag{3.20}$$

where $k = \sigma, C, \varepsilon$, and $i = 1, 2, \ldots, N$.

The global best position among all particles in the swarm $\mathbf{X}_{(k)i} = \left[X_{(k)1}, X_{(k)2}, \ldots, X_{(k)N}\right]$ is shown as Eq. (3.21):

$$P_{(k)g} = \left[p_{(k)g,1}, p_{(k)g,2}, \ldots, p_{(k)g,d}\right], \tag{3.21}$$

where $k = \sigma, C, \varepsilon$, and $g = 1, 2, \ldots, N$.

Then, the new velocity of each particle is computed by Eq. (3.22):

$$V_{(k)i}(t+1) = lV_{(k)i}(t) + q_1 \text{rand}(\cdot)(P_{(k)i} - X_{(k)i}(t)) + q_2 \text{Rand}(\cdot)(P_{(k)g} - X_{(k)i}(t)), \tag{3.22}$$

where $k = \sigma, C, \varepsilon$, and $i = 1, 2, \ldots, N$, l is called the inertia weight that controls the impact of the previous velocity of the particle on its current one, q_1 and q_2 are two positive constants called acceleration coefficients, and rand(·) and Rand(·) are two independently uniformly distributed random variables with range [0,1].

After the velocity has been updated, the new position of the particle for each parameter in the next generation is determined as Eq. (3.23):

$$X_{(k)i}(t+1) = X_{(k)i}(t) + V_{(k)i}(t+1), \tag{3.23}$$

where $k = \sigma, C, \varepsilon$, and $i = 1, 2, \ldots, N$.

Notice that the value of each component in $V_{(k)i}$ can be limited to the range $[-v_{\max}, v_{\max}]$ to control excessive roaming of particles outside the search space. This process is repeated until the defined stopping threshold is reached. The procedure of PSO algorithm is illustrated as follows and the flowchart is shown as Fig. 3.15. Interested readers could refer to [21] for more detail:

Step 1: Initialization. Initialize a defined population of particle pairs (σ_i, C_i, ε_i) with random positions ($X_{\sigma i}$, X_{Ci}, $X_{\varepsilon i}$) and velocities ($V_{\sigma i}$, V_{Ci}, $V_{\varepsilon i}$), where each particle contains n variables.

Step 2: Objective value computing. Compute the objective values (forecasting errors) of all particle pairs. Let own best position ($P_{\sigma i}$, P_{Ci}, $P_{\varepsilon i}$) of each particle pair and its objective value $f_{\text{best}i}$ equal to its initial position and objective value. Let global best position ($P_{\sigma g}$, P_{Cg}, $P_{\varepsilon g}$) and its objective value $f_{\text{globalbest}i}$ equal to the best initial particle pair's position and its objective value.

Step 3: Evaluation of the objective values. According to Eqs. (3.22) and (3.23), update the velocity and position for each particle pair. And evaluate the objective values for all particle pairs.

Step 4: Comparison and update. For each particle pair, compare its current objective value with $f_{\text{best}i}$. If the current objective value is better (i.e., with smaller forecasting accuracy index value), then update best position ($P_{\sigma i}$, P_{Ci}, $P_{\varepsilon i}$) and its objective value by the current position and objective value.

Step 5: The best particle pair determination. Determine the best particle pair of whole population based on the best objective value. If the objective value is smaller than $f_{\text{globalbest}i}$, then update ($P_{\sigma g}$, P_{Cg}, $P_{\varepsilon g}$) and its objective value with the current best particle pair's position and objective.

Step 6: Stop criterion. If a stopping threshold (forecasting accuracy) is reached, then ($P_{\sigma g}$, P_{Cg}, $P_{\varepsilon g}$) and its $f_{\text{globalbest}i}$ would be determined; otherwise, go back to step 3.

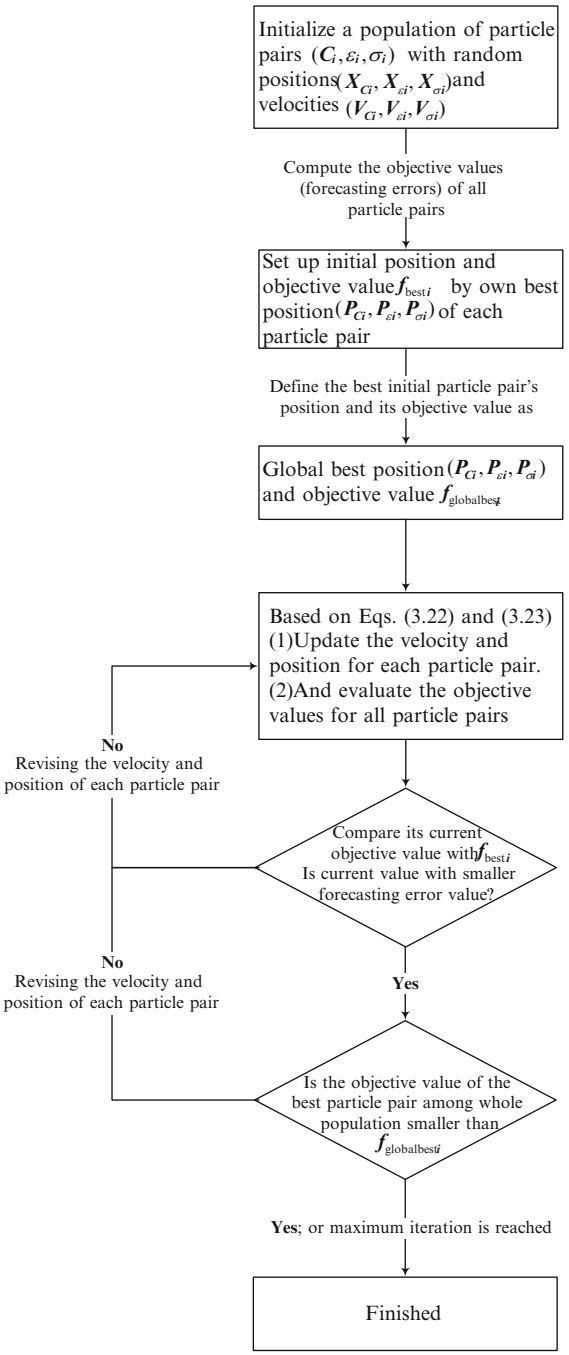

Fig. 3.15 The architecture of PSO algorithm

3.6.2 PSO Algorithm for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with PSO), namely, SVRPSO model.

The parameters of the PSO algorithm in the proposed model are experimentally set as shown in Table 3.16. The population size is set as 20; the total number of function evaluation is fixed as 10,000; q_1 and q_2 for each particle pair (σ, C, ε) are set to 0.05, 100, 0.5, respectively. $v_{\max}$ for σ particle is clamped to be 10 % of its search space (where $\sigma \in [0, 500]$). $v_{\max}$ for C particle is clamped to be 12.5 % of its search space ($C \in [0, 20000]$). $v_{\max}$ for ε particle are both clamped to be 15 % of its search space ($\varepsilon \in [0, 100]$).The standard PSO [21] uses a linearly varying inertia weight over the generations, varying from 1.2 at the beginning of the search to 0.2 at the end.

Similarly, in the training stage, the rolling-based forecasting procedure is also employed to obtain the forecasting load and receive training error in the training stage. Then, if training error improvement occurs, the three kernel parameter s, σ, C, and ε, of the SVRPSO model adjusted by PSO algorithm are employed to calculate the validation error. The adjusted parameters with minimum validation error are also selected as the most appropriate parameters. The forecasting results and the suitable parameters for the SVRPSO model are illustrated in Table 3.17, in which it is also indicated that these two models all perform the best when 25 fed-in data are used.

Table 3.18 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN($\sigma = 3.33$), TF-ε-SVR-SA, and SVRPSO. The MAPE values are calculated to compare fairly the proposed models with other alternative models. The proposed SVRPSO model has smaller MAPE values than other alternative models. Furthermore, to verify the significance of accuracy improvement of SVRPSO model comparing with other alternative models, the Wilcoxon signed-rank test and asymptotic test are also conducted. The test results are shown in Tables 3.19 and 3.20, respectively. Clearly, the SVRPSO model is significantly superior to other alternative models, except versus GRNN model (receives significance with both levels in Wilcoxon test, but all fails with $\alpha = 0.05$ level in asymptotic test). Figure 3.16 is provided to illustrate the forecasting accuracy among different models.

In this section, the PSO algorithm is employed to overcome the shortcomings of GA and SA algorithms, that is, with memory to store the knowledge of good solutions and sharing information with each other. PSO algorithm is capable of searching experience memorizing to avoid inefficacious searching paths and to quickly converge. Therefore, it is expected to receive better forecasting performance than SVRGA and SVRSA models. However, it is also clear that SVRPSO model is not fitting the actual electric loads very well even if it has significant smaller MAPE values than other alternatives. Therefore, it also still requires hybridizing other novel techniques to improve this shortcoming.

Table 3.16 Parameters setting of PSO algorithm

		Velocity limit						Inertia weight l		Acceleration coefficients					
		$-v_{\max}$			$v_{\max}$					q_1			q_2		
Population size	Maximal iteration	σ	C	ε	σ	C	ε	$l_{\min}$	$l_{\max}$	σ	C	ε	σ	C	ε
20	10,000	−0.5	−2,500	−15	0.5	2,500	15	0.2	1.2	0.05	100	0.5	0.05	100	0.5

Table 3.17 Parameter determination of SVRPSO model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	70.34	289.53	2.4341	4.558
10	23.82	81.12	1.2436	4.346
15	111.04	3,158.10	2.8713	4.484
20	93.32	5,683.70	11.4980	4.078
25	**158.44**	**7,014.50**	**2.2836**	**3.638**

Table 3.18 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRGA, SVRSA, and SVRPSO models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA (1,1,1)	GRNN ($\sigma = 3.33$)	TF-ε-SVR-SA	SVRGA	SVRSA	SVRPSO
Oct. 2008	181.07	192.932	191.131	184.504	178.326	184.584	184.042
Nov. 2008	180.56	191.127	187.827	190.361	178.355	185.412	183.577
Dec. 2008	189.03	189.916	184.999	202.980	178.355	185.557	183.471
Jan. 2009	182.07	191.995	185.613	195.753	178.356	185.593	184.210
Feb. 2009	167.35	189.940	184.397	167.580	178.357	185.737	184.338
Mar. 2009	189.30	183.988	178.988	185.936	178.358	184.835	183.725
Apr. 2009	175.84	189.348	181.395	180.165	181.033	184.390	184.529
MAPE (%)		6.044	4.636	3.799	3.676	3.810	3.638

Table 3.19 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRPSO vs. ARIMA(1,1,1)	0[a]	0[a]
SVRPSO vs. GRNN($\sigma = 3.33$)	2[a]	2[a]
SVRPSO vs. TF-ε-SVR-SA	2[a]	2[a]
SVRPSO vs. SVRGA	0[a]	0[a]
SVRPSO vs. SVRSA	1[a]	1[a]

[a]Denotes that SVRPSO model significantly outperforms other alternative models

3.7 Continuous Ant Colony Optimization Algorithm in SVR's Parameter Determination

3.7.1 Basic Concept of ACO Algorithm

Ant colony optimization (ACO) algorithm was firstly proposed by Dorigo [25] and Dorigo et al. [26]. The process by which ants could establish the shorter path between ant nest and food is illustrated in Fig. 3.17. Initially, ants leave their nest in random directions to search for food. As roaming around, ants deposit some

Table 3.20 Asymptotic test

Compared models	Asymptotic (S_1) test	
	$\alpha = 0.05$	$\alpha = 0.10$
SVRPSO vs. ARIMA(1,1,1)	H_0: $e_1 = e_2$ $S_1 = -9.677$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -9.677$; $p = 0.000$ (reject H_0)
SVRPSO vs. GRNN($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -1.567$; $p = 0.0586$ (not reject H_0)	H_0: $e_1 = e_2$ $S_1 = -1.567$; $p = 0.0586$ (reject H_0)
SVRPSO vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -1.852$; $p = 0.0320$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -1.852$; $p = 0.0320$ (reject H_0)
SVRPSO vs. SVRGA	H_0: $e_1 = e_2$ $S_1 = 5.863$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = 5.863$; $p = 0.000$ (reject H_0)
SVRPSO vs. SVRSA	H_0: $e_1 = e_2$ $S_1 = -5.992$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -5.992$; $p = 0.000$ (reject H_0)

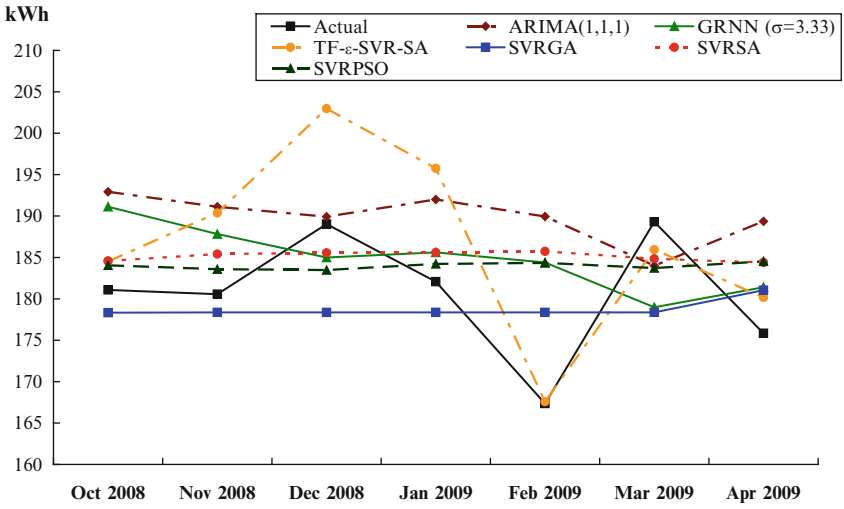

Fig. 3.16 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRGA, SVRSA, and SVRPSO models

amount of pheromone trails, which could be detectable by other ants. For example, assuming ant 1 finds a food source, it will pick up some food and go back to the nest by following its previous pheromone trail, laying *additional* pheromone on the same path while other ants (ant 2, ant 3, etc.) are still roaming randomly. When the second ant group leaves the nest to look for food, those ants could detect much

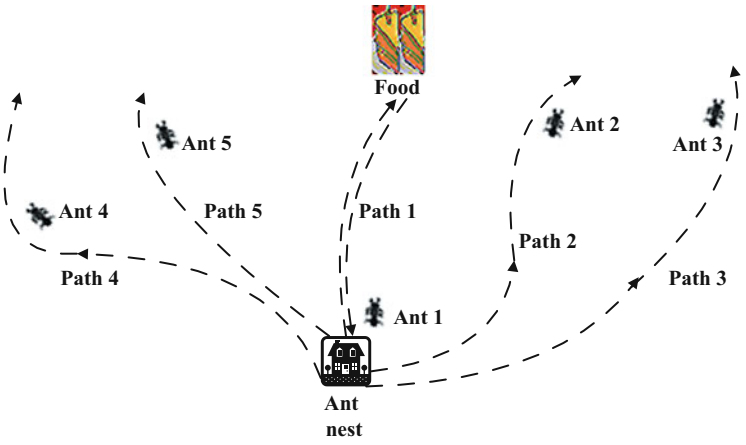

Fig. 3.17 Description of establishing the shorter path between ant nest and food

pheromone (twice) on path 1 than on others (path 2, path 3, etc.). Since the probability for a path to be followed is determined by its pheromone amount, more ants will follow path 1 in this second round of looking for food. In this way, the ants can establish the shortest path from their colony to the food sources. Obviously, even if an isolated ant roams randomly, it can, communicated by pheromones, follow the collective behavior of ant colonies.

Due to their power to learn and search capabilities, ACO algorithm has been successfully used to dealing with different combinatorial optimization problems including job-shop scheduling [27], traveling salesman problem [28], space planning [29], quadratic assignment problems [30], and data mining [31]. ACO imitates the behaviors of real ant colonies as they forage for food, wherein each ant lays down the pheromone on the path to the food sources or back to the nest. The paths with more pheromone are more likely to be selected by other ants. Over time, a colony of ants will select the shortest path to the food source and back to the nest. Therefore, a pheromone trail is the most important process for individual ant to smell and select its route. Meanwhile, ACO algorithm is originally proposed for discrete optimization, and their application to continuous optimization problems requires some specified transformation techniques. In the literature, only a few approaches for continuous optimization have been proposed, such as continuous ACO [32–34], API algorithm [35], and continuous interacting ACO [36]. However, these algorithms added some operational mechanisms that are mostly beyond the regular essences of ACO. Recently, Socha and Dorigo [37] proposed an extension of ACO to continuous domain by applying the continuous probability density function to decide pheromone probabilistic choice, in which, however, other external parameters should be determined in advance; thereby, it would mislead to continuous technological issue instead of appropriate SVR's parameter determination.

3.7.2 Continuing Transformation

Hence, the concepts of transforming a continuous search space to a discrete one by discretization of the continuous decision variables [38] are feasible to be employed, which is so-called the continuous ant colony optimization (CACO) algorithm. In this book, the CACO algorithm for the traveling salesman problem is modified to determine three parameters of a SVR model in the discrete search space. The probability, $P_k(i,j)$, that an ant k moves from city i to city j is expressed as Eq. (3.24):

$$P_k(i,j) = \begin{cases} \arg\max\limits_{S \in M_k} \left\{ [\tau(i,S)]^{\alpha} [\eta(i,S)]^{\beta} \right\}, & \text{if } q \le q_0 \\ \text{Eq.}(3.25), & \text{otherwise} \end{cases}, \tag{3.24}$$

$$P_k(i,j) = \begin{cases} [\tau(i,j)]^{\alpha} [\eta(i,j)]^{\beta} \Big/ \sum\limits_{S \in M_k} [\tau(i,S)]^{\alpha} [\eta(i,S)]^{\beta}, & j \notin M_k \\ 0, & \text{otherwise} \end{cases}, \tag{3.25}$$

where $\tau(i,j)$ is the pheromone level between city i and city j, and $\eta(i,j)$ is the inverse of the distance between cities i and j. In this study, the forecasting error represents the distance between cities. The α and β are parameters determining the relative importance of pheromone level, and M_k is a set of cities in the next column of the city matrix for ant k. q is a random uniform variable [0,1], and the value q_0 is a constant between 0 and 1, that is, $q_0 \in [0,1]$. The values of α, β, and q_0 are set to be 8, 5, and 0.2, respectively.

Once ants have completed their tours, the most pheromone deposited by ants on the visited paths is considered as the information regarding the best paths from the nest to the food sources. Therefore, the pheromone dynamic updating plays the main role in real ant colonies searching behaviors. The local and global updating rules of pheromone are expressed as Eqs. (3.26) and (3.27), respectively,

$$\tau(i,j) = (1-\rho)\tau(i,j) + \rho\tau_0, \tag{3.26}$$

$$\tau(i,j) = (1-\delta)\tau(i,j) + \delta\Delta\tau(i,j), \tag{3.27}$$

where ρ is the local evaporation rate of pheromone, $0 < \rho < 1$; τ_0 is the initial amount of pheromone deposited on each of the paths. In this book, the value of ρ is set to be 0.01. In addition, the initial amount of pheromone, τ_0, generated by Dorigo and Gambardella's [28] proposed approach, is expressed as Eq. (3.28):

$$\tau_0 = \frac{1}{nL_{\text{nn}}} \tag{3.28}$$

where n is the number of cities and L_{nn} is the tour length produced by the nearest neighbor heuristic.

Global trail updating is accomplished according to Eq. (3.27). The δ is the global pheromone decay parameter, $0 < \delta < 1$, and set to be 0.2 for this study. The $\Delta\tau(i,j)$, expressed as Eq. (3.29), is used to increase the pheromone on the path of the solution:

$$\Delta\tau(i,j) = \begin{cases} 1/L, & \text{if } (i,j) \in \text{global best route} \\ 0, & \text{otherwise} \end{cases}, \tag{3.29}$$

where L is the length of the shortest route.

3.7.3 Operation Procedure of CACO Algorithm

More detail of the CACO algorithm on this book is as follows and the flowchart is shown as Fig. 3.18.

Step 1: Initialization. Set upper bounds of three SVR positive parameters, σ, C, and ε. In this study, to discretize those continuous parameters, each digit of the parameters is represented by ten cities. Thus, each digit contains 10 possible values from 0 to 9. Assume the limits of parameters σ, C, and ε are 500, 10,000, and 100, correspondingly. The numbers of digits that represent each parameter (σ, C, and ε) are all set as six. Hence, three ant colonies are defined as σ-ant colony, C-ant colony, and ε-ant colony for three-parameter values searching. The numbers of cities for each ant colony are 40, and the total number of cities is 120.

Step 2: Assigning tasks to each ant colony. From step one, pathway-structure list of each ant colony would be generated. Figure 3.19 shows the parameters represented by the CACO algorithms and pathway-structure list in this study. Each ant will randomly select a pathway from the pathway list in its associate colony and remember the values of the represented parameters (σ, C, and ε). At the end of the pathway, pass the three-parameter values into the SVR model (i.e., objective function) and calculate the forecasting error. The shortest travel pathway in each searching loop would be determined based on the smaller forecasting error. In this book, the MAPE is used as the forecasting error index as given by Eq. (3.13).

Step 3: Determine the numbers of ants and calculate the distance between cities. The numbers of ants are set to be 10 in each ant colony searching, that is, totally 30 ants for each iteration searching. The maximum number of iterations is set to 20,000 to avoid infinite iterations.

Step 4: Stop criterion I. While the maximum number of iterations is reached, then stop the algorithm, and the shortest travel path of the ants colony is an approximate optimal solution. Otherwise, continue to step 4.

Step 5: Calculate the visiting probability. If the maximum number of iterations is not reached, then calculate the probability that an ant k in city i moves to city j in accordance with Eq. (3.24). Repeat steps 1–3.

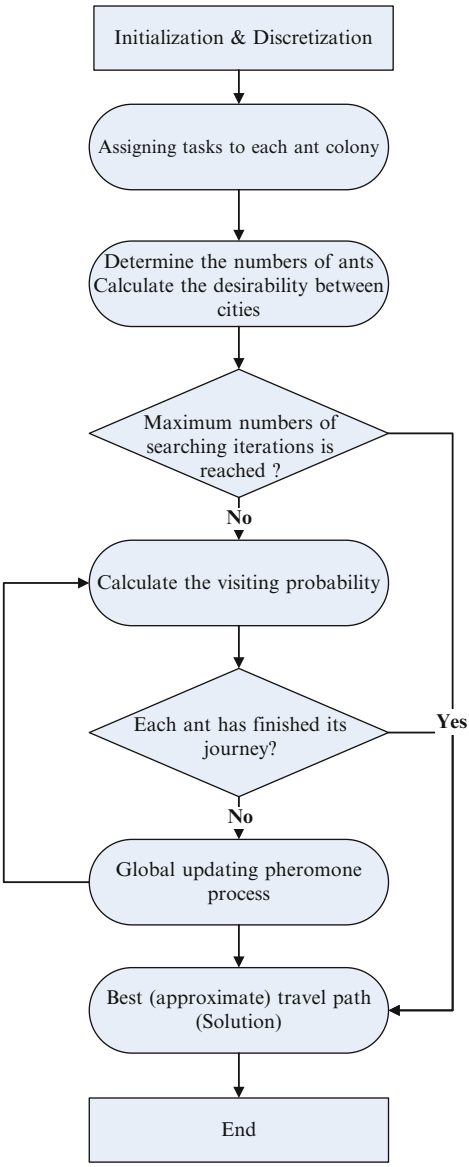

Fig. 3.18 The architecture of CACO algorithm

Step 6: Stop criterion II. If each ant has finished its pathway-structure list from the nest to the food source passing through all cities, then the shortest path is an approximate optimal solution. Otherwise, conduct the pheromone updating process represented as Eqs. (3.26) and (3.27) to renew the reinforcement of pheromone. Then, go back to step 4.

Notice that, in any iteration, while the shorter path is attained, the appropriate solution is determined, and for those three parameters, new search space is then

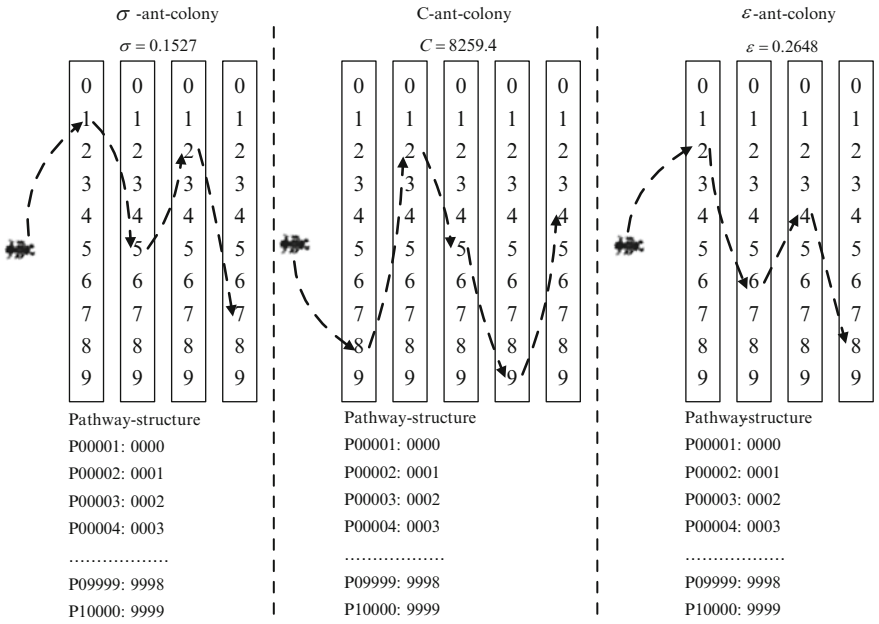

Fig. 3.19 SVR parameter representation by the CACO algorithm

re-discretized. The CACO algorithm is used to seek a better combination of the three parameters in the SVR so that a smaller MAPE is obtained during forecasting iteration.

3.7.4 CACO Algorithm for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with CACO), namely, SVRCACO model.

Similarly, in the training stage, the rolling-based forecasting procedure is also employed to obtain the forecasting load and receive training error in the training stage. Then, if training error improvement occurs, the three kernel parameters, σ, C, and ε, of the SVRCACO model adjusted by CACO algorithm are employed to calculate the validation error. The adjusted parameters with minimum validation error are also selected as the most appropriate parameters. The forecasting results and the suitable parameters for the SVRCACO model are illustrated in Table 3.21, in which it is also indicated that these two models all perform the best when 25 fed-in data are used.

Table 3.22 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN(σ = 3.33), TF-ε-SVR-SA, SVRPSO, and SVRCACO. The MAPE values are calculated to compare fairly the proposed

Table 3.21 Parameter determination of SVRCACO model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	1.49	322.92	6.7778	5.623
10	159.76	198.03	4.5219	5.076
15	12.81	114.24	0.0035	4.510
20	22.99	7,233.00	13.7640	4.003
25	**243.55**	**6,868.10**	**11.2480**	**3.371**

Table 3.22 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRPSO, and SVRCACO models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA (1,1,1)	GRNN ($\sigma = 3.33$)	TF-ε-SVR-SA	SVRPSO	SVRCACO
Oct. 2008	181.07	192.932	191.131	184.504	184.042	180.876
Nov. 2008	180.56	191.127	187.827	190.361	183.577	182.122
Dec. 2008	189.03	189.916	184.999	202.980	183.471	184.610
Jan. 2009	182.07	191.995	185.613	195.753	184.210	185.233
Feb. 2009	167.35	189.940	184.397	167.580	184.338	185.274
Mar. 2009	189.30	183.988	178.988	185.936	183.725	184.247
Apr. 2009	175.84	189.348	181.395	180.165	184.529	184.930
MAPE (%)		6.044	4.636	3.799	3.638	3.371

Table 3.23 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRCACO vs. ARIMA(1,1,1)	1[a]	1[a]
SVRCACO vs. GRNN($\sigma = 3.33$)	3	2[a]
SVRCACO vs. TF-ε-SVR-SA	2[a]	2[a]
SVRCACO vs. SVRPSO	2[a]	2[a]

[a]Denotes that SVRCACO model significantly outperforms other alternative models

models with other alternative models. The proposed SVRCACO model has smaller MAPE values than other alternative models. Furthermore, to verify the significance of accuracy improvement of SVRCACO model comparing with other alternative models, the Wilcoxon signed-rank test and asymptotic test are also conducted. The test results are shown in Tables 3.23 and 3.24, respectively. Clearly, the SVRCACO model is significantly superior to other alternative models, except GRNN model (minor significantly superior to GRNN model, only receives significance with $\alpha = 0.05$ level in Wilcoxon test, but all fails with both levels in asymptotic test) and SVRPSO model (not completely significantly outperforms SVRPSO model, only receives significance with both levels in Wilcoxon test, but all fails with both levels in asymptotic test). Figure 3.20 is provided to illustrate the forecasting accuracy among different models.

Table 3.24 Asymptotic test

	Asymptotic (S_1) test	
Compared models	$\alpha = 0.05$	$\alpha = 0.10$
SVRCACO vs. ARIMA(1,1,1)	H_0: $e_1 = e_2$ $S_1 = -7.174$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -7.174$; $p = 0.000$ (reject H_0)
SVRCACO vs. GRNN($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -1.201$; $p = 0.1149$ (not reject H_0)	H_0: $e_1 = e_2$ $S_1 = -1.201$; $p = 0.1149$ (not reject H_0)
SVRCACO vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -2.018$; $p = 0.0218$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.018$; $p = 0.0218$ (reject H_0)
SVRCACO vs. SVRPSO	H_0: $e_1 = e_2$ $S_1 = 0.6341$; $p = 0.263$ (not reject H_0)	H_0: $e_1 = e_2$ $S_1 = 0.6341$; $p = 0.263$ (not reject H_0)

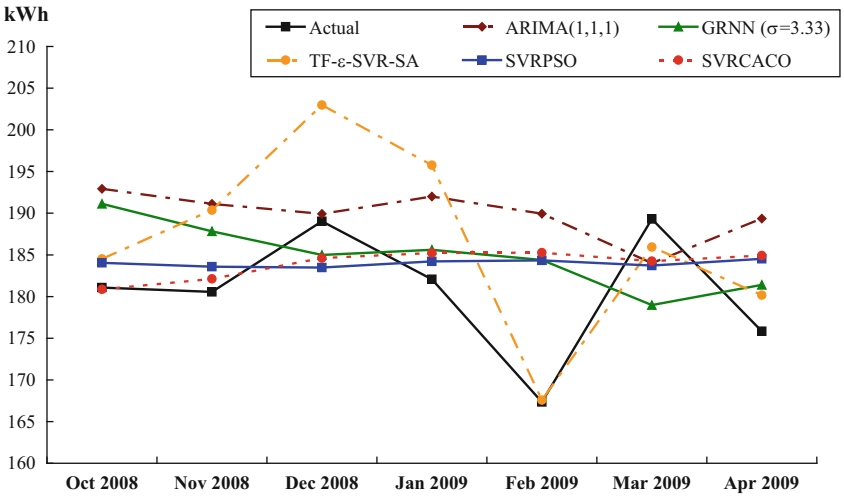

Fig. 3.20 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRPSO, and SVRCACO models

In this section, the CACO algorithm is also applied to overcome the shortcomings of GA and SA algorithms, that is, with memory to store the knowledge of good solutions and sharing information with each other. CACO algorithm has such the mechanism of learning/searching experiences storage and feedback to establish the shorter path (i.e., suitable parameter combination of a SVR model) between ant nest and food (i.e., smaller forecasting error). Therefore, it is expected to receive better forecasting performance than SVRGA and SVRSA models and has the potential to provide some competitive solution comparing with SVRPSO model. However, it is also clear that SVRCACO model is not fitting the actual electric loads very well even

it has significant smaller MAPE values than other alternatives. Therefore, it also still requires hybridizing other novel techniques to improve this shortcoming.

3.8 Artificial Bee Colony Algorithm in SVR's Parameter Determination

3.8.1 Behaviors of Real Bees

The artificial bee colony (ABC) algorithm, proposed by Karaboga et al. [39] and further developed by Karaboga and Basturk [40–42], is inspired by the intelligent foraging behavior of honeybee swarm. As known that lots of optimization algorithms conduct only one search operation during one iteration time, for example, PSO algorithm carries out global search at the beginning and local search in the later stage. For ABC algorithm, it conducts both global search and local search during one iteration time to ensure that ABC algorithm is more probabilistic to receive more suitable parameter combination, and thereby efficiently to avoid local optimum to a large extent, and to receive better performance in optimization problem compared with GA, differential evolution (DE), and PSO algorithm [40–43].

The whole fabric of honeybee society depends on various communication ways among bees, such as waggle dance and special odor, to easily find food sources that produce relatively high amount of nectar [44]. To introduce this kind of forage selection model that leads to the emergence of collective intelligence of honeybee swarms, three essential components are defined: food sources, unemployed foragers, and employed foragers. In addition, two leading modes of the behavior, recruitment to a nectar source and abandonment of a source, are also embedded [42]:

1. Food sources (A and B in Fig. 3.21). The value of a food source depends on many factors, such as its proximity to the nest, richness or concentration of energy, and the ease of extracting this energy. For the sake of simplicity, the "profitability" of a food source can be represented with a single quantity.
2. Unemployed foragers (UF in Fig. 3.21). It is assumed that a bee has no prior knowledge about the food sources in the search field; thus, bee initializes its search as an unemployed forager. Unemployed foragers are looking for a food source to exploit. There are three types of unemployed foragers: scouts, onlookers, and recruits.

   (a) Scouts (S in Fig. 3.21). Without any prior knowledge, they start searching spontaneously for new food sources around the nest. The mean number of scouts averaged over conditions is about 10 % in nature [45].

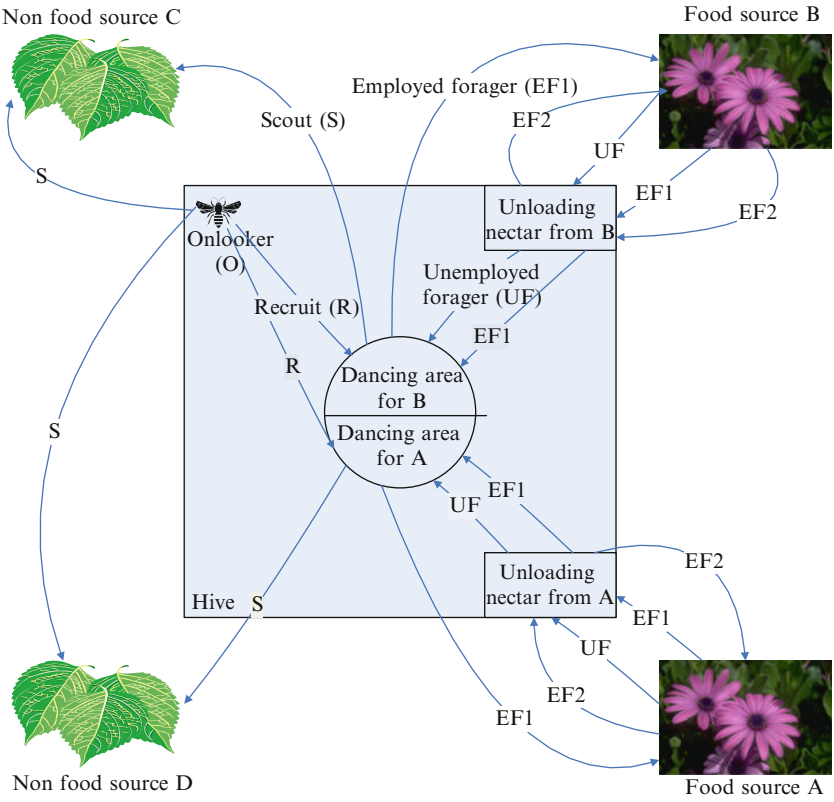

Fig. 3.21 Behavior of honeybee foraging for nectar

(b) Onlookers (O in Fig. 3.21). They wait in the nest and find a food source through the shared information from the employed foragers. There is a greater probability of onlookers choosing more profitable sources [43].
(c) Recruits (R in Fig. 3.21). If the onlookers attend a waggle dance done by some other bees, these onlookers will become recruits and start searching by using the obtained (shared) knowledge from the waggle dance.

3. Employed foragers. They are associated with a particular food source which they are currently exploiting or are "employed" at. They carry the information (profitability) about this particular source and share this information by a certain probability. After the employed foraging bee loads a portion of nectar from the food source, it returns to the hive and unloads the nectar to the food area in the hive. Then, the foraging bee has three possible behaviors related to residual amount of nectar to be acted.

(a) Unemployed foragers (UF in Fig. 3.21). If the nectar amount is decreased to a low critical level or exhausted, the foraging bee abandons the food source and becomes an unemployed bee.

(b) Employed foragers type 1 (EF1 in Fig. 3.21). The foraging bee can go to the dance area to perform waggle dance to inform the nest mates about the food source.
(c) Employed foragers type 2 (EF2 in Fig. 3.21). On the contrary, if there are still sufficient amounts of nectar in the food source, the foraging bee can continue to forage without communicating the food source information with the nest mates.

The communication of information among bees is the most important matter in the whole fabric of honeybee society. Communication among bees related to the quality of food sources occurs in the dancing area. The related dance is called waggle dance, which closely provides the information correlated with the direction of and the distance from the food sources. Employed foragers share their information with a probability, which is proportional to the profitability of the food source. Hence, the recruitment is proportional to the profitability of a food source [46].

3.8.2 *Operation Procedure of ABC Algorithm*

The proposed procedure of ABC algorithm is illustrated as follows and the flowchart is shown as Fig. 3.22:

Step 1: Initialization. Initialize the population size N_p, the number of employed foragers n_e, and the number of unemployed foragers (onlookers) n_0, which satisfy the condition, $N_p = n_e + n_0$. Let x_{ij} $(i = 1, 2, \ldots, N_p; j = 1, 2, \ldots, D)$ represent the initial solution of parameter combination in a SVR model, where D is the number of parameters. D is set as 3 in this book.

Step 2: Criteria of food source determination. Based on ABC algorithm, choosing a food source of an onlooker is dependent on the probability value associated with that food source. However, for forecasting accuracy improvement in the investigation, the onlooker will choose a food source according to the mean absolute percentage error (MAPE), shown as Eq. (3.13).

Step 3: Generate neighboring food sources (potential solutions). A candidate food source (solution) v_{ij} from the old solution x_{ij} can be generated as Eq. (3.30):

$$v_{ij} = x_{ij} + \Phi_{ij}(x_{ij} - x_{kj}), \tag{3.30}$$

where $k \in \{1, 2, \ldots, N\}$ is randomly chosen index, k must be different from i, and Φ_{ij} is a random number in the range $[-1,1]$. If the MAPE value of the candidate solution v_{ij} is equal to or smaller than x_{ij}'s, then v_{ij} will be set as the new solution; else, x_{ij} will remain as the solution. The parameter Φ_{ij} in ABC is the key factor that affects convergence [65].

Step 4: Determine the abandoned food source. If a solution cannot be improved through a predetermined threshold (limited iterations), then the food source is

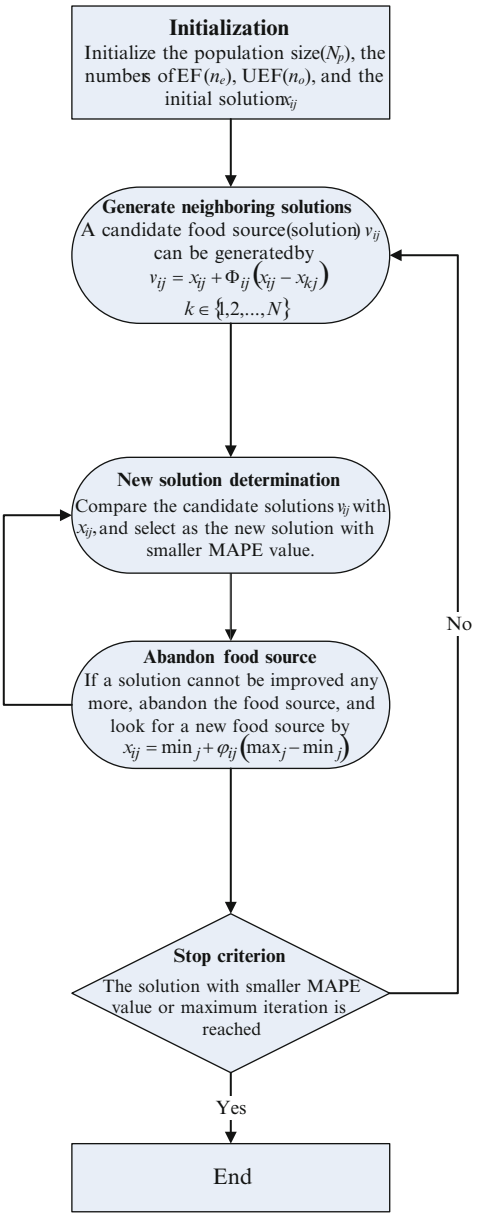

Fig. 3.22 The architecture of artificial bee colony (ABC) algorithm

considered to be abandoned. The employed bee will reinstate to be a scout, according to Eq. (3.31), to look for another new food source to replace the abandoned source:

$$x_{ij} = \min_j + \varphi_{ij}(\max_j - \min_j), \tag{3.31}$$

Table 3.25 Parameter determination of SVRABC model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	115.78	130.01	2.9542	3.812
10	193.26	44.08	2.4476	3.665
15	30.27	9,652.50	12.7640	3.509
20	620.15	4,246.00	13.1820	3.588
25	**38.348**	**4,552.10**	**16.8450**	**3.458**

where $\max_j$ is the maximal solution, that is, $\max_j = \max\{x_{1j}, x_{2j}, \ldots, x_{Nj}\}$; $\min_j$ represents the minimal solution, that is, $\min_j = \min\{x_{1j}, x_{2j}, \ldots, x_{Nj}\}$; and φ_{ij} is a random number in the range $[-1,1]$.

Step 5: Stop criterion. If the new food source value is with smaller MAPE value or maximum iteration is reached, then the new three parameters $x_i^{(n+1)}$ and its corresponding objective value are the final solution; otherwise, go to the next iteration and go back to step 2.

3.8.3 ABC Algorithm for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with ABC), namely, SVRABC model.

Similarly, in the training stage, the rolling-based forecasting procedure is also employed to obtain the forecasting load and receive training error in the training stage. Then, if training error improvement occurs, the three kernel parameters, σ, C, and ε, of the SVRABC model adjusted by ABC algorithm are employed to calculate the validation error. The adjusted parameters with minimum validation error are also selected as the most appropriate parameters. The forecasting results and the suitable parameters for the SVRABC model are illustrated in Table 3.25, in which it is also indicated that these two models all perform the best when 25 fed-in data are used.

For simplified comparison among alternative models, SVRGA, SVRSA, and SVRGASA models are not considered due to their law forecasting accuracy levels; SVRCACO model is also not included in comparison due to minor relationship between CACO and ABC algorithms. Table 3.26 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN ($\sigma = 3.33$), TF-ε-SVR-SA, SVRPSO, and SVRABC. The MAPE values are calculated to compare fairly the proposed models with other alternative models. The proposed SVRABC model has smaller MAPE values than other alternative models. Furthermore, to verify the significance of accuracy improvement of SVRABC model comparing with other alternative models, the Wilcoxon signed-rank test and asymptotic test are also conducted. The test results are shown in Tables 3.27

Table 3.26 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRPSO, and SVRABC models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA (1,1,1)	GRNN ($\sigma = 3.33$)	TF-ε-SVR-SA	SVRPSO	SVRABC
Oct. 2008	181.07	192.932	191.131	184.504	184.042	184.498
Nov. 2008	180.56	191.127	187.827	190.361	183.577	183.372
Dec. 2008	189.03	189.916	184.999	202.980	183.471	183.323
Jan. 2009	182.07	191.995	185.613	195.753	184.210	183.549
Feb. 2009	167.35	189.940	184.397	167.580	184.338	183.774
Mar. 2009	189.30	183.988	178.988	185.936	183.725	183.999
Apr. 2009	175.84	189.348	181.395	180.165	184.529	183.420
MAPE (%)		6.044	4.636	3.799	3.638	3.458

Table 3.27 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRABC vs. ARIMA(1,1,1)	1[a]	1[a]
SVRABC vs. GRNN($\sigma = 3.33$)	2[a]	2[a]
SVRABC vs. TF-ε-SVR-SA	2[a]	2[a]
SVRABC vs. SVRPSO	2[a]	2[a]

[a]Denotes that SVRABC model significantly outperforms other alternative models

Table 3.28 Asymptotic test

	Asymptotic (S_1) test	
Compared models	$\alpha = 0.05$	$\alpha = 0.10$
SVRABC vs. ARIMA (1,1,1)	H_0: $e_1 = e_2$ $S_1 = -13.231$; $p = 0.000$(reject H_0)	H_0: $e_1 = e_2$ $S_1 = -13.231$; $p = 0.000$(reject H_0)
SVRABC vs. GRNN ($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -2.257$; $p = 0.01199$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.257$; $p = 0.01199$ (reject H_0)
SVRABC vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -2.066$; $p = 0.0194$(reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.066$; $p = 0.0194$(reject H_0)
SVRABC vs. SVRPSO	H_0: $e_1 = e_2$ $S_1 = -2.723$; $p = 0.0032$(reject H_0)	H_0: $e_1 = e_2$ $S_1 = -2.723$; $p = 0.0032$(reject H_0)

and 3.28, respectively. Clearly, the SVRABC model is significantly superior to other alternative models. Figure 3.23 is provided to illustrate the forecasting accuracy among different models.

In this section, the ABC algorithm is also applied to overcome the shortcomings of PSO algorithm (only carries out global search at the beginning and local search in the later stage), that is,, conducting both global search and local search during one

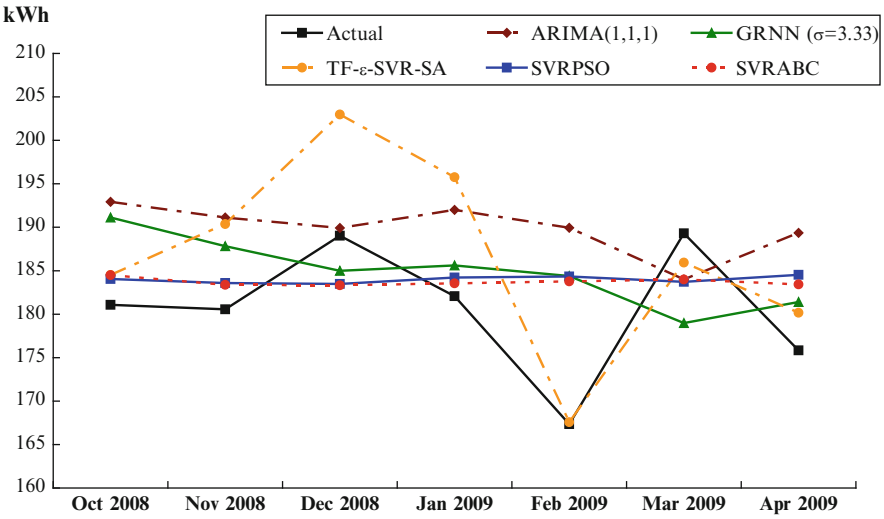

Fig. 3.23 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRPSO, and SVRABC models

iteration time to enrich the searching behavior to avoid trapping into local optimum. Therefore, it is also expected to receive better forecasting performance than SVRPSO model and has the potential to provide some alternative solution comparing with SVRCACO model. However, it is also clear that SVRABC model is not fitting the actual electric loads very well even if it has significant smaller MAPE values than other alternatives. Therefore, it also still requires hybridizing other novel techniques to improve this shortcoming.

3.9 Immune Algorithm in SVR's Parameter Determination

3.9.1 Operation Procedure of IA

The immune algorithm (IA), proposed by Mori et al. [47] and used in this book, is based on the learning mechanism of natural immune systems. Similar to GA, SA, and PSO, IA is also a population-based evolutionary algorithm; therefore, it provides a set of solutions for exploration and exploitation of search space to obtain optimal/near optimal solutions [48]. The natural immune system is a complex adaptive system that efficiently employs several mechanisms to recognize all cells within the body and classify those cells as self or nonself. Additionally, the nonself cells are further categorized to stimulate an appropriate type of defensive mechanism for defending against foreign invaders, such as bacteria and viruses. The lymphocyte is the main type of immune cell participating in the immune response. The lymphocyte contains two subclasses: *T* and *B*. Each subclass has its own function. When an antigen enters

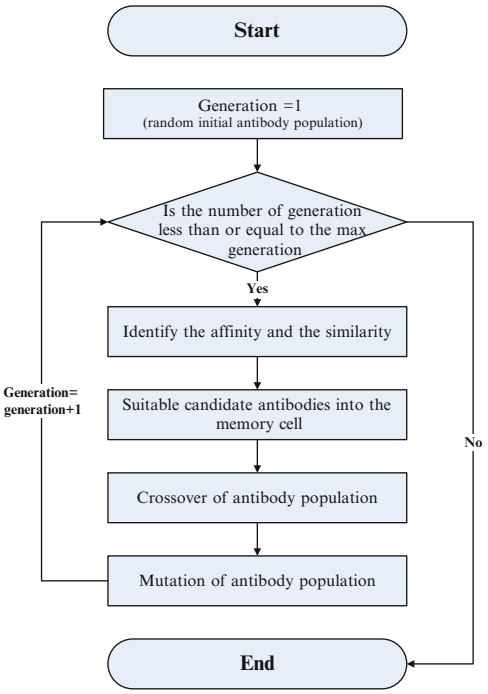

Fig. 3.24 The architecture of immune algorithm (IA)

the bloodstream and lymphatic system, the antigen encounters *B*-cells, and antibodies anchored in the membrane of *B*-cells recognize antigens in the bacteria. *T*-cells, which have already received communication from macrophages about the antigen, then communicate with *B*-cells and stimulate their proliferation. The proliferated *B*-cells turn into memory cells and produce antibodies. After the antibodies enter the bloodstream via the heart, the antibodies bind to antigens and kill them with the help of macrophages and other proteins.

Analogous to the natural immune system, the IA has the ability to seek out the best solution for optimization problems. In the IA procedure, the optimization problem can be viewed as antigens. Conversely, the feasible solutions of the optimization problem are treated as antibodies (*B*-cells). The procedure of IA is illustrated as follows and the flowchart is shown as Fig. 3.24.

Step 1: Random initialization of antibody population. The initial antibody population represented by binary-code string, including three parameters (σ, C, and ε) of a SVR model, is generated randomly. For example, assume that an antibody contains 12 binary codes to represent three SVR parameters. Each parameter is thus expressed by four binary codes. Therefore, for example, assume the set boundaries for parameters σ, C, and ε are 2, 10, and 0.5, respectively; then, the antibody with binary code "1 0 0 1 0 1 0 1 0 0 1 1" implies that the real values of the three parameters σ, C, and ε are 1.125, 3.125, and 0.09375, respectively. The number of initial antibodies is the same as the size of the memory cell. The size of the memory cell is set to 10 in this book.

Step 2: Identifying the affinity and the similarity. A higher affinity value implies that an antibody has a higher activation with an antigen. To maintain the diversity of the antibodies stored in the memory cells, the antibodies with lower similarity have higher probability of being included in the memory cell. Therefore, an antibody with a higher affinity value and a lower similarity value has a good likelihood of entering the memory cells. The affinity between the antibody and antigen is defined as Eq. (3.32):

$$\mathrm{Ag}_k = 1/(1 + d_k), \tag{3.32}$$

where d_k denotes the SVR forecasting errors obtained by the antibody k.

The similarity between antibodies is expressed as Eq. (3.33):

$$\mathrm{Ab}_{ij} = 1/(1 + T_{ij}), \tag{3.33}$$

where T_{ij} denotes the difference between the two SVR forecasting errors obtained by the antibodies inside (existed) and outside (will be entering) the memory cell.

Step 3: Selection of antibodies in the memory cell. Antibodies with higher values of Ag_k are considered to be potential candidates for entering the memory cell. However, the potential antibody candidates with Ab_{ij} values exceeding a certain threshold are not qualified to enter the memory cell. In this investigation, the threshold value is set to 0.9.

Step 4: Crossover and mutation of antibody population. New antibodies are created via crossover and mutation operations. To perform crossovers, strings representing antibodies are paired randomly. Moreover, segments of paired strings between two determined breakpoints are swapped. Mutations are performed randomly by converting a "1" code into a "0" code or a "0" code in to a "1" code. The crossover and mutation rates are determined using probabilities. In this investigation, the probabilities are set to 0.5 and 0.1 for crossover and mutation, respectively.

Step 5: Stopping criterion. If the number of generations equals a given scale, then the best antibody is a solution; otherwise, return to step 2.

The IA is used to seek a better combination of the three parameters in SVR. The value of the mean absolute percent error (MAPE) is used as the criterion (the smallest value of MAPE) of forecasting errors to determine the suitable parameters used in SVR model, which is given by Eq. (3.13).

3.9.2 IA for Three-Parameter Determination and Forecasting Results

This subsection will demonstrate the three-parameter determination of the proposed hybrid model (SVR with IA), namely, SVRIA model.

Similarly, in the training stage, the rolling-based forecasting procedure is also employed to obtain the forecasting load and receive training error in the training

Table 3.29 Parameter determination of SVRIA model

	Parameters			
Nos. of fed-in data	σ	C	ε	MAPE of testing (%)
5	758.12	409.33	3.7736	4.940
10	11.74	180.91	0.6728	4.079
15	43.21	2,367.70	13.5250	3.504
20	282.38	2,365.50	2.4397	3.880
25	**149.93**	**4,293.10**	**9.4790**	**3.211**

Table 3.30 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRCACO, SVRABC, and SVRIA models (unit, hundred million kWh)

Time point (month)	Actual	ARIMA (1,1,1)	GRNN ($\sigma = 3.33$)	TF-ε-SVR-SA	SVRCACO	SVRABC	SVRIA
Oct. 2008	181.07	192.932	191.131	184.504	180.876	184.498	181.322
Nov. 2008	180.56	191.127	187.827	190.361	182.122	183.372	181.669
Dec. 2008	189.03	189.916	184.999	202.980	184.610	183.323	183.430
Jan. 2009	182.07	191.995	185.613	195.753	185.233	183.549	183.964
Feb. 2009	167.35	189.940	184.397	167.580	185.274	183.774	184.030
Mar. 2009	189.30	183.988	178.988	185.936	184.247	183.999	182.829
Apr. 2009	175.84	189.348	181.395	180.165	184.930	183.420	183.463
MAPE (%)		6.044	4.636	3.799	3.371	3.458	3.211

stage. Then, if training error improvement occurs, the three kernel parameters, σ, C, and ε, of the SVRIA model adjusted by IA are employed to calculate the validation error. The adjusted parameters with minimum validation error are also selected as the most appropriate parameters. The forecasting results and the suitable parameters for the SVRIA model are illustrated in Table 3.29, in which it is also indicated that these two models all perform the best when 25 fed-in data are used.

For simplified comparison among alternative models, SVRGA, SVRSA, SVRGASA, and SVRPSO models are not considered due to their law forecasting accuracy levels. Table 3.30 shows the actual values and the forecast values obtained using various forecasting models: ARIMA(1,1,1), GRNN($\sigma = 3.33$), TF-ε-SVR-SA, SVRCACO, and SVRABC. The MAPE values are calculated to compare fairly the proposed models with other alternative models. The proposed SVRIA model has smaller MAPE values than other alternative models. Furthermore, to verify the significance of accuracy improvement of SVRIA model comparing with other alternative models, the Wilcoxon signed-rank test and asymptotic test are also conducted. The test results are shown in Tables 3.31 and 3.32, respectively. Clearly, the SVRIA model is almost significantly superior to other alternative models, except SVRABC model (only receives significance with $\alpha = 0.05$ level in Wilcoxon test, but all fails with both levels in asymptotic test). Figure 3.25 is provided to illustrate the forecasting accuracy among different models.

Table 3.31 Wilcoxon signed-rank test

	Wilcoxon signed-rank test	
	$\alpha = 0.025$	$\alpha = 0.05$
Compared models	$W = 2$	$W = 3$
SVRIA vs. ARIMA(1,1,1)	0[a]	0[a]
SVRIA vs. GRNN($\sigma = 3.33$)	2[a]	2[a]
SVRIA vs. TF-ε-SVR-SA	2[a]	2[a]
SVRIA vs. SVRCACO	1[a]	1[a]
SVRIA vs. SVRABC	3	3[a]

[a]Denotes that SVRIA model significantly outperforms other alternative models

Table 3.32 Asymptotic test

	Asymptotic (S_1) test	
Compared models	$\alpha = 0.05$	$\alpha = 0.10$
SVRIA vs. ARIMA(1,1,1)	H_0: $e_1 = e_2$ $S_1 = -9.143$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -9.143$; $p = 0.000$ (reject H_0)
SVRIA vs. GRNN($\sigma = 3.33$)	H_0: $e_1 = e_2$ $S_1 = -1.768$; $p = 0.03856$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -1.768$; $p = 0.03856$ (reject H_0)
SVRIA vs. TF-ε-SVR-SA	H_0: $e_1 = e_2$ $S_1 = -3.910$; $p = 0.000$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -3.910$; $p = 0.000$ (reject H_0)
SVRIA vs. SVRCACO	H_0: $e_1 = e_2$ $S_1 = -3.632$; $p = 0.00014$ (reject H_0)	H_0: $e_1 = e_2$ $S_1 = -3.632$; $p = 0.00014$ (reject H_0)
SVRIA vs. SVRABC	H_0: $e_1 = e_2$ $S_1 = 0.218$; $p = 0.4136$ (not reject H_0)	H_0: $e_1 = e_2$ $S_1 = 0.218$; $p = 0.4136$ (not reject H_0)

In this section, the IA is also employed to overcome the shortcomings of GA, SA, and PSO algorithms, that is, providing a set of solutions for exploration and exploitation of search space to obtain optimal/near optimal solutions by using immune system to find out the feasible solutions of the optimization problem. IA has such a mechanism to recognize all cells within the body and classify those cells as self or nonself; the nonself cells are categorized to the defensive mechanism for defending against foreign invaders; the lymphatic system contains two subclasses, *T*-cells and *B*-cells, to communicate with each other when an antigen enters the bloodstream. The proliferated *B*-cells turn into memory cells and produce antibodies (i.e., suitable parameter combination of a SVR model). The optimization problem (i.e., smaller forecasting error) is viewed as antigens. Conversely, the feasible solutions of the optimization problem are treated as antibodies (*B*-cells). Therefore, it is expected to receive better forecasting performance than SVRGA, SVRSA, and SVRPSO models and has the potential to provide some competitive

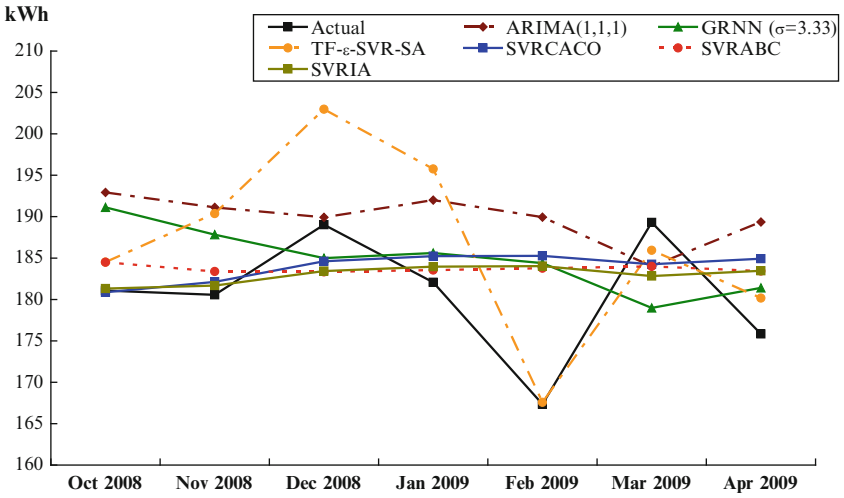

Fig. 3.25 Forecasting results of ARIMA, GRNN, TF-ε-SVR-SA, SVRCACO, SVRABC, and SVRIA models

solution comparing with SVRCACO and SVRABC models. However, it is also clear that SVRIA is not fitting the actual electric loads very well even if it has significant smaller MAPE values than other alternatives. Therefore, it also still requires hybridizing other novel techniques to improve this shortcoming.

References

1. Wang J, Zhu W, Zhang W, Sun D (2009) A trend fixed on firstly and seasonal adjustment model combined with the ε-SVR for short-term forecasting of electricity demand. Energ Policy 37:4901–4909. doi:10.1016/j.enpol.2009.06.046
2. Daniel WW (1978) Applied nonparametric statistics. Houghton Mifflin Co., Boston, MA
3. Diebold FX, Mariano RS (1995) Comparing predictive accuracy. J Bus Econ Stat 13:253–263. doi:10.1080/07350015.1995.10524599
4. Morgan WA (1939) A test for the significance of the difference between the two variances in a sample from a normal bivariate population. Biometrika 31:13–19. doi:10.1093/biomet/31.1-2.13
5. Granger CWJ, Newbold P (1977) Forecasting economic time series. Academic, Orlando, FL
6. Meese RA, Rogoff K (1988) Was it real? the exchange rate—interest differential relation over the modern floating-rate period. J Financ 43:933–948. doi:10.1111/j.1540-6261.1988.tb02613.x
7. Holland J (1975) Adaptation in natural and artificial system. University of Michigan Press, Ann Arbor, MI
8. Cercignani C (1988) The Boltzmann equation and its applications. Springer, Berlin
9. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH (1953) Equations of state calculations by fast computing machines. J Chem Phys 21:1087–1092. doi:10.1063/1.1699114
10. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680. doi:10.1126/science.220.4598.671
11. Van Laarhoven PJM, Aarts EHL (1987) Simulated annealing: theory and applications. Kluwer Academic, Dordrecht

12. Dekkers A, Aarts EHL (1991) Global optimization and simulated annealing. Math Program 50:367–393. doi:10.1007/BF01594945
13. Lee J, Johnson GE (1983) Optimal tolerance allotment using a genetic algorithm and truncated Monte Carlo simulation. Comput Aided Des 25:601–611. doi:10.1016/0010-4485(93)90075-Y
14. Shieh HJ, Peralta RC (2005) Optimal in situ bioremediation design by hybrid genetic algorithm-simulated annealing. J Water Resour Plan Manage-ASCE 131:67–78. doi:10.1061/(ASCE) 0733-9496(2005) 131:1(67)
15. Ponnambalam SG, Reddy MM (2003) A GA-SA multiobjective hybrid search algorithm for integrating lot sizing and sequencing in flow-line scheduling. Int J Adv Manuf Technol 21:126–137. doi:10.1007/s001700300015
16. Zhao F, Zeng X (2006) Simulated annealing—genetic algorithm for transit network optimization. J Comput Civil Eng 20:57–68. doi:10.1061/(ASCE)0887- 3801(2006) 20:1(57)
17. Cordón O, Moya F, Zarco C (2002) A new evolutionary algorithm combining simulated annealing and genetic programming for relevance feedback in fuzzy information retrieval systems. Soft Comput 6:308–319. doi:10.1007/s00500- 002-0184-8
18. Ganesh K, Punniyamoorthy M (2005) Optimization of continuous-time production planning using hybrid genetic algorithms-simulated annealing. Int J Adv Manuf Technol 26:148–154. doi:10.1007/s00170-003-1976-4
19. Wang ZG, Wong YS, Rahman M (2004) Optimisation of multi-pass milling using genetic algorithm and genetic simulated annealing. Int J Adv Manuf Technol 24:727–732. doi:10.1007/s00170-003-1789-5
20. Bergey PK, Ragsdale CT, Hoskote M (2003) A simulated annealing genetic algorithm for the electrical power districting problem. Ann Oper Res 121:33–55. doi:10.1023/A:1023347000978
21. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of IEEE international conference neural networks, Washington, DC, pp 1942–1948. doi:10.1109/ICNN.1995.488968
22. Eberhart RC, Shi Y (2001) Particle swarm optimization: developments, applications and resources. In: Proceedings of the 2001 congress on evolutionary computation, Seoul, South Korea, pp 81–86. doi:10.1109/CEC.2001.934374
23. Angeline PJ (1998) Evolutionary optimization versus particle swarm optimization: philosophy and performance differences. In: Proceedings of the 7th international conference on evolutionary programming, San Diego, CA, pp 601–610. doi:10.1007/BFb0040811
24. Liu B, Wang L, Jin YH, Tang F, Huang DX (2005) Improved particle swarm optimization combined with chaos. Chaos Soliton Fract 25:1261–1271. doi:10.1016/j.chaos.2004.11.095
25. Dorigo M (1992) Optimization, learning, and natural algorithms (Doctoral Dissertation), Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy
26. Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating ants. IEEE Trans Syst Man Cybern B Cybern 26:29–41. doi:10.1109/3477.484436
27. Colorni A, Dorigo M, Maniezzo V, Trubian M (1994) Ant system for job-shop scheduling. Belg J Oper Res Stat Comput Sci 34:39–53
28. Dorigo M, Gambardella L (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1:53–66. doi:10.1109/4235.585892
29. Bland JA (1999) Space-planning by ant colony optimization. Int J Comput Appl Technol 12:320–328. doi:10.1504/IJCAT.1999.000215
30. Maniezzo V, Colorni A (1999) The ant system applied to the quadratic assignment problem. IEEE Trans Knowl Data Eng 11:769–778. doi:10.1109/69.806935
31. Parpinelli RS, Lopes HS, Freitas AA (2002) Data mining with an ant colony optimization algorithm. IEEE Trans Evol Comput 6:321–332. doi:10.1109/TEVC. 2002.802452
32. Bilchev G, Parmee IC (1995) The ant colony metaphor for searching continuous design spaces. Lect Notes Comput Sci 993:25–39. doi:10.1007/3-540-60469-3_ 22
33. Mathur M, Karale SB, Priye S, Jyaraman VK, Kulkarni BD (2000) Ant colony approach to continuous function optimization. Ind Eng Chem Res 39:3814–3822. doi:10.1021/ie990700g

34. Wodrich M, Bilchev G (1997) Cooperative distributed search: the ant's way. Control Cybern 26:413–446
35. Monmarche N, Venturini G, Slimane M (2000) On how Pachycondyla apicalis ants suggest a new search algorithm. Futur Gener Comp Syst 16:937–946. doi:10.1016/S0167-739X(00)00047-9
36. Dreo J, Siarry P (2002) A new ant colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions. Lect Notes Comput Sci 2463:216–221
37. Socha K, Dorigo M (2008) Ant colony optimization for continuous domains. Eur J Oper Res 185:1155–1173. doi:10.1016/j.ejor.2006.06.046
38. Abbaspour KC, Schulin R, van Genuchten MT (2001) Estimating unsaturated soil hydraulic parameters using ant colony optimization. Adv Water Resour 24:827–841. doi:10.1016/S0309-1708(01)00018-5
39. Karaboga N, Kalinli A, Karaboga D (2004) Designing digital IIR filters using ant colony optimisation algorithm. Eng Appl Artif Intell 17(3):301–309. doi:10.1016/j.engappai.2004.02.009
40. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39(3):459–471. doi:10.1007/s10898-007-9149-x
41. Karaboga D, Basturk B (2007) Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. Lect Notes Comput Sci 4529:789–798. doi:10.1007/978-3-540-72950-1_77
42. Karaboga D, Basturk B (2008) On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput 8:687–697. doi:10.1016/j.asoc.2007.05.007
43. Xu C, Duan H, Liu F (2010) Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning. Aerosp Sci Technol 14:535–541. doi:10.1016/j.ast.2010.04.008
44. Fathian M, Amiri B, Maroosi A (2007) Application of honey bee mating optimization algorithm on clustering. Appl Math Comput 190:1502–1513. doi:10. 1016/j.amc.2007.02.029
45. Özbakir L, Baykasoglu A, Tapkan P (2010) Bees algorithm for generalized assignment problem. Appl Math Comput 215:3782–3795. doi:10.1016/j.amc. 2009.11.018
46. Tereshko V, Loengarov A (2005) Collective decision making in honey-bee foraging dynamics. Comput Inform Syst 9(3):1–7
47. Mori K, Tsukiyama M, Fukuda T (1993) Immune algorithm with searching diversity and its application to resource allocation problem. Trans Inst Electr Eng Jpn 113-C:872–878
48. Prakash A, Khilwani N, Tiwari MK, Cohen Y (2008) Modified immune algorithm for job selection and operation allocation problem in flexible manufacturing system. Adv Eng Softw 39:219–232. doi:10.1016/j.advengsoft. 2007.01.024