



A novel CACO_R-SVR multi-objective optimization approach and its application in aerodynamic shape optimization of high-speed train

Ye Zhang¹ · Guo Wei Yang^{2,3} · Di Long Guo² · Zhen Xu Sun² · Da Wei Chen¹

© Springer-Verlag GmbH Germany, part of Springer Nature 2018

Abstract

A chaos ant colony optimization algorithm for continuous domain is proposed based on chaos optimization theory and ant colony optimization algorithm. The searching abilities of optimization algorithms with different coding methods are compared, and the results indicate that the proposed algorithm has better performance than genetic algorithm and particle swarm optimization algorithm. Based on the non-dominated sorting concept and niching method, a multi-objective chaos ant colony optimization algorithm is also constructed and numerical results show that the improved algorithm performs well at solving multi-objective optimization problems. An optimal support vector regression model based on radial basis kernel function is developed for the small sample size and nonlinear characteristics of streamlined head optimization. On the basis of the above work, a multi-objective optimization design for the aerodynamic head shape of high-speed train is developed using a modified vehicle modeling function parametric approach. The optimization results demonstrate that the new optimization design method has exceptional searching abilities and high prediction accuracy. After optimization, the aerodynamic drag of the simplified train with three carriages is reduced by 10.52% and the aerodynamic lift of the tail car is reduced by 35.70%. The optimization approach proposed in the present paper is simple yet efficient and sheds light on the engineering design of aerodynamic shape of high-speed trains.

Keywords Chaos ant colony optimization · Support vector machine · Multi-objective optimization · Vehicle modeling function · High-speed trains

1 Introduction

In recent years, metaheuristics have been improved rapidly and have been applied to numerous problems in fields of engineering design, pattern recognition, scientific computation, economics, and so on. Different kinds of biological-inspired algorithms have been proposed to solve optimization problems with single objective or multiple objectives. For single-objective optimization problems, genetic algorithm (GA) (Holland 1975), particle swarm optimization (PSO)

(Kennedy and Eberhart 1995) and ant colony optimization (ACO) algorithm (Dorigo 1992) are some of the competitive algorithms that attract a lot of attention. The above basic algorithms have good performance over many optimization problems; however, these algorithms have deficiencies of low searching efficiency and local optimum in optimizing some problems. Therefore, a number of variations have been developed to improve performance of the basic algorithms. Prominent examples of the variations include AGA (Srinivas and Patnaik 1994), HGA (Gonçalves et al. 2005), CLPSO (Liang et al. 2006), APSO (Zhan et al. 2009) and ACO_R (Socha and Dorigo 2008). However, when devising optimization models for a problem, it is frequently the case that there is not one but several objectives that we would like to optimize. These problems with two or more objective functions are called multi-objective optimization problems (MOP). There are two general approaches to multiple-objective optimization. One is to combine the individual objective functions into a single composite function or move all but one objective to the constraint set. The second general approach is to deter-

Communicated by V. Loia.

✉ Guo Wei Yang
gwyang@imech.ac.cn

¹ CRRC Qingdao Sifang Co., LTD, Qingdao 266111, China

² Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China

³ School of Engineering Sciences, University of Chinese Academy of Sciences, Beijing 100049, China

mine an entire Pareto optimal solution set or a representative subset (Coello 2006; Konak et al. 2006). The most representative multi-objective optimization algorithms include NPGA (Horn et al. 1994), NSGA-II (Deb et al. 2002) and NSPSO (Li 2003).

In the field of engineering optimization design, algorithms are usually combined with surrogate models to reduce the design cost (so-called surrogate-based optimization, SBO). In surrogate-based optimization, direct evaluations of the expensive high-fidelity simulation models are replaced by function approximation surrogate models which can be created by approximating sampled high-fidelity model data. Therefore, the design cost can be greatly reduced. SBO methods differ mostly in the optimization algorithm it adopted and how the surrogate model is created. The most popular surrogate modeling techniques include polynomial approximation, neural networks (Lee et al. 2011), Kriging (Simpson et al. 2001), radial basis function (RBF) interpolation (Forrester and Keane 2009) and support vector regression (SVR) (Smola and Schölkopf 2004; Bourinet 2016). During the modeling process, a certain number of simulation data training samples are required to ensure the reasonable accuracy. Currently, SBO methods have received more and more attention for the reason that it has shorter design cycle and lower investment than traditional design method (Koziel and Leifsson 2012; Koziel et al. 2014; Datta and Regis 2016). In the past decades, various engineering optimization design based on surrogate models has been performed. Bellman et al. (2009) employed a GA-ANN optimization technique for shape optimization of low Reynolds number airfoils to generate maximum lift. Muñoz-Paniagua et al. (2014) extracted three design variables and conducted the shape optimization of a high-speed train entering a tunnel with the unconstrained single-objective genetic algorithm. Lee and Geem (2005) proposed a new meta-heuristic algorithm and applied it into pressure vessel design and welded beam design.

In the present paper, on the basis of the above literature, a chaos ant colony optimization algorithm for continuous domain (CACO_R) is proposed based on chaos optimization theory and ACO_R. Meanwhile, an optimal SVR model is established for the reason that it performs well at response accuracy when the number of training samples is limited. Combined with CACO_R and SVR surrogate model, a multi-objective efficient optimization method CACO_R-SVR is developed. In order to verify the efficiency of the CACO_R-SVR approach in engineering design problems, aerodynamic shape optimization for high-speed trains is performed using CACO_R-SVR. The aerodynamic drag and aerodynamic lift are treated as optimization objectives and a modified vehicle modeling function (VMF) (Ku et al. 2010) is devised to generate 3-D nose shapes of high-speed trains. After optimization, the Pareto optimal solutions are obtained and the aerodynamic performance of the optimized shape and the

original shape of high-speed train with three carriages is comparatively analyzed.

2 Chaos ant colony optimization algorithm

2.1 General ant colony optimization algorithm

The ant colony optimization algorithm is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. It was initially proposed by Dorigo in 1992, and it has become a promising approach in solving a wider class of numerical issues. The first algorithm was used to solve the traveling salesman problem (TSP), a well-known NP-Hard problem. While searching for food, the ants deposit a chemical substance called pheromone in the ground and they will return and reinforce the pheromone trail if they eventually find food. Meanwhile, the pheromones evaporate over time and the more time it takes for the ant to travel from the nest to the food source, the more time the pheromones have to evaporate. As a result, a short path gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than the longer ones. If the path has a large concentration of pheromone, this is probably due to its shorter length that allowed ants to travel faster, resulting in a larger number of ants depositing pheromone on it. Ants are more likely to follow the path that contains more pheromones, and this positive feedback mechanism will lead to all the ants following a single path eventually. The mathematical model of ACO is given by:

$$P_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}^\alpha(t)\eta_{ij}^\beta(t)}{\sum_{s \in \text{allowed}_k} \tau_{is}^\alpha(t)\eta_{is}^\beta(t)}, & j \in \text{allowed}_k \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where $P_{ij}^k(t)$ is the probability of ant k , positioned at city i , traveling to city j at the moment t . The pheromone quantity is given by τ_{ij} , η_{ij} is the heuristic information of the problem and is given by the inverse of the distance between city i and city j : $\eta_{ij} = 1/d_{ij}$. allowed_k is the set of cities not yet visited by ant k while at city i , $\alpha > 0$ and $\beta > 0$ are parameters weighting the relative importance of the pheromone and of the heuristic information, respectively. The pheromone evaporation is performed according to the following update function:

$$\begin{cases} \tau_{ij}(t + \Delta t) = \rho \cdot \tau_{ij}(t) + \Delta \tau_{ij} \\ \Delta \tau_{ij} = \sum_{k=1}^m \Delta \tau_{ij}^k, \end{cases} \quad (2)$$

where $\rho \in [0, 1]$ is the persistence parameter for the pheromone. $\Delta \tau_{ij}^k$ represents the pheromone quantity to be deposited by ant k , and it is given by:

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{F^k(t)} & \text{if arc } (i, j) \text{ used by ant } k, \\ 0 & \text{otherwise,} \end{cases} \quad (3)$$

where Q is a positive proportionality parameter and $F^k(t)$ is the length of the tour constructed by ant k at iteration t .

According to the existing references (Socha and Dorigo 2008; Liu et al. 2014), general ant colony algorithm can be applied to continuous domain by either discretizing the continuous domain into several regions or shifting from using a discrete probability distribution to using a continuous one such as a Gaussian probability density function (PDF). The research results show that probability density function could improve searching efficiency significantly compared with basic ACO. As a result, in the present paper, the ACO_R approach is adopted for continuous optimization.

For a D -dimensional real-valued minimization problem, ACO_R initializes k solutions at random firstly: $S_j^i (i = 1, 2, 3, \dots, D; j = 1, 2, 3, \dots, k)$ and the k solutions are kept sorted according to their objective values (from minimum to maximum) and each solution S_j has a weight ω_j which is defined by Gaussian function:

$$\omega_j = \frac{1}{qk\sqrt{2\pi}} e^{-\frac{(\text{rank}(j)-1)^2}{2q^2k^2}}, \quad (4)$$

where $\text{rank}(j)$ is the rank of S_j in the sorted archive and q is a positive parameter of the algorithm. Obviously, $\omega_1 \geq \omega_2 \geq \dots \omega_j \geq \dots \omega_k$. The probability of choosing solution S_j as guiding solution is given by:

$$P_j = \frac{\omega_j}{\sum_{r=1}^k \omega_r}. \quad (5)$$

Therefore, the better solution has higher chances to be chosen as the guiding solution. Population updating is accomplished by generating a certain number of new solutions near S_{guide} and adding them to the solution set; meanwhile, remove the same number of worst solutions so that the size of solution set does not change. However, the mechanism of pheromone updating tends to induce a potential problem of trapping into local optimum in case that S_{guide} is near the local minimum point. In the present paper, a chaotic disturbance is introduced into the algorithm to avoid this problem.

2.2 Construction of CACO_R algorithm

Chaos is a kind of universal nonlinear phenomena in many areas of science. It is highly sensitive to the changes in initial conditions that small differences in initial conditions yield widely diverging outcomes for the system. Chaos theory studies the behavior of systems that follow deterministic laws but appear random and unpredictable (May 1976; Cong

et al. 2010). There are three main properties of the chaotic map: randomness, ergodicity and regularity. The ergodicity property of chaos can ensure chaotic variables to go through every state in certain scale. As a result, it could be introduced into the optimization strategy to avoid falling into local minimum solution. Meanwhile, the randomness and ergodicity can accelerate the optimum seeking operation (Ikeguchi et al. 2011).

Chaotic behavior can arise from very simple nonlinear dynamical models. The logistic map is generally used in chaos optimization algorithm, and it was popularized by the biologist Robert May in 1976. The map is written by:

$$X_{n+1} = L(\mu, X_n) = \mu X_n(1 - X_n), n = 0, 1, 2, \dots \quad (6)$$

Here the initial value of X is chosen between 0 and 1, and $\mu \in [0, 4]$ is the control parameter. In theory, X_n will traverse all values non-repeatedly in $[0, 1]$ in condition that n is large enough. When $n = 1000$, the initial value of X is 0.6, the randomness and ergodicity for logistic map are shown in Fig. 1. By varying the parameter μ , different behavior of the dynamic model can be observed, and when $\mu = 4.0$, the system is in the state of complete chaos. Figure 2 illustrates the amplitude and frequency content of some logistic map iterates in phase space.

The CACO_R is proposed based on the problem that traditional ACO is easy to fall in local best, and its searching speed is slow in some special optimization problems. The main idea of CACO_R is to generate the initial ant colony by the logistic map and update the population with chaotic disturbance. For a D -dimensional real-valued minimization problem, the initial colony is given by:

$$\text{Ant } j : \begin{cases} X_i = \mu_1 X_{i-1}(1 - X_{i-1}), i = 1, 2, \dots, D \\ S_j^i = (b_i - a_i) X_i + a_i, j = 1, 2, \dots, k, \end{cases} \quad (7)$$

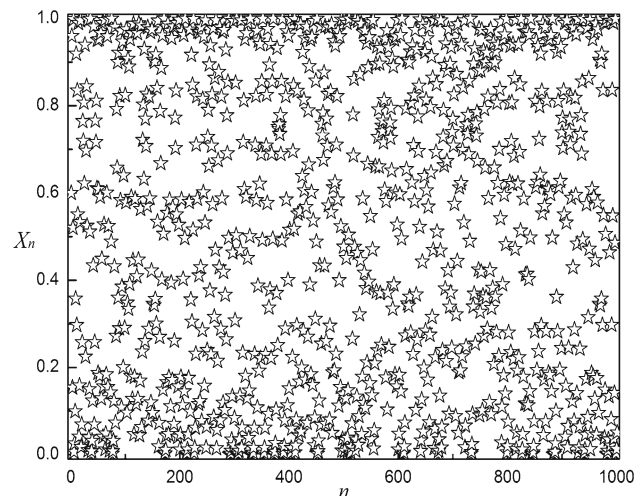


Fig. 1 Randomness and ergodicity for logistic map

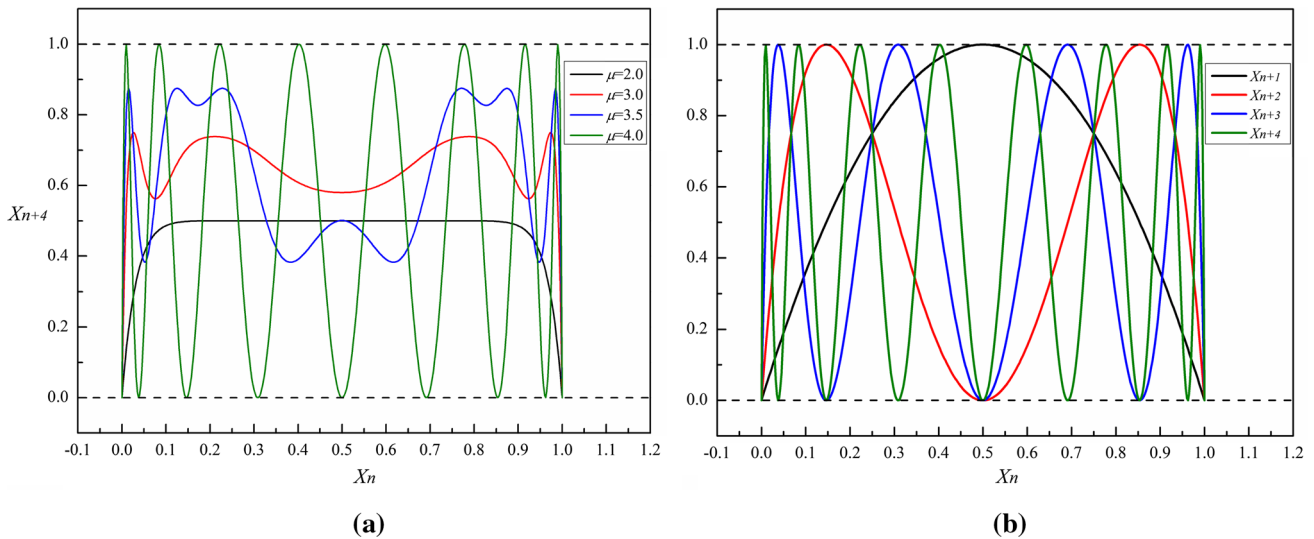


Fig. 2 **a** Logistic map for different parameter μ in phase space. **b** Logistic map with 4 iterations ($\mu = 4.0$) in phase space

where S_j^i is the component of \mathbf{S}_j , the i -th component definitional domain of \mathbf{S}_j is $[a_i, b_i]$. X_i is the logistic map with $\mu_1 = 4.0$ and $X_0 = 0.6$, and k is the population number of ant colony. Hence, a randomly generated initial population is obtained. When updating the population at each iteration, sort all the individuals according to their quality (from best to worst) and calculate the weight ω_j for each solution \mathbf{S}_j . Choose the individual with highest ω_j as the guiding solution $\mathbf{S}_{\text{guide}}$, and then, remove m ($m < k$) worst solutions and generate the same number of new solutions by introducing the chaos perturbation to the current population:

$$\begin{cases} S_l^i = \rho \cdot S_{\text{guide}}^i + \lambda (S_{\text{guide}}^i - a_i) Y_l, & i = 1, 2, \dots, D \\ Y_l = \mu_2 Y_{l-1} (1 - Y_{l-1}), & l = 1, 2, \dots, m, \end{cases} \quad (8)$$

where $\rho \in [0, 1]$ represents how much the new individual inherited from the guiding solution, $\lambda \in [0, 1]$ is the coefficient related to the chaos perturbation quantity, and Y_l is the logistic map with $\mu_2 = 4.0$ and $Y_0 = 0.5$. In general, the CACO_R is described as follows:

Step 1 Initialize the ant colony: generate a certain number of ants with logistic map and compute objective values of every ant. Initialize the parameters of the program, set the value of stop iteration number N and weighting coefficients ρ and λ .

Step 2 Sort the solutions by objective values, calculate ω_j , P_j and choose the guiding solution $\mathbf{S}_{\text{guide}}$. Save $\mathbf{S}_{\text{guide}}$ and the corresponding best value J_{best} of the current generation.

Step 3 Generate m new solutions using chaotic theory according to equation (8) and remove the same number of worst solutions. Calculate the current objective values of every solution.

Step 4 Compute the difference of best values between neighboring generations and check the current iteration number n , if the precision or iteration number is not be satisfied, then loop *Step 2–Step 4*.

Step 5 If the termination conditions are satisfied, stop iteration and put out the best solution.

2.3 Single-objective optimization test

To examine the performance of the proposed CACO_R algorithm, 5 test functions are adopted in this paper; meanwhile, the CACO_R algorithm is compared with the standard GA and PSO, the original ACO_R and the CLPSO algorithms. All the test functions are minimization problems, as shown in Table 1.

Perform three optimizing searches to find out the minimum solution of the functions in Table 1 and each algorithm was run by 30 times. Besides, the mean, max and min (best) value of the solution are calculated. In GA algorithm, the population size is set to 100, crossover rate is set to 0.75, and mutation rate is set to 0.2. For PSO algorithm, 100 particles are generated and the acceleration coefficients c_1 and c_2 are both set to 2.0. Meanwhile, linearly decreasing inertia weight ω is adopted and the maximal and minimal weights are set to 0.9 and 0.4. For CLPSO, algorithm, we adopt the

learning probability $P_{c_i} = 0.05 + 0.45 \times \frac{\exp(\frac{5(i-1)}{S-1}) - 1}{\exp(5) - 1}$. In the proposed CACO_R algorithm and the original ACO_R algorithm, ant colony population is set to 30, parameter ρ is set to 0.9. The default number of generation N of all algorithms is set to 200. The numerical results and the comparison of performance differences between traditional GA, PSO, ACO_R, CLPSO and CACO_R are summarized in Table 2.

Table 1 Test functions

Function	Mathematical representation	Range	Modality	Optimum
De Jong F1	$f_1(x) = \sum_{i=1}^n x_i^2$	$x_i \in (-5.12, 5.12)$	Unimodal	0
Rosenbrock	$f_2(x) = \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$	$x_i \in (-10, 10)$	Unimodal	0
Shubert	$f_3(x, y) = \left\{ \sum_{i=1}^n i \cos[(i+1)x + 1] \right\} \left\{ \sum_{i=1}^n i \cos[(i+1)y + 1] \right\}$	$x, y \in (-10, 10)$	Multimodal	-186.730909
Humpback	$f_4(x) = (4 - 2.1x^2 + x^4/3)x^2 + xy + (-4 + 4y^2)y^2$	$x \in (-3, 3), y \in (-2, 2)$	Multimodal	-1.031628
Rastrigin	$f_5(x) = \sum_{i=1}^n (x_i^2 - 10 \cos 2\pi x_i + 10)$	$x_i \in (-5.12, 5.12)$	Multimodal	0

Table 2 Numerical results

Function	Criterion	GA	PSO	ACOR	CLPSO	CACOR
$f_1(x)$	Mean	0.000013	0.000000	0.000000	0.000000	0.000000
	Min	0.000003	0.000000	0.000000	0.000000	0.000000
	Max	0.000028	0.000000	0.000000	0.000000	0.000000
$f_2(x)$	Mean	0.000411	0.000134	0.000000	0.000000	0.000000
	Min	0.000027	0.000007	0.000000	0.000000	0.000000
	Max	0.000962	0.000415	0.000000	0.000000	0.000000
$f_3(x)$	Mean	-186.730812	-186.730894	-186.730881	-186.730907	-186.730909
	Min	-186.730900	-186.730909	-186.730909	-186.730909	-186.730909
	Max	-186.730761	-186.730888	-186.730797	-186.730900	-186.730909
$f_4(x)$	Mean	-1.031614	-1.031628	-1.031628	-1.031628	-1.031628
	Min	-1.031623	-1.031628	-1.031628	-1.031628	-1.031628
	Max	-1.031600	-1.031628	-1.031628	-1.031628	-1.031628
$f_5(x)$	Mean	0.000123	0.000005	0.000344	0.000000	0.000000
	Min	0.000008	0.000000	0.000109	0.000000	0.000000
	Max	0.000634	0.000028	0.000795	0.000001	0.000000

It can be concluded that the CACOR algorithm gets a better performance at optimization precision and reliability compared to GA, PSO, original ACOR and CLPSO algorithms. Furthermore, the original ACOR was trapped into the local optimum when solving the Rastrigin function while the CACOR algorithm could get the correct minimum. Figure 3 gives two examples of convergence curves ($f_1(x), f_2(x)$) for different algorithms in logarithmic coordinate. Obviously, the convergence rate of the CACOR is faster than the other algorithms.

2.4 Non-dominated sorting technique and niching method

When applied to multi-objective optimization problems, the main difficulties come from how to obtain a set of non-dominated solutions as closely as possible to the true Pareto front and to maintain a well-distributed solution set along the Pareto front. The non-dominated sorting concept which originated from NSGA-II (Deb et al. 2002) is widely used in

multi-objective optimization algorithms. It solves the problem of selecting the better individuals by sorting the entire population into different non-domination levels. Hence, it is introduced into the CACOR algorithm to obtain a series of solutions on the Pareto front. Besides, to preserve diversity in the final group of solutions, the niching method (Li 2003) is adopted in the present paper. Niching methods have been successfully used in GA and PSO algorithms, Goldberg and Richardson (1987) proposed a sharing function model for GA and Li (2003) introduced the σ_{share} distance into NSPSO. Figure 4 shows the non-dominated sorting concept and niche counts method.

The dynamically updated σ_{share} is defined as follows:

$$\sigma_{share} = \frac{\sum_{i=1}^m (u_i - l_i)}{n - 1}, \tag{9}$$

where u_i and l_i are the upper and lower bounds for every objective value, n is the size of the ant population, and

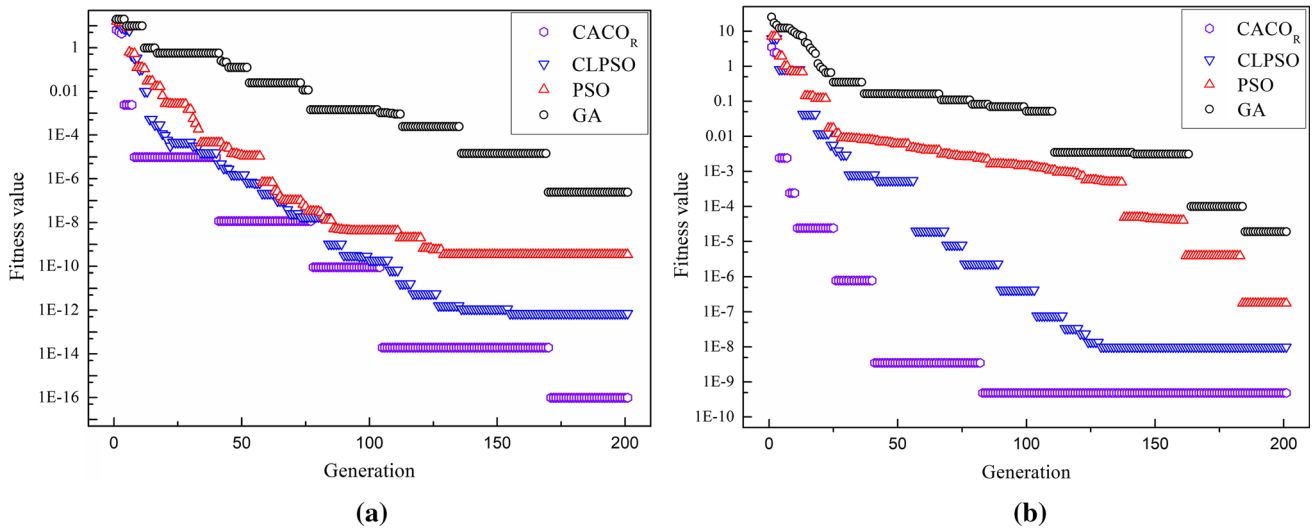


Fig. 3 **a** Convergence curves for $f_1(x)$, **b** convergence curves for $f_2(x)$

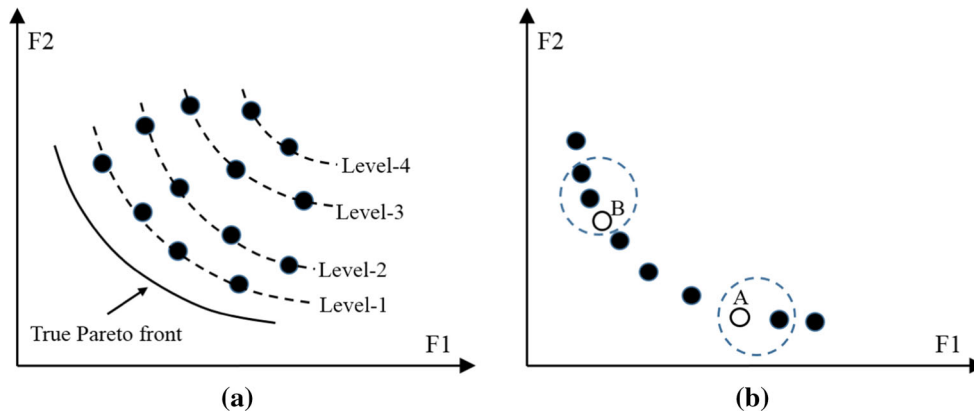


Fig. 4 **a** Non-dominated fronts of different levels, **b** niche counts for two-objective optimization

m is the number of objectives. Calculate the niche count of each candidate solution within the σ_{share} distance, and select the one with smaller niche count as a new front solution. Obviously, A will be preferred over B in Fig. 4b. In addition, to illustrate the key steps of the algorithm, Fig. 5 shows the construction process of the multi-objective CACO_R.

2.5 Multi-objective optimization test

In order to verify the multi-objective optimization efficiency of the algorithm, two test functions are used. Test1 function was proposed by Kursawe (1991), and Test2 function was proposed by Zitzler et al. (2000), these functions have been widely used in algorithm testing because of disconnectedness of Pareto front and multiple local fronts.

$$\text{Test1: } \begin{cases} f_1(\mathbf{x}) = \sum_{i=1}^2 [-10 \exp(-0.2 \sqrt{x_i^2 + x_{i+1}^2})] \\ f_2(\mathbf{x}) = \sum_{i=1}^3 (|x_i|^{0.8} + 5 \sin(x_i^3)) \end{cases} \quad (10)$$

where $-5 \leq x_i \leq 5, i = 1, 2, 3$.

$$\text{Test2: } \begin{cases} f_1(\mathbf{x}) = x_1 \\ f_2(\mathbf{x}) = g(\mathbf{x}) h(f_1(\mathbf{x}), g(\mathbf{x})) \\ g(\mathbf{x}) = 1 + \frac{9}{29} \sum_{i=2}^{30} x_i \\ h(f_1(\mathbf{x}), g(\mathbf{x})) = 1 - \sqrt{\frac{f_1(\mathbf{x})}{g(\mathbf{x})}} \\ \quad - \left(\frac{f_1(\mathbf{x})}{g(\mathbf{x})}\right) \sin(10\pi f_1(\mathbf{x})) \end{cases} \quad (11)$$

where $0 \leq x_1, x_2 \leq 1, 1 \leq i \leq 30$.

For all test functions, the initial ant population is 150 and the number of iterations is 500. Figure 6 shows comparison between the true Pareto front and CACO_R front for each

Fig. 5 Flowchart of the multi-objective CACO_R algorithm

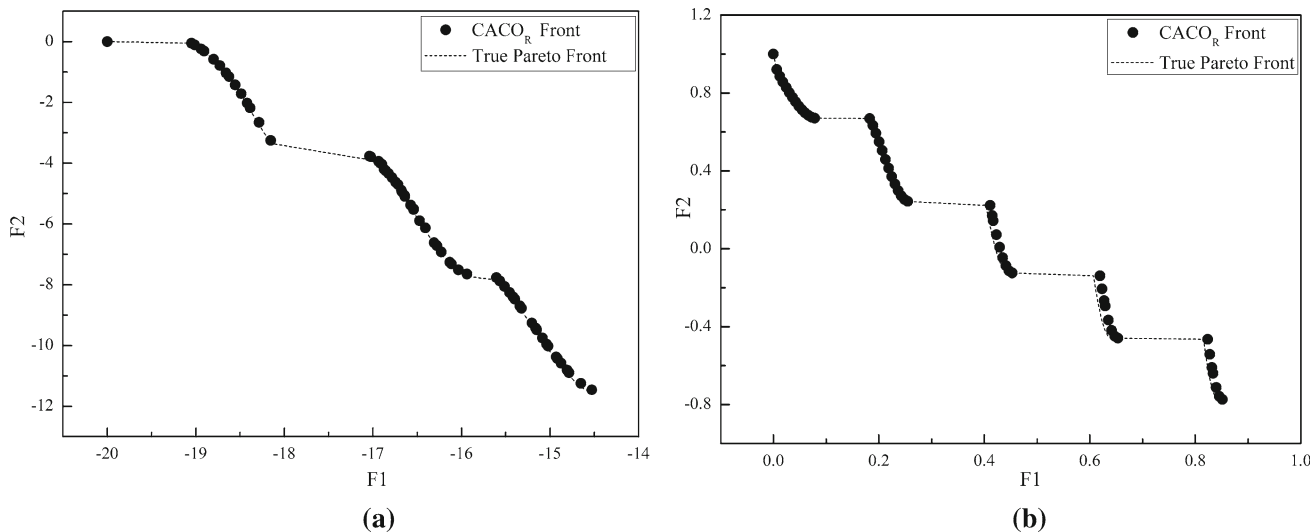
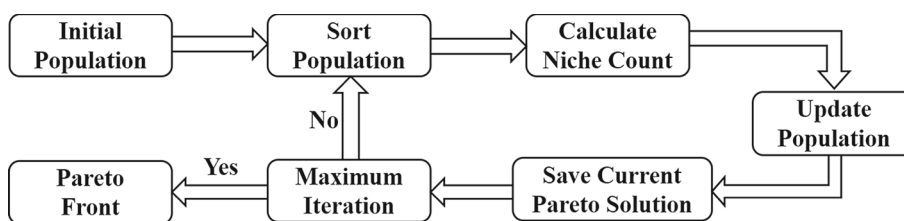


Fig. 6 **a** Pareto solution of Test 1, **b** Pareto solution of Test 2

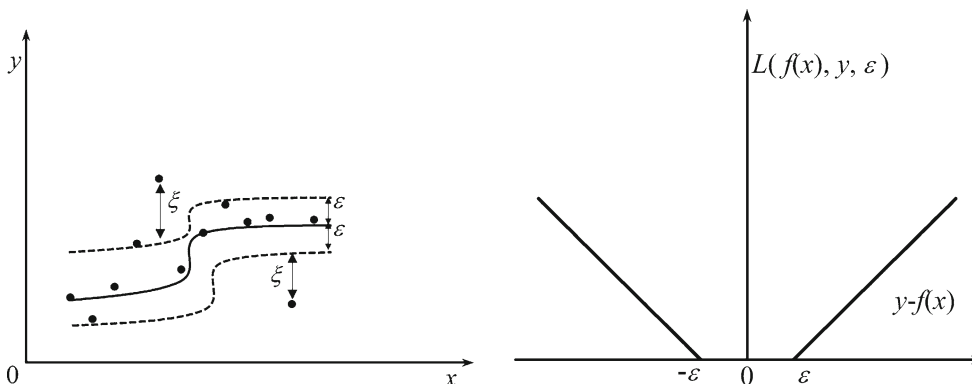


Fig. 7 Soft margin loss setting for a nonlinear SVR

test function, and both results illustrate good multi-objective search ability of the improved algorithm.

3 Support vector regression

Support vector machine (SVM), which is based on statistical learning theory and characterized by usage of kernel tricks and support vectors, is a standard classification technique that has been widely used in many classification problems. It was

developed by Vapnik in 1995 and has been extensively developed during the past two decades (Vladimir and Vapnik 1995; Iosifidis and Gabbouj 2016). The main idea of the SVM is to minimize classification errors by maximizing the margin between the separating hyperplane and the data sets. Smola and Vapnik (1997) developed the SVR method to solve non-linear regression problems by introducing an ϵ -insensitive loss function into SVM. A number of studies have indicated that SVR has advantages of high-dimensional input space, few irrelevant features and good accuracy. As a result, SVR has been successfully applied to various problems: regres-

sion estimation (Bourinet 2016), signal processing, pattern recognition, medical diagnosis, forecasting problems, etc.

3.1 Construction of SVR model

The training set is defined as $\{(\mathbf{x}_i, y_i), i = 1, 2, \dots, l\}$, where l is the number of samples, $\mathbf{x}_i (\mathbf{x}_i \in \mathbf{R}^d)$ is the i th input, $\mathbf{x}_i = [x_i^1, x_i^2, \dots, x_i^d]^T$, d is the dimension of input space and $y_i \in \mathbf{R}$ is the i th output, correspondingly. Kernel trick uses a classifier algorithm to solve a nonlinear problem by mapping data from the input space into a higher-dimensional space. Define the linear discriminant function

$$f(\mathbf{x}) = \mathbf{w}\Phi(\mathbf{x}) + b, \tag{12}$$

where $\Phi(\mathbf{x})$ is the nonlinear mapping function and (\mathbf{w}, b) is the functional margin. Define the ε -insensitive loss function

$$L(f(\mathbf{x}), y, \varepsilon) = \begin{cases} 0, & |y - f(\mathbf{x})| \leq \varepsilon \\ |y - f(\mathbf{x})| - \varepsilon, & |y - f(\mathbf{x})| > \varepsilon \end{cases} \tag{13}$$

where y is the true value and $f(\mathbf{x})$ is the corresponding prediction. Analogously to the soft margin loss function which was adapted to SVM, positive-valued slack variables ξ_i, ξ_i^* can be introduced and the problem of searching (\mathbf{w}, b) can be written as an optimization problem:

$$\begin{cases} \min \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^l (\xi_i + \xi_i^*), & C > 0 \\ \text{s. t.} \begin{cases} y_i - \mathbf{w}\Phi(\mathbf{x}_i) - b \leq \varepsilon + \xi_i, & i = 1, 2, \dots, l \\ -y_i + \mathbf{w}\Phi(\mathbf{x}_i) + b \leq \varepsilon + \xi_i^* \\ \xi_i \geq 0, \xi_i^* \geq 0. \end{cases} \end{cases} \tag{14}$$

Parameter C determines the trade-off between the model complexity (flatness) and the amount to which deviations larger than ε are tolerated in optimization formulation. If C is too large, then the objective is to minimize the empirical risk only, without regard to model complexity part in the optimization formulation. Parameter ε controls the width of the ε -insensitive zone, used to fit the training data, as shown in Fig. 7.

The main idea to solve the above optimization problem is to construct a Lagrange function from the objective function and the corresponding constraints, by introducing a dual set of variables:

$$L := \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^l (\xi_i + \xi_i^*) - \sum_{i=1}^l (\eta_i \xi_i + \eta_i^* \xi_i^*) - \sum_{i=1}^l \alpha_i (\varepsilon + \xi_i - y_i + \mathbf{w}\Phi(\mathbf{x}_i) + b)$$

$$- \sum_{i=1}^l \alpha_i^* (\varepsilon + \xi_i^* + y_i - \mathbf{w}\Phi(\mathbf{x}_i) - b) \tag{15}$$

where $\alpha_i, \eta_i, \alpha_i^*, \eta_i^* \geq 0$ are Lagrange multipliers. The saddle point condition is given by:

$$\begin{cases} \partial_b L = \sum_{i=1}^l (\alpha_i^* - \alpha_i) = 0 \\ \partial_{\mathbf{w}} L = \mathbf{w} - \sum_{i=1}^l (\alpha_i - \alpha_i^*) \Phi(\mathbf{x}_i) = 0 \\ \partial_{\xi_i} L = C - \alpha_i - \eta_i = 0 \\ \partial_{\xi_i^*} L = C - \alpha_i^* - \eta_i^* = 0. \end{cases} \tag{16}$$

Eliminate the dual variables η_i, η_i^* and we can get:

$$\begin{cases} \max_{\alpha, \alpha^*} \left[-\frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l (\alpha_i - \alpha_i^*) (\alpha_j - \alpha_j^*) K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^l (\alpha_i + \alpha_i^*) \varepsilon + \sum_{i=1}^l (\alpha_i - \alpha_i^*) y_i \right] \\ \text{s. t.} \begin{cases} \sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0 \\ 0 \leq \alpha_i \leq C \\ 0 \leq \alpha_i^* \leq C, \end{cases} \end{cases} \tag{17}$$

where $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \Phi(\mathbf{x}_j)$ is the kernel function. Define N_{nsv} as the number of support vectors and the solution can be written as:

$$\begin{cases} \mathbf{w}^* = \sum_{i=1}^l (\alpha_i - \alpha_i^*) \Phi(\mathbf{x}_i) \\ b^* = \frac{1}{N_{nsv}} \\ \times \left\{ \sum_{0 < \alpha_i < C} y_i - \left[\sum_{\mathbf{x}_i \in SV} (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}_j) - \varepsilon \right] \right. \\ \left. + \sum_{0 < \alpha_i^* < C} y_i - \left[\sum_{\mathbf{x}_j \in SV} (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}_j) + \varepsilon \right] \right\}. \end{cases} \tag{18}$$

Finally, we get the regression function

$$f(\mathbf{x}) = \mathbf{w}^* \Phi(\mathbf{x}) + b^* = \sum_{i=1}^l (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}_j) + b^*. \tag{19}$$

It is well known that SVR performance (estimation accuracy) depends on a good setting of parameters C, ε and the kernel parameters. Selecting a particular kernel type and kernel function parameters is usually based on characteristics of the training data. Common kernel functions include:

D-th-order polynomial function:

$$K(x_i, x_j) = (x_i \cdot x_j)^d \tag{20}$$

Gaussian radial basis function (RBF):

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2) \tag{21}$$

Hyperbolic tangent function (Sigmoid):

$$K(x_i, x_j) = \tanh(\gamma x_i \cdot x_j + c). \tag{22}$$

3.2 Regression accuracy analysis

According to existing studies, RBF kernel function is widely used in SVM classification and regression for the reason that it has better generalization performance. As a result, RBF kernel function is adopted in the present paper. Meanwhile, SVR model is sensitive to the input parameters and only obtain suitable parameters can the model perform well enough. Hence, the input parameters are optimized with CACOR before use. Figure 8 shows the regression and prediction test results of SVR mode, and Table 3 gives the mean squared error (mse) and squared correlation coefficient (R²) for different cases. Here $f_1(x)$ is an exponential function, $f_2(x)$ is a polynomial function, $f_3(x)$ is a two-dimensional projection transform of Ackley function:

$$f_3(x) = -20 \exp\left(-0.2\sqrt{0.5x^2}\right) - \exp(0.5 \cos(2\pi x)) + e + 20. \tag{23}$$

And $f_4(x)$ is Schwefel function:

$$f_4(x) = 418.9829 - x \sin(\sqrt{|x|}). \tag{24}$$

It can be concluded from Fig. 8 and Table 3 that SVR model constructed with RBF kernel is sufficiently accurate.

4 Optimization of high-speed train nose shape

Optimization design of aerodynamic shape is a research focus in the field of fluid mechanics, especially in the design of aircraft. In recent years, optimization of the streamlined head of high-speed trains has attracted more and more attention as the aerodynamic problems become more serious with the increasing running speed. The aerodynamic drag of high-speed trains can be up to 80% of the total drag (Yang et al. 2012) at the speed of 300 km/h. Resistance characteristics of the trains are directly related to the ability of energy saving and environmental protection (Raghunathan et al. 2002;

Baker 2010; Tian 2009). In aerodynamic lift research, the wheel-track force is significantly reduced while excessive positive lift act on the train, which affects the operation safety. Meanwhile, negative aerodynamic lift increases the dynamic axle load and aggravates rail abrasion. For high-speed maglev trains, the time difference between the change in electromagnetic suspension force and aerodynamic lift causes server vibration and affects the amenity of the train. Therefore, to reduce aerodynamic drag and lift of the train becomes the key issue of optimization design of head shape of high-speed trains (Tian 2007).

In the initial stage of optimization design, the parametric method to efficiently represent the shape with significantly less data should be developed. It plays an important role in aerodynamic optimal design for the reason that an efficient parametric method can not only describe the changes of shape completely, but also reduce the optimization cycle and improve optimization efficiency. For airfoil optimization, various parametric approaches have been investigated since the 1970s. Free form deformation (FFD), class function transformation (CST), Hicks-Henne approach and B-spline representation method are widely used in airfoil design. The head shape of a high-speed train is much more complicated than the airfoil, and there are much fewer references on parametric methods for high-speed trains. Rho et al. (2009) proposed a vehicle modeling function in the form of an exponential function to smoothly express the complex shapes of an automobile. Ku et al. (2010) applied the VMF method in the optimization design of high-speed trains to reduce the micro-pressure wave and aerodynamic drag. In the present paper, a modified VMF parametric approach for high-speed train nose shape is adopted.

4.1 Vehicle modeling function parametric approach

The basic VMF proposed by Rho and Ku is given by:

$$F\left(\frac{x}{c}\right) = \left(\frac{x}{c}\right)^{A_1} \left(1 - \frac{x}{c}\right)^{A_2} S\left(\frac{x}{c}\right) + \left(1 - \frac{x}{c}\right) Y_1 + \left(\frac{x}{c}\right) Y_2, \tag{25}$$

where Y_1 and Y_2 are the heights of the starting and finishing points and x and c are the dimension and length of each section box. $S(x/c)$ can be defined by the simple polynomial functions. In order to generate outline curves of the streamlined nose shape, the equation is modified by:

$$F\left(\frac{x}{c}\right) = \left(\frac{x}{c}\right)^{A_1} \left(1 - \frac{x}{c}\right)^{A_2} a_k + g\left(\frac{x}{c}\right), \tag{26}$$

where $S(x/c)$ and $g(x/c)$ are given by:

$$g\left(\frac{x}{c}\right) = 2(Y_2 - Y_1) \frac{x}{c} - (Y_2 - Y_1) \left(\frac{x}{c}\right)^2 \tag{27}$$

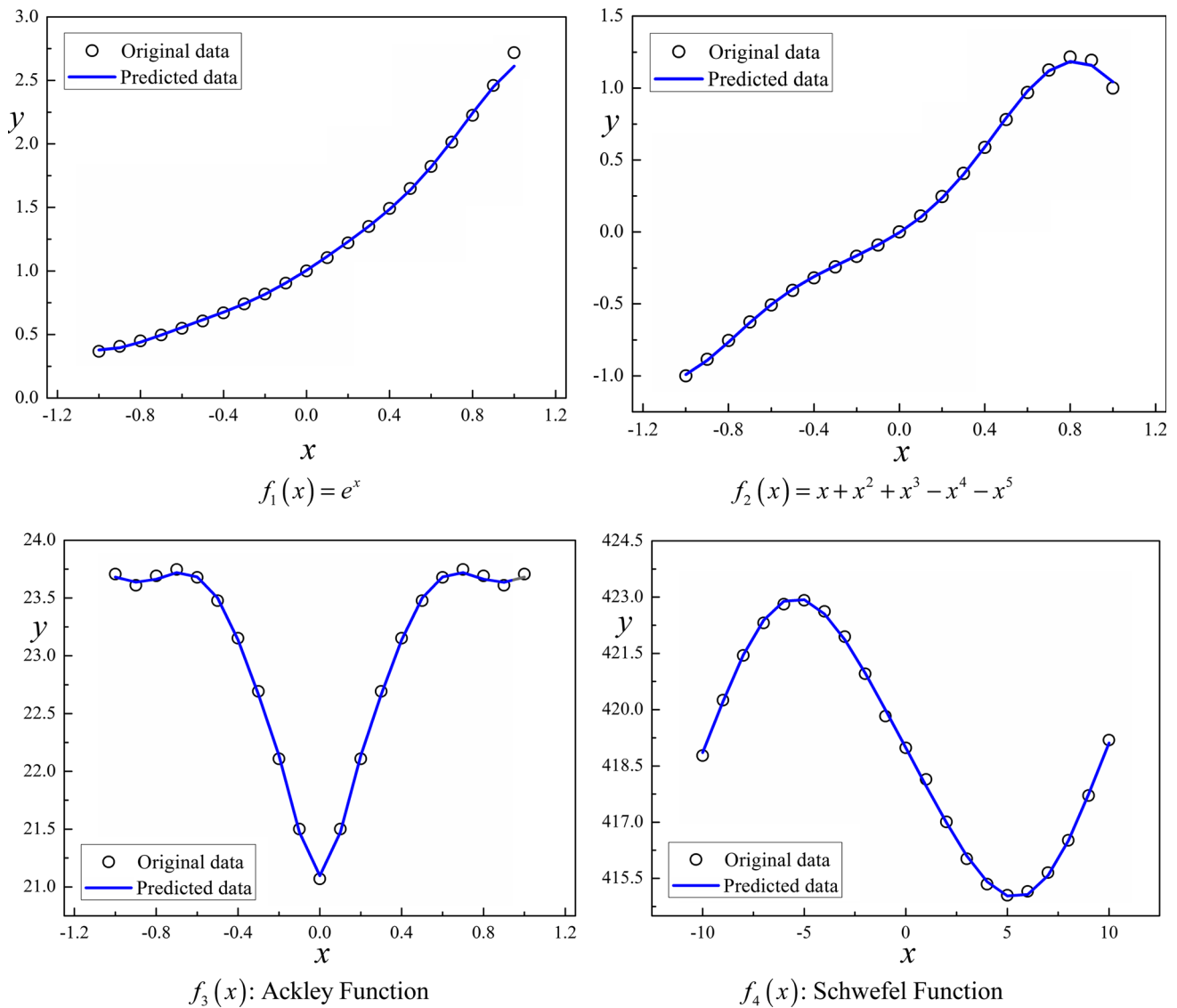


Fig. 8 Test results of SVR model

Table 3 Regression accuracy of SVR

Function	Number of points	Mse	R^2
$f_1(x)$	20	6.0347×10^{-4}	0.9990
$f_2(x)$	20	2.3552×10^{-4}	0.9995
$f_3(x)$	20	8.2278×10^{-5}	0.9992
$f_4(x)$	20	9.4670×10^{-5}	0.9992

A_1 , A_2 and a_k are design variables. A_1 and A_2 control the basic curve shapes, and a_k affects the amplitude of $F(x/c)$. Using the modified function, we can define different complex shapes by varying the design variables. Figure 9 illustrates how the shape changes with the exponent A_1 and A_2 in the modified VMF equation.

The basic shape of streamlined head of high-speed trains is controlled by the critical two-dimensional line, and surface configuration is generated by means of interpolation between different profile lines. According to the common nose shapes of high-speed trains, basic lines and control points are extracted, as shown in Fig. 10.

L1 is the upper longitudinal section profile line, L2 is horizontal section profile, L3 is the bottom horizontal section profile, L4 is the cowcatcher profile line, and L5 is the maximum cross-sectional profile line. According to the functional requirements, the shape of cowcatcher profile line cannot be changed too much; therefore, L4 adopts realistic geometric structure. In addition, L5 has a fixed shape since it should matches with the carriage of high-speed train. Equation of L1 is given by:

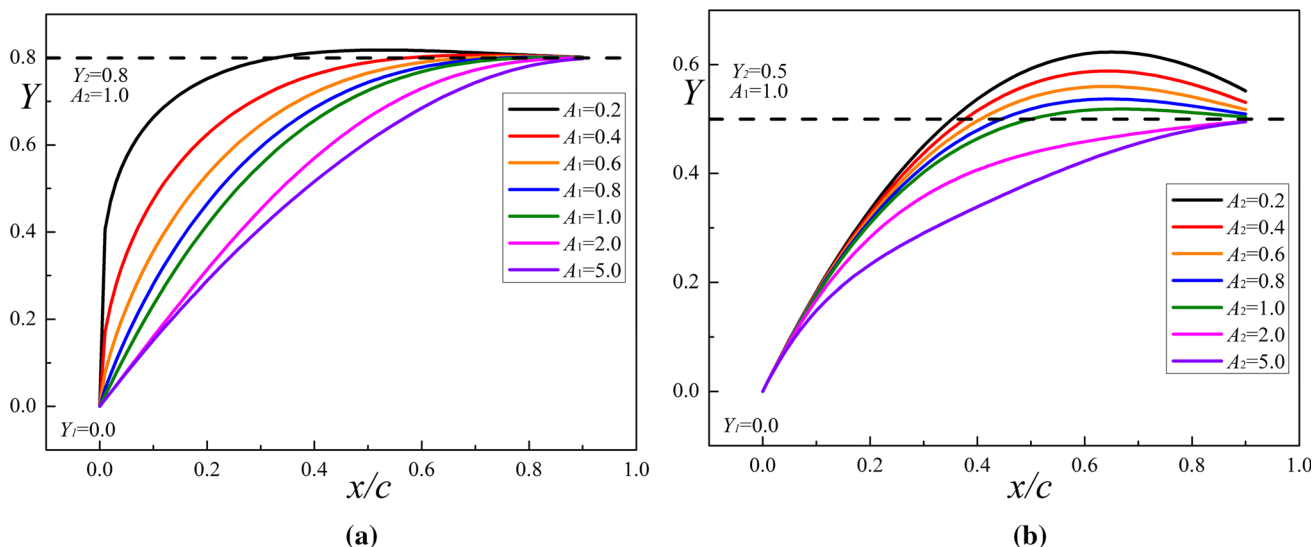


Fig. 9 a Curves variation with the parametric A_1 . b Curves variation with the parametric A_2

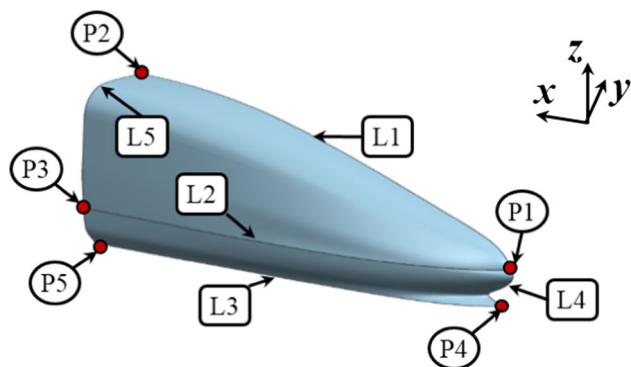


Fig. 10 Basic lines and control points in VMF design

$$z(x) = \left(\frac{x - x_{11}}{x_{12} - x_{11}}\right)^{A_{11}} \left(1 - \frac{x - x_{11}}{x_{12} - x_{11}}\right)^{A_{12}} a_{k1} + 2(z_{12} - z_{11}) \frac{x - x_{11}}{x_{12} - x_{11}} - (z_{12} - z_{11}) \left(\frac{x - x_{11}}{x_{12} - x_{11}}\right)^2, \tag{28}$$

where $(x_{11}, z_{11}), (x_{12}, z_{12})$ are the coordinate values of P1 and P2. Analogously, equation of L2 is given by:

$$y(x) = \left(\frac{x - x_{11}}{x_{22} - x_{11}}\right)^{A_{21}} \left(1 - \frac{x - x_{11}}{x_{22} - x_{11}}\right)^{A_{22}} a_{k2} + 2(z_{22} - z_{11}) \frac{x - x_{11}}{x_{22} - x_{11}} - (z_{22} - z_{11}) \left(\frac{x - x_{11}}{x_{22} - x_{11}}\right)^2, \tag{29}$$

Table 4 Design variables and their ranges

Design variables	Range	Design variables	Range
A_{11}	(0.20,0.60)	A_{32}	(2.20,3.50)
A_{12}	(9.80,12.00)	a_{k1}	(0.90,2.70)
A_{21}	(0.17,0.43)	a_{k2}	(1.15,3.45)
A_{22}	(1.20,2.50)	a_{k3}	(1.25,3.75)
A_{31}	(0.30,0.90)		

where (x_{22}, z_{22}) is the coordinate value of P3. For L3, the equation is given by:

$$y(x) = \left(\frac{x - x_{31}}{x_{32} - x_{31}}\right)^{A_{31}} \left(1 - \frac{x - x_{31}}{x_{32} - x_{31}}\right)^{A_{32}} a_{k3} + 2(z_{32} - z_{31}) \frac{x - x_{31}}{x_{32} - x_{31}} - (z_{32} - z_{31}) \left(\frac{x - x_{31}}{x_{32} - x_{31}}\right)^2, \tag{30}$$

where $(x_{31}, z_{31}), (x_{32}, z_{32})$ are the coordinate values of P4 and P5. During the process of shape construction, control points P1 to P5 are unchanged, as a result, the basic design parameters are a_{ki}, A_{i1} and $A_{i2}, (i = 1, 2, 3)$.

Table 4 shows the ranges of design parameters, and Fig. 11 shows different types of streamlined nose shapes obtained by adjusting the design parameters.

4.2 Computational domain and CFD algorithm

In the present paper, a mixture spatial mesh that combines with Cartesian grid and prism mesh is adopted. The total

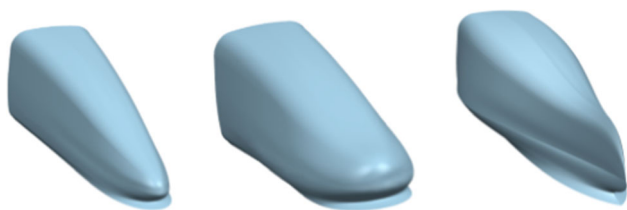


Fig. 11 Different shapes obtained by adjusting design parameters

length of the train is about 78 m, in which the length of the head car and the tail car is about 26 m. The height H of the train is 3.5 m, while the width of the car-body is 3.38 m. Take the height H as the characteristic length, and the computational domain extends $30H$ ahead of the train nose and $60H$ from the train tail to the exit of the computational domain. The top of the computational domain is at a distance $30H$ from the bottom of the rail and the sides are at a distance of $30H$ from the center axis of the train, the outline of computational domain and the model are shown in Fig. 12.

In this paper, the speed of high-speed train is 350 km/h, so the Mach number was 0.2859. In this condition, the air compression characteristic had an obvious effect on the aerodynamic drag of the train. Therefore, the steady compressible Reynolds-averaged Navier-Stokes equation (Blazek 2015) that is based on the finite volume method was used to predict the aerodynamic drag. Roe's FDS scheme was used to calculate convective fluxes, and lower-upper symmetric Gauss-Seidel (LU-SGS) was chosen for temporal discretization. The $k-\omega$ SST model was selected as the turbulence model. The standard wall functions were used near the wall so that the accuracy of the CFD results could be ensured with a limited amount of mesh. The flow velocity is 97.222 m/s, the far-field pressure is 1 atm, the temperature is 288 K, and the reference area is the maximum cross-sectional area of the train. As a result of the compressibility calculation model, one-dimensional inviscid flow of the Riemann invariants is introduced as the far-field boundary conditions, which are also known as non-reflective boundary conditions. Inflow, outflow and the top boundaries are all set as far-field boundary conditions and the train body is nonslip solid wall boundary condition. The ground is treated as the moving wall so as to simulate the ground effect, and the moving speed is equal to the train speed.

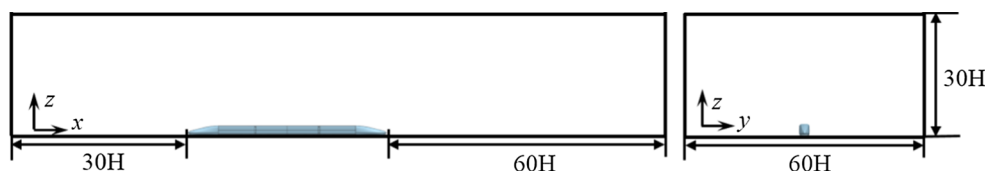


Fig. 12 Computational domain

4.3 Optimization process

In order to improve the optimization efficiency, a reasonable optimization process is designed in the present paper, as shown in Fig. 13.

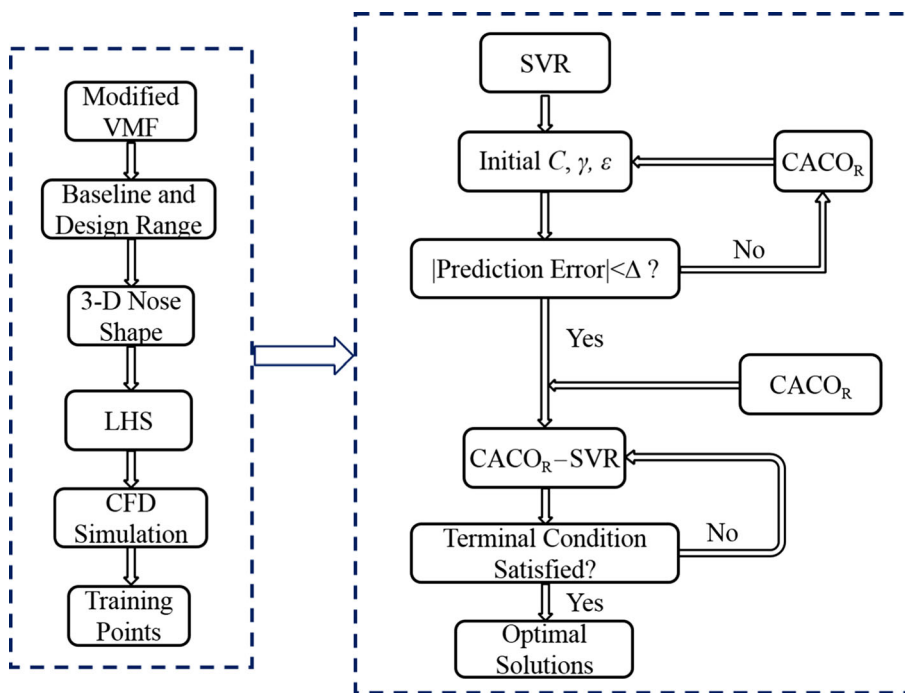
All the steps are listed as follows:

- (1) Modify the VMF equation and extract the design parameters.
- (2) Determine baselines composing the streamlined surface and then determine the range of design parameters.
- (3) Generate a certain number of sample points using Latin hypercube sampling (LHS) method.
- (4) Obtain the accurate value of the aerodynamic drag and lift for every sample point using the CFD simulation. Train the SVR model constructed with RBF kernel. Because there are two objectives, two sets of SVR models need to be constructed. On the premise of sufficient precision, two sets of models share the same parameters C , ε and γ in the present paper. The sum of the squares of the two errors is defined as the final prediction error. If the prediction error is greater than limited value Δ , then use the proposed algorithm to find the optimal values of correlation coefficients C , ε and γ .
- (5) Construct multi-objective CACO_R algorithm using non-dominated sorting concept and niche counts method, test the performance with different functions and then get the CACO_R-SVR model.
- (6) Based on the CACO_R-SVR model, perform the optimization for 3D nose shape of high-speed trains, obtain the optimal Pareto solutions.
- (7) Chose one of the solutions as the design point and generate the optimal shape correspondingly. Verify their performance using the CFD approach.

4.4 Sample points and training of SVR

A uniform distribution of samples is essential for training the SVR model, and reasonable sampling method could provide the detailed information with fewer samples in the design space. Latin hypercube sampling is a recent development in sampling technology designed to accurately recreate the input distribution through sampling in a few iterations. References (Joseph and Hung 2008) have shown that LHS offers great benefits in terms of increased sampling efficiency and

Fig. 13 Optimization process



better reflects the underlying distribution for a smaller number of samples. As a result, LHS is adopted in the present paper to generate initial samples according to the design variables and their ranges listed in Table 4. Twenty-two initial sampling points have been generated for training the SVR model, of which the first 20 points are chosen as training points while the last two are chosen as the test points. For all the sampling points, the aerodynamic drag of the whole train (C_d) and the lift of the tail car (C_l) are calculated and designed as optimization objectives, as shown in Table 5.

In general, the essence of training the SVR model based on RBF kernel is to find the most appropriate value of C , ϵ and γ . A basic SVR model could be firstly constructed from the initial 20 training points. If the average prediction error for the other two test points is greater than 5%, then optimize the key parameters C , ϵ and γ with CACO_R until the accuracy requirement is satisfied. For the selection of the most appropriate values of SVR model parameters, take C , ϵ and γ as the vector of independent variables and mean square error as the optimization objective, then the appropriate parameters could be obtained by the CACO_R algorithm. Table 6 compares the prediction error of the initial parameters and the optimal parameters. The initial values of C , ϵ and γ are 2.0, 0.1 and 3.0, while after optimization the parameters are changed to 45.6238, 0.0101 and 2.4630, respectively.

It can be observed that aerodynamic lift coefficient of the tail car is more sensitive to the input parameters of SVR model. Prediction error of the initial C , ϵ and γ for aerodynamic drag coefficient is about 3.0, while the error of the optimal C , ϵ and γ is reduced to 0.31%. For aerodynamic

lift of the tail car, prediction error declines from about 26% to 3.0% after optimization. Hence, the optimal SVR models have met the accuracy requirement.

5 Results and discussion

Based on the constructed SVR model, the Pareto optimal solution of the aerodynamic drag of the train and the aerodynamic lift of the tail car is found. For the setting of parameters in CACO_R algorithm, ant colony population is set to 100, parameter ρ is set to 0.9. The default number of generation N is set to 300.

Figure 14 shows the Pareto solution of the C_d - C_l optimization based on CACO_R-SVR. The result shows that a suitable Pareto front is obtained after iterations. Theoretically, the objective values of each point on the Pareto frontier are better than those of the original one. In the shape design for high-speed train, when the aerodynamic parameters are balanced, the running stability of the train is the best. As a result, points near the center of the Pareto frontier are more likely to be the final design point. In the present paper, a specific individual as the red star shows in Fig. 14 is chosen as the design point for comparison.

Table 7 shows aerodynamic forces comparison between original and optimal shape. After optimization, the aerodynamic drag of the whole train is reduced by 10.52%, while the aerodynamic lift of the tail car is reduced by 35.70%. The results indicate that the aerodynamic lift of the tail car is more sensitive to the change of nose shape.

Table 5 Training points and test points

Type	Number	A_{11}	A_{12}	A_{21}	A_{22}	A_{31}	A_{32}	a_{k1}	a_{k2}	a_{k3}	Cd	Cl
Training points	1	0.2991	9.9801	0.2226	1.6503	0.8136	3.3644	2.2130	1.4009	3.6174	0.16230	0.08490
	2	0.3135	11.8896	0.2670	2.4229	0.3714	2.5927	2.1553	1.2370	3.0004	0.16241	0.08447
	3	0.5585	11.2598	0.3106	1.4789	0.6960	2.8826	1.6966	2.2095	2.6580	0.16217	0.06746
	4	0.5247	11.0884	0.2972	2.2250	0.8690	3.1583	1.2923	3.0381	1.6319	0.16692	0.08359
	5	0.4051	10.0944	0.3222	2.2942	0.6572	2.2456	2.6628	2.3088	1.2659	0.15035	-0.00192
	6	0.4860	10.6549	0.3270	1.7647	0.8975	2.3317	2.4289	1.6899	2.8705	0.14983	-0.02569
	7	0.3661	11.1728	0.3670	1.2092	0.7666	3.2856	1.6277	3.2748	1.4871	0.16399	0.00882
	8	0.4516	10.4577	0.3788	2.0605	0.4505	2.7845	1.8045	1.4952	2.0942	0.14982	0.00727
	9	0.4207	11.6026	0.2542	2.3638	0.8221	2.8415	1.0067	3.1376	2.5292	0.17045	0.05569
	10	0.4652	11.7447	0.2457	1.9343	0.5334	2.6965	1.0674	2.6701	1.3974	0.16319	0.04437
	11	0.4976	9.8646	0.1733	1.5449	0.3274	3.4006	1.7342	2.9372	2.3210	0.20527	0.16202
	12	0.2510	10.1521	0.3912	1.3023	0.5936	2.4132	1.4498	2.5244	3.1954	0.16751	0.01952
	13	0.5910	10.5372	0.3388	2.4622	0.6231	2.3129	2.3500	2.7842	2.7835	0.16665	0.02353
	14	0.3907	11.3937	0.3579	1.4041	0.7341	3.2583	1.2173	1.8671	2.2394	0.14836	-0.02226
	15	0.2159	10.3830	0.4043	1.8728	0.4360	2.4713	1.5204	1.9837	2.4140	0.16186	0.02981
	16	0.2610	10.9849	0.2337	2.0140	0.5047	2.5151	2.0512	2.8824	1.7938	0.18084	0.04759
	17	0.3536	11.4373	0.1877	1.3616	0.3074	2.6640	1.3846	1.6724	3.6448	0.17622	0.22635
	18	0.2357	11.9593	0.2137	2.1709	0.6300	3.4671	2.0291	2.0442	3.4539	0.16840	0.01069
	19	0.2735	11.5873	0.4280	1.8319	0.5593	3.1398	1.9052	1.3084	1.8186	0.14703	-0.01617
	20	0.5450	10.2934	0.2836	2.1422	0.7523	3.0292	0.9445	2.4391	1.9569	0.15521	0.00926
Test points	21	0.5671	10.7668	0.2004	1.6009	0.4026	2.9996	2.5351	2.1359	3.0683	0.18263	0.09535
	22	0.3394	10.8207	0.4102	1.7144	0.4659	2.9514	2.5949	3.3969	3.3170	0.18985	0.07477

Table 6 Prediction error comparison

Objective	Points	Actual value	Initial(C, ε, γ)		Optimal(C, ε, γ)	
			Predicted value	Error	Predicted value	Error
Cd	21	0.18263	0.17682	3.18%	0.18206	0.31%
	22	0.18985	0.18402	3.07%	0.18926	0.31%
Cl	21	0.09535	0.07016	26.42%	0.09271	2.77%
	22	0.07477	0.05504	26.39%	0.07224	3.38%

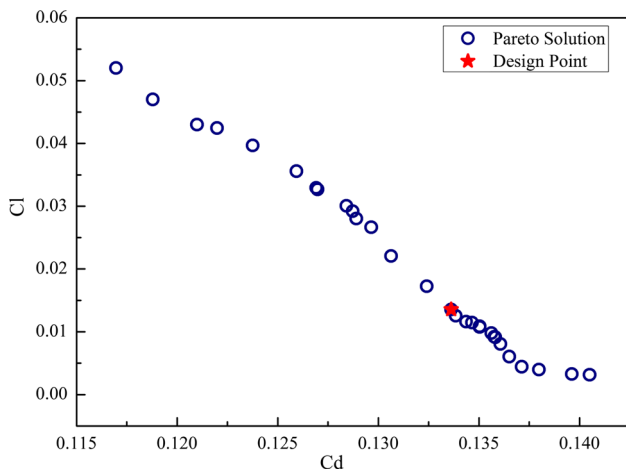


Fig. 14 Pareto solution based on CACOR-SVR

Table 7 Aerodynamic forces reduction after CACOR-SVR optimization

Model	Cd	Cl
Original shape	0.14933	0.02109
Optimal shape	0.13315	0.01304
Reduction	10.52%	35.70%

Figure 15 illustrates the transformation of L1, L2 and L3 after optimization. It can be seen from Fig.15 that L1 changes a little near P1 and almost remains unchanged near P2. Curvature of L2 and the first half of L3 decreases significantly, while curvature of the second half of L3 increases slightly.

In order to better understand the influence on aerodynamic performance due to the change of nose shape of high-speed train, pressure distribution of head car and the iso-surface of

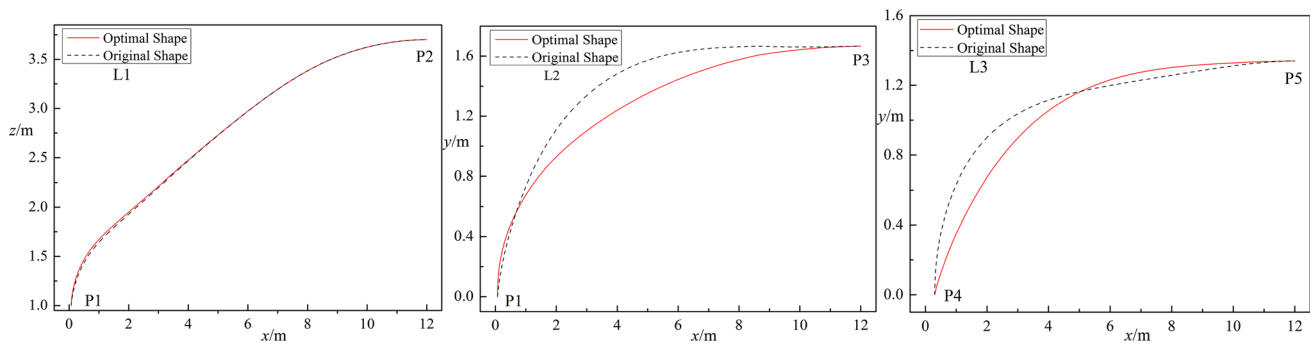


Fig. 15 Comparison for section profile lines

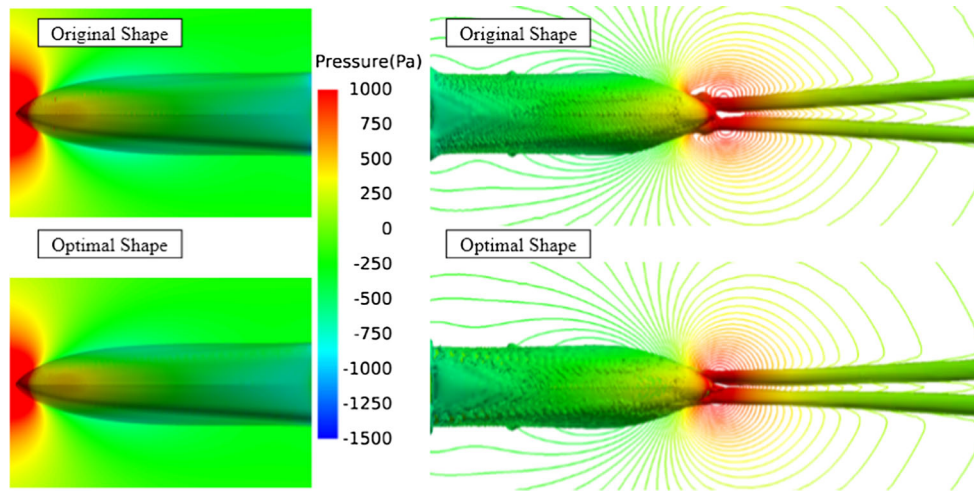


Fig. 16 Pressure distribution of head car and the iso-surface of $Q = 80$ in the wake flow

the second invariant of the velocity gradient Q in the wake flow are shown in Figure 16. It can be concluded that the high-pressure area of the head car is reduced after optimization. Meanwhile, the strengths of two steady vortices which were developed along the surface of the tail car are reduced; hence, negative pressure in the wake region is weaker. The results indicate that decreasing the curvature of horizontal section profile and the curvature of bottom horizontal section profile of the nose can be beneficial to local pressure distribution of the train.

6 Conclusions

In the present paper, a chaos ant colony optimization algorithm for continuous domain (CACO_R) is proposed based on chaos optimization theory and ant colony optimization algorithm. Five single-objective test functions and two multi-objective functions are used to verify the searching efficiency and global optimization capability of the proposed algorithm. Meanwhile, the proposed algorithm is improved for multi-objective optimization by introducing the non-dominated

sorting concept and niche counts method. The results show that CACO_R algorithm gets a good performance at single-objective optimization as well as searching for Pareto front in multi-objective optimization.

With limited sample points, an optimal-parameter SVR surrogate model based on RBF kernel function is constructed. Both results of test functions and test points demonstrate sufficient prediction accuracy of the SVR model.

According to common shape of high-speed trains, a modified VMF parametric method is improved and 9 design variables are extracted. Combined with the proposed CACO_R-SVR approach, optimization for nose shape of high-speed train has been performed, the aerodynamic drag coefficient is reduced by 10.52%, and the aerodynamic lift of the tail car is reduced by 35.70%. The results can be beneficial to the shape design of high-speed trains.

Acknowledgements This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (class B) (Grant No. XDB22020000) and National Key Research & Development Projects (Grant No. 2017YFB0202800), and the Computing Facility for Computational Mechanics Institute of Mechanics at the Chinese Academy of Sciences is also gratefully acknowledged.

Compliance with ethical standards

Conflict of interest Authors ZHANG Ye, YANG GuoWei, GUO DiLong, SUN ZhenXu, and CHEN DaWei declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

- Baker C (2010) The flow around high speed trains. *J Wind Eng Ind Aerodyn* 98(6):277–298. <https://doi.org/10.1016/j.jweia.2009.11.002>
- Bellman M, Straccia J, Morgan B, Maschmeyer K, Agarwal R (2009) Improving genetic algorithm efficiency with an artificial neural network for optimization of low Reynolds number airfoils. *AIAA Pap.* <https://doi.org/10.2514/6.2009-1096>
- Blazek J (2015) *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, Oxford
- Bourinet JM (2016) Rare-event probability estimation with adaptive support vector regression surrogates. *Reliab Eng Syst Saf* 150:210–221. <https://doi.org/10.1016/j.res.2016.01.023>
- Coello CC (2006) Evolutionary multi-objective optimization: a historical view of the field. *IEEE Comput Intell Mag* 1(1):28–36
- Cong S, Li G, Feng X (2010) An improved algorithm of chaos optimization. In: *control and automation (ICCA), 2010 8th IEEE international conference*, pp 1196–1200. <https://doi.org/10.1109/ICCA.2010.5524290>
- Datta R, Regis RG (2016) A surrogate-assisted evolution strategy for constrained multi-objective optimization. *Expert Syst Appl* 57:270–284. <https://doi.org/10.1016/j.eswa.2016.03.044>
- Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197. <https://doi.org/10.1109/4235.996017>
- Dorigo M (1992) *Optimization, learning and natural algorithms* [in Italian]. PhD Thesis, di Elettronica, Politecnico di Milano, Milan, Italy
- Forrester AI, Keane AJ (2009) Recent advances in surrogate-based optimization. *Prog Aerosp Sci* 45(1):50–79. <https://doi.org/10.1016/j.paerosci.2008.11.001>
- Goldberg DE, Richardson J (1987) Genetic algorithms with sharing for multimodal function optimization. In: *Genetic algorithms and their applications: proceedings of the second international conference on genetic algorithms*, vol 60, no 6. Lawrence Erlbaum, Hillsdale, NJ, pp 41–49
- Gonçalves JF, de Magalhães Mendes JJ, Resende MGC (2005) A hybrid genetic algorithm for the job shop scheduling problem. *Eur J Oper Res* 167(1):77–95. <https://doi.org/10.1016/j.ejor.2004.03.012>
- Holland JH (1975) *Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence*. University of Michigan Press, Ann Arbor
- Horn J, Nafpliotis N, Goldberg D E (1994) A niched Pareto genetic algorithm for multi-objective optimization. In: *Proceedings of the first IEEE conference on evolutionary computation*. IEEE world congress on computational intelligence, 27–29 June, 1994. IEEE, Orlando, FL, USA
- Ikeguchi T, Hasegawa M, Kimura T, Matsuura T, Aihara K (2011) *Theory and applications of chaotic optimization methods. Innovative computing methods and their applications to engineering problems*. Springer, Berlin, pp 131–161
- Iosifidis A, Gabbouj M (2016) Multi-class support vector machine classifiers using intrinsic and penalty graphs. *Pattern Recogn* 55:231–246. <https://doi.org/10.1016/j.patcog.2016.02.002>
- Joseph VR, Hung Y (2008) Orthogonal-maximin Latin hypercube designs. *Statistica Sinica*, pp 171–186
- Kennedy J, Eberhart R (1995) Particle swarm optimization. In: *Proceedings of the IEEE international conference on neural networks*, vol 4. IEEE Service Center, Piscataway, NJ, pp 1941–1948
- Konak A, Coit DW, Smith AE (2006) Multi-objective optimization using genetic algorithms: a tutorial. *Reliab Eng Syst Saf* 91(9):992–1007. <https://doi.org/10.1016/j.res.2005.11.018>
- Koziel S, Leifsson L (2012) Surrogate-based aerodynamic shape optimization by variable-resolution models. *AIAA J* 51(1):94–106. <https://doi.org/10.2514/1.J051583>
- Koziel S, Leifsson L, Yang XS (2014) Solving computationally expensive engineering problems. Springer International Publishing, New York, pp 25–51
- Ku YC, Kwak MH, Park HI, Lee DH (2010) Multi-objective optimization of high-speed train nose shape using the vehicle modeling function. In: *48th AIAA aerospace sciences meeting*. Orlando, USA. <https://doi.org/10.2514/6.2010.1501>
- Kursawe F (1991) A variant of evolution strategies for vector optimization. *Parallel Probl Solving Nat.* <https://doi.org/10.1007/BFb0029752>
- Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Comput Methods Appl Mech Eng* 194(36):3902–3933. <https://doi.org/10.1016/j.cma.2004.09.007>
- Lee SJ, Kim B, Baik SW (2011) Neural network modeling of inter-characteristics of silicon nitride film deposited by using a plasma-enhanced chemical vapor deposition. *Expert Syst Appl* 38(9):11437–11441. <https://doi.org/10.1016/j.eswa.2011.03.016>
- Li X (2003) A non-dominated sorting particle swarm optimizer for multi-objective optimization. *Genetic and Evolutionary Computation—GECCO 2003*. Springer, Berlin, pp 198–198. https://doi.org/10.1007/3-540-45105-6_4
- Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans Evol Comput* 10(3):281–295. <https://doi.org/10.1109/TEVC.2005.857610>
- Liu L, Dai Y, Gao J (2014) Ant colony optimization algorithm for continuous domains based on position distribution model of ant colony foraging. *Sci World J.* <https://doi.org/10.1155/2014/428539>
- May RM (1976) Simple mathematical models with very complicated dynamics. *Nature* 261(5560):459–467
- Muñoz-Paniagua J, García J, Crespo A (2014) Genetically aerodynamic optimization of the nose shape of a high-speed train entering a tunnel. *J Wind Eng Ind Aerodyn* 130:48–61. <https://doi.org/10.1016/j.jweia.2014.03.005>
- Ragunathan RS, Kim HD, Setoguchi T (2002) Aerodynamics of high-speed railway train. *Prog Aerosp Sci* 38(6):469–514. [https://doi.org/10.1016/S0376-0421\(02\)00029-5](https://doi.org/10.1016/S0376-0421(02)00029-5)
- Rho JH, Ku YC, Kee JD, Lee DH (2009) Development of a vehicle modeling function for three-dimensional shape optimization. *J Mech Des* 131(12):121004. <https://doi.org/10.1115/1.4000404>
- Simpson TW, Mauery TM, Korte JJ, Mistree F (2001) Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA J* 39(12):2233–2241. <https://doi.org/10.2514/2.1234>
- Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. *Stat Comput* 14(3):199–222. <https://doi.org/10.1023/B:STCO.0000035301.49549.88>
- Smola A, Vapnik V (1997) Support vector regression machines. *Adv Neural Inf Process Syst* 9:155–161
- Socha K, Dorigo M (2008) Ant colony optimization for continuous domains. *Eur J Oper Res* 185(3):1155–1173. <https://doi.org/10.1016/j.ejor.2006.06.046>

- Srinivas M, Patnaik L (1994) Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Trans Syst Man Cybern* 24(4):656–667. <https://doi.org/10.1109/21.286385>
- Tian H (2007) Train aerodynamics. China Railway Publishing House, Beijing, pp 1–15 (in Chinese)
- Tian HQ (2009) Formation mechanism of aerodynamic drag of high-speed train and some reduction measures. *J Cent South Univ Technol* 16:166–171. <https://doi.org/10.1007/s11771-009-0028-0>
- Vladimir VN, Vapnik V (1995) The nature of statistical learning theory. Springer, New York
- Yang G, Guo D, Yao S, Liu C (2012) Aerodynamic design for China new high-speed trains. *Sci China Technol Sci* 55(7):1923–1928. <https://doi.org/10.1007/s11431-012-4863-0>
- Zhan ZH, Zhang J, Li Y, Chung HSH (2009) Adaptive particle swarm optimization. *IEEE Trans Syst Man Cybern Part B (Cybern)* 39(6):1362–1381. <https://doi.org/10.1109/TSMCB.2009.2015956>
- Zitzler E, Deb K, Thiele L (2000) Comparison of multi-objective evolutionary algorithms: empirical results. *Evol Comput* 8(2):173–195. <https://doi.org/10.1162/106365600568202>