Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 2028196, 9 pages
https://doi.org/10.1155/2018/2028196

Research Article

Parameter Identification of the Vortex-Induced Vertical Force Model Using a New Adaptive PSO

Xiaoxia Tian,[1,2] **Jingwen Yan**,[2] **and Chi Xiao**[1]

[1]*Computer and Information Engineering College, Hanshan Normal University, Chaozhou 521041, China*
[2]*College of Engineering, Shantou University, Shantou 515063, China*

Correspondence should be addressed to Xiaoxia Tian; 14xxtian@stu.edu.cn

Received 10 May 2018; Revised 10 October 2018; Accepted 16 October 2018; Published 28 October 2018

Guest Editor: Eduardo Rodriguez-Tello

The paper proposes a new adaptive PSO (NAPSO) that adaptively adjust the inertial weight of every particle according to its own current fitness. In NAPSO, the searching ability of each particle is controlled by the inertial weight. In pursuit of the optimal solution, if a particle has a rather small value of normalized fitness, it has a small inertia weight so as to increase local searching ability; on the contrary, it has a large inertia weight to increase global searching ability. Simulation results include three parts: the NAPSO shows fast convergence and good stability compared with other PSOs; the NAPSO shows good fit and short run-time compared with GA and GALMA; according to the identified parameters, the time history of predicted vertical displacement is quite in accordance with the time history of measured displacement. As far as the nonlinear VIVF model is concerned, the NAPSO is a simple and effective identification method.

1. Introduction

In wind engineering, the vortex-induced vibration (VIV) is the current research hotspot. Many researchers have studied the VIVs for many years and achieved a lot of results [1–5]. Ehsan and Scanlan [1] proposed a nonlinear model to estimate the maximum VIV response of bluff bodies, and this model has been widely applied in engineering [6]. However, some researchers [5, 7–9] have found that Scanlan's nonlinear model fails to depict the VIV phenomena; they have proposed their own models, respectively. Among these new models, the improved nonlinear model presented by Zhu et al. [9] is special, because it is based on the measured vortex-induced vertical force (VIVF) and its validity is verified using the experimental data.

Due to the complexity of VIV, most VIV models are usually nonlinear. Parameter identification for nonlinear models is complex and the accuracy of parameters has direct influence on predicting prototype VIV response. The earliest identification method is the grow-to-resonance method (GTR) [1] that uses the vertical vibration displacement in wind tunnel tests to identify parameters. It is difficult to accurately measure the vertical vibration displacement due to its small amplitude and noise in wind tunnel tests. To reduce the effect of incorrect measurement, Gupta et al. [10] attempted to use least squares method to identify parameters of VIVF model. Zhu et al. [9] used the nonlinear least squares fitting to identify parameters of their model from wind tunnel tests. Levenberg-Marquardt algorithm (LMA) is adopted to solve nonlinear equations. However, LMA is sensitive to the initial solution, and the initial solution corresponding to the global optimal solution is difficult to be known in advance. Weng et al. [11] used the genetic algorithm (GA) to identify parameters of VIVF models. However, GA has crossover and mutation operations, which increase the algorithm complexity and the amount of calculation. Tian et al. [12] proposed a hybrid algorithm based on GA and LMA. This algorithm includes two parts: GA generates a solution randomly and LMA uses this solution as the initial solution to converge to a better solution. Though GALMA identifies the models well, the upper bound and lower bound of solutions also need to be set ahead of time. Hence, it is urgent to develop a reliable identification method that does not solely depend upon the appropriate initial conditions.

The particle swarm optimization (PSO) is a stochastic algorithm motivated by the social behavior of individuals in

a swarm such as bird flocking [13]. It has good characteristics including fast convergence, easy implementation, and few parameters. These characteristics make PSO a potential method for solving optimization problems in various areas [14–16], and PSO is considered especially in the field of models' parameter identification [17–19].

The performance of PSO algorithm highly depends on the inertia weight which can be adjusted to get a balance between the global and local searching abilities. Since this balance is vital to the success of an optimization algorithm, the improvement of inertia weight is a focus [20]. The function of inertia weight may be a fixed number, a random number, a nonlinear function, an adaptive function, etc. An appropriate strategy lies in three parts: the inertia weight should vary with running situations; the function of inertia weight should describe main features of each particle in running; the stability and complexity of the algorithm need to be taken into account.

However, Shi and Eberhart [21] used random inertia weight without considering the running condition of each particle. Liao et al. [22] proposed a nonlinear decreasing strategy of the inertia weight. This PSO only considers iterations rather than the environment of the swarm; its stability is not satisfied. Arumugam [23] presented an adaptive PSO, in which every particle's own situation is not considered, so the stability of this PSO is difficult to be controlled. Taherkhani and Safabakhsh [20] proposed an adaptive stable PSO which uses the complex function of inertia weight to compute the inertia weight of every particle in different dimensions. The strategy undoubtedly increases the complexity of the algorithm and harms its features, such as rapid convergence and simplicity.

In the paper, a new adaptive PSO (NAPSO) is proposed to identify parameters of Zhu's model. In NAPSO, the inertia weight of each particle is adaptively adjusted by its normalized fitness at last iteration. This strategy focuses on the running environment of every particle so that it can improve the stability and the speed of convergence. The performance of NAPSO is amply demonstrated by applying it to Zhu's model. Compared with other identification algorithms, the NAPSO has fast convergence, good stability, and high identification accuracy.

The rest of the paper is organized as follows. In Section 2, the principle of PSO is discussed in a brief, and different variant PSOs are also described. In Section 3, NAPSO for VIVF models is proposed; the change of inertia weight is illustrated in detail. In Section 4, first, NAPSO is compared with different variant PSOs to show its good performance. Second, NAPSO is compared with other identification methods to show its simplicity and reliability. Last, the parameters identified by NAPSO are verified by comparing the reconstructed data with the measured data. Section 5 summarizes the paper.

2. Related Knowledge of PSO

2.1. PSO Theory. PSO is an optimization algorithm based on swarm intelligence. In PSO, every particle is a potential solution and contains several parameters such as the current position, velocity, and the best position achieved so far by itself, and so on. For a k-dimensional solution space, the position and velocity of the ith particle are presented, respectively.

$$x_i = (x_{i1}, x_{i2}, \cdots, x_{ik})$$
$$v_i = (v_{i1}, v_{i2}, \cdots, v_{ik}) \tag{1}$$

at each iteration, the position and velocity of this particle are updated according to the following equations:

$$v_i^{j+1} = w * v_i^j + c_1 * r_1 * \left(pbest_i - x_i^j\right) + c_2 * r_2 * \left(gbest - x_i^j\right)$$
$$x_i^{j+1} = x_i^j + v_i^{j+1} \tag{2}$$

where $pbest_i = (pbest_{i1}, pbest_{i2}, \ldots, pbest_{ik})$ denotes the best position of the ith particle; $gbest$ denotes the best $pbest$ among all the particles in the swarm; c_1 and c_2 are learning factors which are nonnegative constants, respectively; r_1 and r_2 are two random values in the interval of $[0, 1]$, respectively; j denotes the number of iteration; w is the inertia weight. Importantly, the initialized position and velocity of every particle are generated randomly.

2.2. Different Variant of PSO. The inertia weight of PSO has great influence on the algorithm performance. A Large value of w increases the global searching ability and keeps the diversity of the population, whereas a small value enhances the local search capability. The tradeoff between global and local search is critical to the success of optimization algorithms. In addition, good stability is also important for this optimization.

Shi and Eberhart [24] proposed a fixed weight PSO algorithm (FPSO) which has a fixed w value. In their opinion, PSO can do well when w is assigned a value in the interval of $[0.8, 1.2]$. The global and local search abilities remain unchanged throughout the iteration.

Eberhart and Shi [25] presented a random PSO (RPSO) which uses a random function of inertia weight to enable all particles to track the best particle. This function can be expressed as follows.

$$w = 0.5 + 0.5 * r \tag{3}$$

where r is a random number in the interval $[0, 1]$. This strategy accelerates convergence in the early time of the algorithm, and its searching ability does not vary with time or iteration.

Researchers point out inertia weights should decrease with run-time or iteration number; they have defined different functions to show a nonlinear variation of the inertia weight. The value of inertial weight nonlinearly decreased, as the global searching ability of particles is corresponding to decrease and the local ability becomes strong. Liao et al. [22] proposed a nonlinear PSO (NLPSO), which uses a nonlinear decreasing function of inertia weight. The nonlinear function is written as follows.

$$w^j = \left(\frac{iter_{\max} - j}{iter_{\max}}\right)^d * (w_{\max} - w_{\min}) + w_{\min} \quad (4)$$

where j is the current number of iterations and d is a constant greater than 1. At the early time of the run, particles have quite large inertia weight; their global searching ability is strong; then their inertia weight nonlinearly reduces and their local search capability gradually becomes strong. This strategy increases the computational complexity and does not consider running situations of all particles.

Adaptive PSOs monitor the running situation and adjust the inertia weight according to the information of last iteration. Since the fitness is the most significant characteristic of all particles, Arumugam [23] proposed an adaptive PSO (APSO) which determines the inertia weight in next iteration by the ratio of the global best fitness achieved so far and the average of particles' local best fitness achieved so far.

$$w = 1.1 - \frac{fit(gbest)}{(1/n)\sum_{i=1}^{N} fit(pbest_i)} \quad (5)$$

where N denotes the number of particles. In this strategy, the inertia weight of all particles is adjusted according to the ratio of the fitness of the *gbest* particle to the mean fitness of all particles in the last iteration. If the ratio is small, the inertia weight becomes large; the particles have great global searching ability; otherwise, all particles have strong local searching ability. However, this strategy does not consider the running situation of each particle.

3. Proposed Method

In this section, we will discuss the proposed method in detail. Firstly, Zhu's model is described in brief, which is suitable to depict the nonlinear VIVF. Then, NAPSO is proposed, which uses a specific formulation of inertia weight to quickly get the optimal solution and ensure the stability of NAPSO. At last, the proposed PSO is applied in Zhu's model.

3.1. Zhu's Model. Zhu et al. [9] proposed an improved VIVF model. This model is suitable to describe the VIVF on a flat closed-box bridge deck. Zhu's model is written as follows.

$$m\left(\ddot{y} + 2\xi\omega_0\dot{y} + \omega_0^2 y\right) = F(\dot{y}, y, t) \quad (6)$$

$$F(\dot{y}, y, t) = \frac{1}{2}\rho U^2 D\left[Y_1(K)\left(1 - \varepsilon(K)\dot{y}^2\right)\dot{y} + Y_2(K)\, y + Y_3(K)\, y\dot{y} + \frac{1}{2}C_L(K)\sin(\omega t + \phi)\right] \quad (7)$$

where m is the mass per unit span length; ξ is the mechanical damping; ω_0 is the circular natural frequency of the bluff bodies; y, $\dot{y}$, and $\ddot{y}$ are the vertical displacement, velocity, and accelerated velocity, respectively; D is the characteristic size of the bluff body; U is the velocity of oncoming flow; ρ is the air density; ω is the circular frequency of vortex-shedding; ϕ is the phase angle. At lock-in, ω is appropriately close ω_0. $Y_1(K)$, $Y_2(K)$, $Y_3(K)$, $\varepsilon(K)$, and $C_L(K)$ are functions of the reduced frequency $K = \omega D/U$.

The dimensionless motion equation of VIVF can be expressed in the following format.

$$F_{VI} = m_r\left[Y_1\left(1 - \varepsilon\eta'^2(s)\right)\eta'(s) + Y_2\eta(s) + Y_3\eta(s)\eta'(s) + \frac{1}{2}C_L\sin(\omega_{VS}s + \phi)\right] \quad (8)$$

where $\eta = y/D$ denotes dimensionless response of vertical displacement; $m_r = \rho D^2/m$ denotes mass ratio; ω_{VS} denotes dimensionless circular frequency; $s = Ut/D$ is the dimensionless time; F_{VI} is the dimensionless VIVF. These notations Y_1, Y_2, Y_3, ε, C_L, ω_{VS}, and ϕ are aeroelastic parameters that need to be obtained from wind tunnel tests.

3.2. Proposed PSO. A new adaptive PSO (NAPSO) is proposed, which can adjust adaptively inertia weight to seek the optimal solution and keep the stability. To make a good decision about the inertia weight, the fitness of each particle in the last iteration is used as the feedback. The formula for the inertia weight of every particle can be expressed as below.

$$w = w_{\min} + \delta * \frac{\left(p_i^j - \min\left(p^j\right)\right)}{\left(\max\left(p^j\right) - \min\left(p^j\right)\right)} \quad (9)$$

where δ denotes a scale factor, which controls the range of inertia weight. p_i^j denotes the fitness of the ith particle in the jth iteration. p^j denotes the fitness vector of the swarm in the jth iteration. $(p_i^j - \min(p^j))/(\max(p^j) - \min(p^j))$, mapping the fitness of the ith particle to the interval $[0, 1]$, denotes the normalized fitness of the swarm.

According to (9), a particle that has the lowest fitness has smallest inertial weight in next iteration. Conversely, if a particle has the highest fitness, it has the largest inertial weight so that it may explore the largest search areas in the pursuit of the optimal solution. Figure 1 displays the inertial weight of the 3rd particle varying during the whole iteration. The blue circle represents the 3rd particle; its features include inertia weight, velocity, and position. Figure 1 highlights NAPSO adaptively adjusting the inertia weight of the 3rd particle in each iteration.

3.3. NAPSO for Zhu's Model. For every particle in NAPSO, $x = (Y_1, \varepsilon, Y_2, Y_3, C_L, \omega_{VS}, \phi)$ denotes its position and the least squares is utilized to define the objective function which can be written as follows.

$$Obj(x) = \left\|\tilde{F}_{VI} - \hat{F}_{VI}\right\|_2 \quad (10)$$

where $\|\cdot\|_2$ is the 2 norm; $\tilde{F}_{VI}$ denotes the time history of measured VIVF; $\hat{F}_{VI}$ is the time history of VIVF reconstructed by (8). So the fitness of the ith particle can be defined as follows:

$$p_i = Obj(x_i) \quad (11)$$

For Zhu's model, at the lock-in, a global optimal x^* is sought to minimize (10); it can be expressed that

$$x^* = \arg\min_{x \in R^7} Obj(x) \quad (12)$$

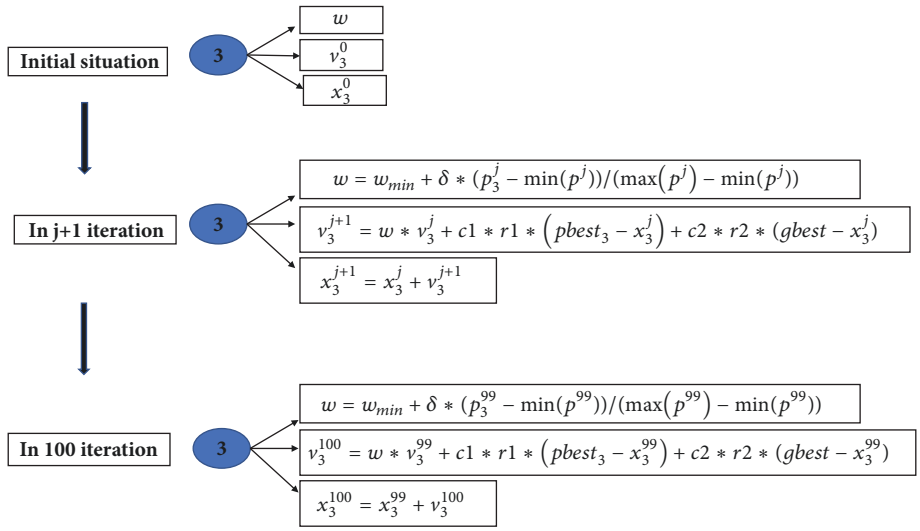

FIGURE 1: The situation of the 3rd particle.

Algorithm 1 (a pseudo-code of NAPSO).

Input: the maximum number of iterations $IterMax = 80$, $N = 40$, $w_{\min} = 0.5$, $\delta = 0.6$, $c_1 = 2.2$, and $c_2 = 1.7$.

For i=1 to N

A random vector of 7 dimension $\longrightarrow v_i^0$

A random vector of 7 dimensions, and $x_i^0 \longrightarrow pbest_i$

Compute p_i^0 according to Equation (11)

End

$x_1^0 \longrightarrow gbest$

For i= 2 to N

If $p_i^0 < Obj(gbest)$ $x_i^0 \longrightarrow gbest$

End

For j=1 to $IterMax$

For i=1 to N

Compute w_i^j according to Equation (9)

Compute v_i^j and x_i^j according to Equation (2), respectively

Compute p_i^j according to Equation (11)

If $p_i^j < Obj(pbest)$ $x_i^j \longrightarrow pbest_i$

If $p_i^j < Obj(gbest)$ $x_i^j \longrightarrow gbest_i$

End

End

4. Result Analysis

The details of the operating environment are given as follows.

Software. Windows 10 is the operating system; MATLAB 2016b is the development software.

Hardware. Intel(R) Core(TM) i7-4710HQ CPU@ 2.5GHZ; RAM 4.0GB.

Wind tunnel tests on the prototype of a long-span bridge have been carried out in the State Key Laboratory for Disaster Reduction in Civil Engineering at Tongji University. The experimental details can be referred to in the literature [9]. 20600 data points at wind speed $U = 9.1m/s$ for the case of damping ratio $\xi = 0.5\%$ are used to identify parameters of Zhu's model; 10000 data points are used to verify the validity of the identified parameters.

4.1. Comparison with Other PSOs. In this section, different algorithms are used to contrast with the NAPSO. These algorithms include FPSO, RPSO, APSO, and NLPSO.

In the PSOs, the swarm has 40 particles. The learning rates c_1 and c_2 are 2.2 and 1.7, respectively. The maximum number of iterations is 100. The inertia weight in FPSO is 0.8. For NLPSO, d, $w_{\max}$, and $w_{\min}$ are 3, 1, and 0.5, respectively. The search space is 7 dimensions, Y_1, Y_2, Y_3, ε, C_L, ω_{VS}, and ϕ.

These PSOs have been run 10 times, and their results are compared and analyzed using tables or figures. Figure 2 shows the change of fitness with the number of iterations. Different PSOs have different convergence rates and converge to different fitness values. The worst, mean, and best fitness values of the swarm in every iteration are shown in Figures 2(a), 2(b), and 2(c), respectively. As is indicated in Figure 2(a), FPSO converges to the maximum fitness; RPSO, APSO, and NLPSO converge to the same fitness which is lower than the fitness of FPSO but is higher than the fitness of NAPSO; NAPSO gets the minimum fitness at 84th iteration. In Figure 2(b), five algorithms converge to different fitness values,

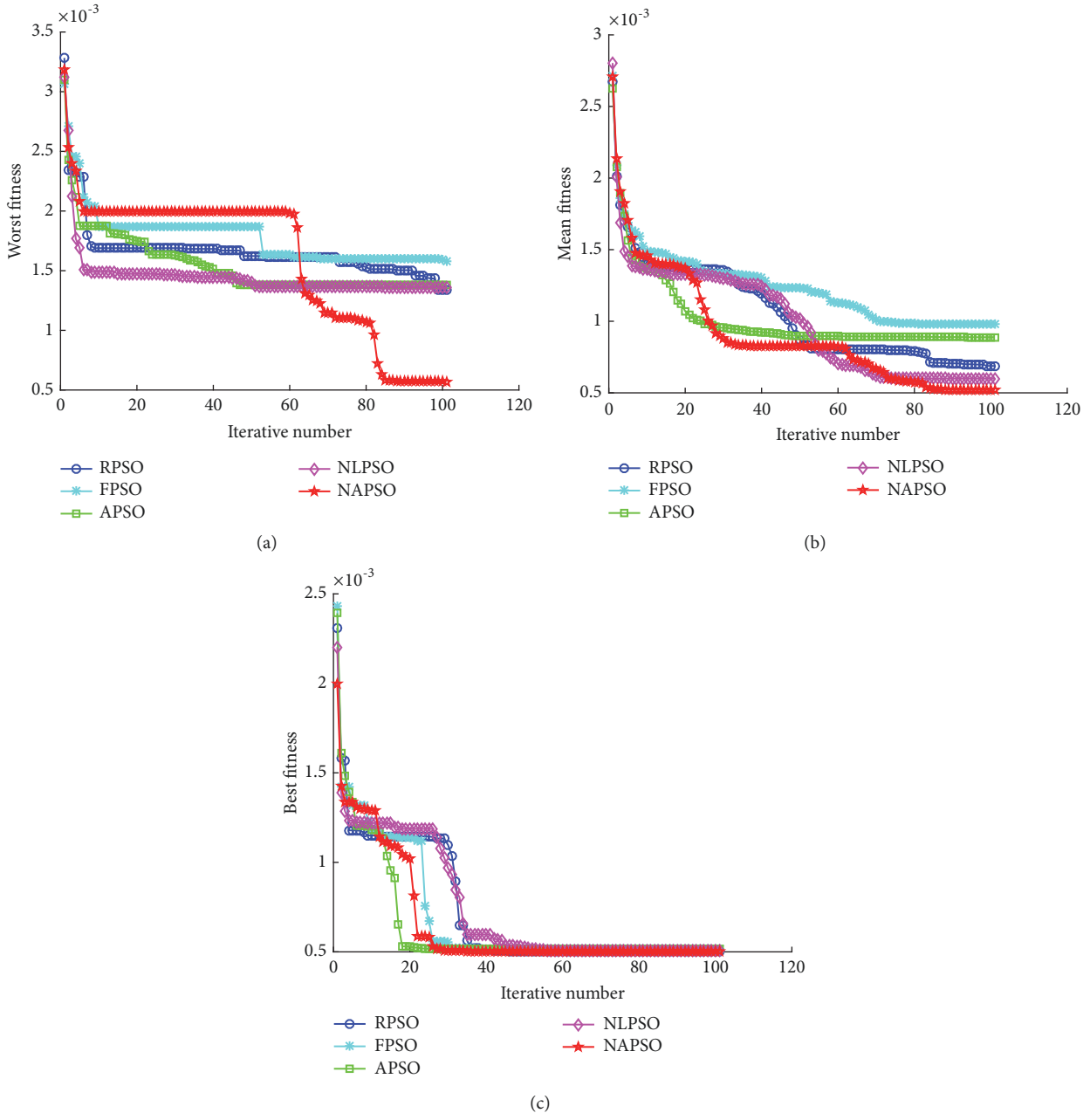

Figure 2: The variation of fitness for different algorithms. (a) Worst fitness; (b) mean fitness; (c) best fitness.

respectively. FPSO converges to the worst solution, followed by APSO, RPSO, and NLPSO. The NAPSO converges to the best solution at the 82nd iterations. As can be seen from Figure 2(c), PSOs converge to the same optimal solution at different rates of convergence. APSO is the fastest; NLPSO is the slowest; NAPSO is slower than APSO but faster than other PSOs.

Figure 3 illustrates that inertia weights of the 24th particle from different PSO algorithms vary with the number of iterations, respectively. RPSO has a random inertia weight that fluctuates sharply throughout the iterative process, and the quick change of inertia weight may bring about the instability of algorithm performance. FPSO has a constant inertia weight, 0.8. The inertia weight of APSO falls quickly and converges to the minimum inertia weight and then rebounds and slowly descends to the minimum inertia weight. The inertia weight of NLPSO nonlinearly decreases with the iterative number. For NAPSO, the inertia weight gets to the minimum value after two rapid drops. Figure 3 highlights that NAPSO adaptively adjusts the inertia weight of every particle in the pursuit of the optimal solution.

Figures 2 and 3 lead to the conclusion that each particle's own condition and the change of the inertia weight have great influence on the convergence. From the angle of convergence, FPSO is the worst due to the constant inertia weight. Although APSO, RPSO, and NLPSO have the change

Table 1: The results of 10 runs.

Algorithms	R^2			Run-time(s)		
	max	mean	min	max	mean	min
RPSO	0.8362	0.7761	0.5638	114.6517	112.3677	110.9941
FPSO	0.8324	0.6817	0.4856	114.2825	112.7262	111.1013
APSO	0.8317	0.7108	0.5509	117.0573	116.4893	115.7839
NLPSO	0.8357	0.8048	0.5580	115.1727	113.3600	111.4057
NAPSO	0.8365	0.8305	0.8146	112.9625	112.4936	112.0529

Table 2: Parameters in Zhu's model with different algorithms.

Algorithm	Y_1	ε	Y_2	Y_3	C_L	ω_{VS}	ϕ
GA	9.6967	55.3564	-2.1638	89.7142	0.0306	0.3908	0.5652
GALMA	11.3496	113.7826	-2.1644	89.9420	0.0274	0.3906	1.7688
NAPSO	11.3731	114.5930	-2.1647	89.9440	0.0024	0.3911	4.9534

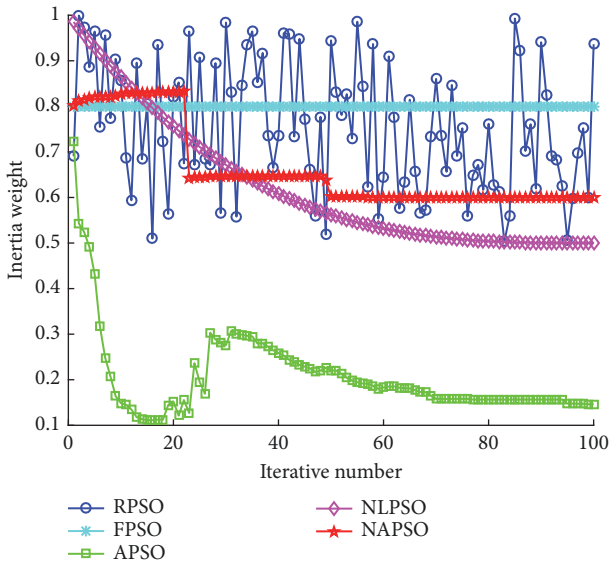

Figure 3: The change of inertia weight.

of inertia weight, those algorithms do not take each particle's own condition into account; their convergence is better than FPSO and worse than NAPSO. The most outstanding feature of NAPSO is that the inertia weight is automatically adjusted according to the current state of each particle. Therefore, NAPSO has better convergence than other PSOs.

Table 1 displays the run-time and R^2 of 10 runs of different algorithms. Due to the randomness of the initial solution, the result of each run is different. As is shown in Table 1, NAPSO has good stability in R^2 because its R^2 changes in a very small interval [0.8365 0.8146] and its run-time keeps around 112(s), while R^2 and run-time of other PSOs vary in a large interval. Therefore, Table 1 highlights that NAPSO has good stability compared with other PSOs.

In the pursuit of the globally optimal solution, every particle in NAPSO is able to adaptively adjust its own inertia weight according to the current fitness, so that it keeps the balance of local and global search abilities. It is because of this adaptive ability that NAPSO has good stability and high accuracy of identified parameters.

Table 3: R^2 and runtime of different methods.

	GA	GALMA	NAPSO
R^2	0.8343	0.8362	0.8363
Runtime (s)	113	123	112

4.2. Comparison with Other Non-PSOs. Through the above analysis, NAPSO is a good method to identify parameters of Zhu's model. In this selection, GA, GALMA, and NAPSO are carried out, respectively. For GA, every individual also has 7 features, the population size is 40, the maximum iterative number is 100, the lower bound is [-8 -18 -4 -100 -0.1 0.3 -5], and the upper bound is [20 130 6 100 0.1 0.4 5]. For LMA in GALMA, the maximum iterative number is 100, the lower bound is [-8 -18 -4 -100 -0.1 0.3 -5], and the upper bound is [20 130 6 100 0.1 0.4 5]. The identified parameters are shown in Table 2.

Table 3 shows R^2 and run-time of different algorithms. NAPSO has the highest R^2 and GA80 has the lowest. R^2 of GALMA approaches that of NAPSO. NAPSO spends least time and achieves the highest R^2. As far as Table 3 is concerned, the NAPSO takes the least time to get the optimal solution, so it outperforms other algorithms.

GA and PSO belong to the evolutionary algorithm, and they have their own features. GALMA pursues the better solution in the neighborhood of the initial solution generated by GA. GA and GALMA need the scope of solutions in advance. As far as Zhu's model is concerned, NAPSO has better convergence than GA.

4.3. Nonparameter Test for These Algorithms. R^2 is used to evaluate the advantages and disadvantages of these algorithms. Nonparameter tests for these algorithms include Wilcoxon's test and Friedman's test [26].

The results of Wilcoxon's test between NAPSO and other algorithms are shown in Table 4, where the values of R^+ and R^- of the test are specified, together with whether the

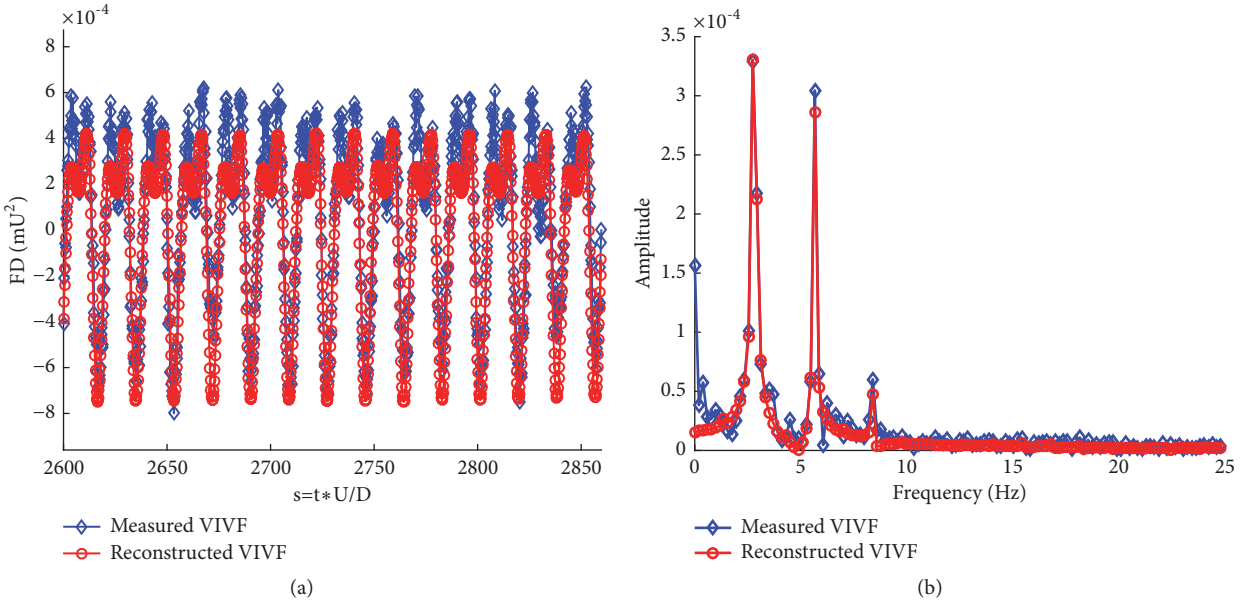

Figure 4: Comparison of the measured and reconstructed VIVFs. (a) Time domain. (b) Frequency domain.

Table 4: Wilcoxon's Test.

Algorithm	R^+	R^-	Hyp.α = 0.01	Hyp.α = 0.02	Hyp.α = 0.05	Hyp.α = 0.1
APSO	50	5	**A**	**R**	**R**	**R**
FPSO	47	8	**A**	**A**	**R**	**R**
GA	40	15	**A**	**A**	**A**	**A**
RPSO	39.5	15.5	**A**	**A**	**A**	**A**
GALMA	33	22	**A**	**A**	**A**	**A**
NLPSO	32	23	**A**	**A**	**A**	**A**

hypothesis for different levels of α are accepted (**A**) or rejected (**R**). In Table 4, the critical values of the Wilcoxon test are 3, 5, 8, and 10 for $\alpha = 0.01$, $\alpha = 0.02$, $\alpha = 0.05$, and $\alpha = 0.10$, respectively.

According to Table 4, NAPSO has the significant difference with respect to APSO or FPSO, whereas it has no significant difference with respect to other algorithms.

Friedman's test is a nonparametric test that uses rank to detect significant differences in the distribution of multiple paired samples. Table 5 shows the results of the Friedman's test. It can be found that the average ranking of NAPSO is the largest. In addition, P value computed by Friedman test is 0.0056, which is less than $\alpha = 0.05$. Thus the null hypothesis is rejected; these algorithms have significant difference.

According to the average ranking of the algorithms shown in Table 5, NAPSO is the best.

Table 5: Average ranking of the algorithms (Friedman).

Algorithm	Ranking
NAPSO	5.3
NLPSO	5.2
GALMA	4.9
RPSO	4
GA	3.5
FPSO	2.8
APSO	2.3

4.4. Simulation Test Based on NAPSO. These identified parameters are utilized to reconstruct the VIVF. Figure 4 shows a comparison of the measured and reconstructed VIVFs. As can be seen from Figure 4(a), both the measured and reconstructed VIVFs have nearly the same curve pattern and their difference lies in different peak values. For the reconstructed VIVF, one peak value is 2.5×10^{-4} and another is 4.4×10^{-4}, while these peak values of the measured VIVF change from 4.2×10^{-4} to 6.4×10^{-4}. All in all, the reconstructed VIVF depict the measured VIVF well.

Frequency spectrum analysis is utilized to catch the essence of a complex signal. Figure 4(b) illustrates spectra of different VIVFs. The measured VIVF has three distinct frequency points: f (2.8142Hz), $2f$ (5.6284Hz), and $3f$ (8.4426Hz). The reconstructed VIVF has the same spectrum as the measured one. Both the reconstructed and measured VIVFs have the same high amplitude at the 2.8142 Hz point; the reconstructed VIVF has lower amplitudes than the measured VIVF at 5.6284Hz and 8.4426Hz, respectively. The difference in spectra may result from the VIVF model. As a

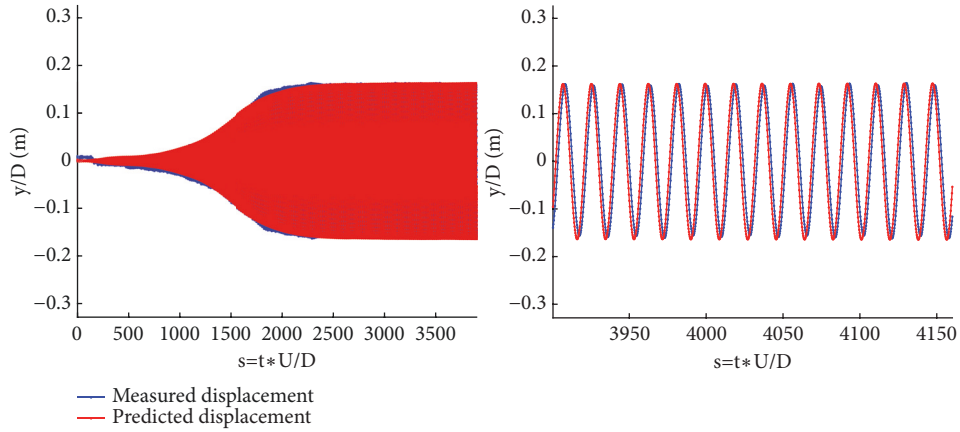

Figure 5: Comparison of the measured and predicted displacements.

whole, the reconstructed VIVF illustrates the measured VIVF well.

With the reconstructed VIVF, the time history of displacement response is predicted by the Newmark-β method at the wind speed 9.1m/s for the case of $\xi = 0.5\%$. To check the validity of identified parameters, a comparison of the measured and predicted displacements has been performed. In Figure 5, the time history of the reconstructed displacement is in accordance with that of the measured one. Hence, the identified parameters are valid.

5. Conclusions

The NAPSO is proposed to identify parameters of Zhu's nonlinear model. In NAPSO, each particle's situation in the swarm directly affects the adjustment of its own inertia weight and searching ability. If a particle has a large normalized fitness, its inertia weight becomes large, and it will search big area of the solution space in next iteration to pursuit a better solution. If a particle has a small fitness, its inertia weight will shrink, and it will approach the optimal solution with accuracy. This strategy ensures the stability of NAPSO.

There are comparisons of different algorithms in terms of convergence speed, fitting effect, and solution accuracy. Experimental results clearly show the NAPSO outperforms other algorithms. Most importantly, with the parameters identified by NAPSO, the vertical displacement of the structure is precisely predicted. Therefore, when an effective identification method is needed, the NAPSO is considered at first.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

Acknowledgments

Thanks are due to Dr. Xiaoliang Meng, Tongji University, for experimental data. This work has been partially supported by State Natural Science Fund project of China (Study on Hybrid Time-Frequency Analysis Method for Long-Span Bridge under Nonstationary Wind Load, Grant no. 51308330) and by 2015 Open Fund of Structure and Wind Tunnel Key Laboratory of Guangdong Province (Regularization Parameter Identification Method for Bridge Vortex Vibration model, no. 201502).

References

[1] F. Ehsan and R. H. Scanlan, "Vortex-induced vibrations of flexible bridges," *Journal of Engineering Mechanics*, vol. 116, no. 6, pp. 1392–1411, 1990.

[2] Y. Fujino and Y. Yoshida, "Wind-induced vibration and control of Trans-Tokyo Bay Crossing bridge," *Journal of Structural Engineering*, vol. 128, no. 8, pp. 1012–1025, 2002.

[3] H. G. Sung, H. Baek, S. Hong, and J.-S. Choi, "Numerical study of vortex-induced vibration of pivoted cylinders," *Ocean Engineering*, vol. 93, pp. 98–106, 2015.

[4] A. Larsen, S. Esdahl, J. E. Andersen, and T. Vejrum, "Storebaelt suspension bridge - vortex shedding excitation and mitigation by guide vanes," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 88, no. 2-3, pp. 283–296, 2000.

[5] A. M. Marra, C. Mannini, and G. Bartoli, "Van der Pol-type equation for modeling vortex-induced oscillations of bridge decks," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 99, no. 6-7, pp. 776–785, 2011.

[6] L.-D. Zhu, X.-L. Meng, L.-Q. Du, and M.-C. Ding, "A Simplified Nonlinear Model of Vertical Vortex-Induced Force on Box Decks for Predicting Stable Amplitudes of Vortex-Induced Vibrations," *Engineering Journal*, vol. 3, no. 6, pp. 854–862, 2017.

[7] A. Larsen, "A generalized model for assessment of vortex-induced vibrations of flexible structures," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 57, no. 2-3, pp. 281–294, 1995.

[8] T. Wu and A. Kareem, "Vortex-induced vibration of bridge decks: Volterra series-based model," *Journal of Engineering Mechanics*, vol. 139, no. 12, pp. 1831–1843, 2013.

[9] L.-D. Zhu, X.-L. Meng, and Z.-S. Guo, "Nonlinear mathematical model of vortex-induced vertical force on a flat closed-box bridge deck," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 122, pp. 69–82, 2013.

[10] H. Gupta, P. P. Sarkar, and K. C. Mehta, "Identification of vortex-induced-response parameters in time domain," *Journal of Engineering Mechanics*, vol. 122, no. 11, pp. 1031–1037, 1996.

[11] X. Y. Weng, G. E. Yao-Jun, and H. X. Chen, "New approach for identifying aerodynamic parameters in nonlinear vortex induced force model," *Journal of Vibration & Shock*, vol. 33, pp. 82–85+96, 2014.

[12] X. Tian, J. Yan, and Q. Zhou, "A novel method of parameter identification based on nonlinear empirical model for vortex-induced vibration," *Journal of Engineering Research*, vol. 5, no. 4, pp. 44–58, December 2017.

[13] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, Perth, Western Australia, November-December 1995.

[14] Y. Dong, J. Wang, F. Chen, Y. Hu, and Y. Deng, "Location of Facility Based on Simulated Annealing and "ZKW" Algorithms," *Mathematical Problems in Engineering*, vol. 2017, Article ID 4628501, 9 pages, 2017.

[15] N. Kherici and Y. Mohamed Ben Ali, "Using PSO for a walk of a biped robot," *Journal of Computational Science*, vol. 5, no. 5, pp. 743–749, 2014.

[16] T. Zhang and X. Gao, "Hybridizing particle swarm optimization with differential evolution solving constrained problems," *Microcomputer & Its Applications*, 2014.

[17] M. Fontan, A. Ndiaye, D. Breysse, F. Bos, and C. Fernandez, "Soil-structure interaction: parameters identification using particle swarm optimization," *Computers & Structures*, vol. 89, no. 17-18, pp. 1602–1614, 2011.

[18] D. Sendrescu, "Parameter Identification of Anaerobic Wastewater Treatment Bioprocesses Using Particle Swarm Optimization," *Mathematical Problems in Engineering*, vol. 2013, Article ID 103748, 8 pages, 2013.

[19] I. Khoja, T. Ladhari, A. Sakly, and F. M'Sahli, "Parameter Identification of an Activated Sludge Wastewater Treatment Process Based on Particle Swarm Optimization Method," *Mathematical Problems in Engineering*, vol. 2018, Article ID 7823930, 11 pages, 2018.

[20] M. Taherkhani and R. Safabakhsh, "A novel stability-based adaptive inertia weight for particle swarm optimization," *Applied Soft Computing*, vol. 38, pp. 281–295, 2016.

[21] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, pp. 1945–1950, Piscataway, NJ, USA, July 1999.

[22] W. Liao and J. Wang, "Nonlinear Inertia Weight Variation for Dynamic Adaptation in Particle Swarm Optimization," *Computers & Operations Research*, vol. 33, pp. 859–871, 2006.

[23] M. S. Arumugam and M. Rao, "On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems," *Applied Soft Computing*, vol. 8, no. 1, pp. 324–336, 2008.

[24] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation and IEEE World Congress on Computational Intelligence*, (Cat. No.98TH8360), pp. 69–73, Anchorage, Alaska, USA, May 1998.

[25] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 94–100, Seoul, Republic of Korea, May 2001.

[26] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.