Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 628619, 11 pages
http://dx.doi.org/10.1155/2014/628619

Research Article

Intelligent Platform for Model Updating in a Structural Health Monitoring System

Danhui Dan,[1] Tong Yang,[2] and Jiongxin Gong[1]

[1] *Department of Bridge Engineering, Tongji University, Room 711, Bridge Building, 1239 Siping Road, Shanghai 200092, China*
[2] *Lin Tung-Yen & Li Guo-Hao Consultants Shanghai Ltd., Shanghai 200092, China*

Correspondence should be addressed to Danhui Dan; dandanhui@tongji.edu.cn

Received 10 October 2013; Accepted 12 December 2013; Published 20 January 2014

Academic Editor: Ting-Hua Yi

The main aim of this study is to develop an automated smart software platform to improve the time-consuming and laborious process of model updating. We investigate the key techniques of model updating based on intelligent optimization algorithms, that is, accuracy judgment methods for basic finite element model, parameter choice theory based on sensitivity analysis, commonly used objective functions and their construction methods, particle swarm optimization, and other intelligent optimization algorithms. An intelligent model updating prototype software framework is developed using the commercial software systems ANSYS and MATLAB. A parameterized finite element modeling technique is proposed to suit different bridge types and different model updating requirements. An objective function library is built to fit different updating targets. Finally, two case studies are conducted to verify the feasibility of the techniques used by the proposed software platform.

1. Introduction

The use of monitoring information from structural health monitoring systems facilitates the updating of finite element models with real-time and online structures. By doing so, we can not only update the structural benchmark analyses model but also continuously track the changes in the target physical parameters and the characteristic index in any location [1–3]. The automation and intelligence of the model updating process is the key problem that hinders the achievement of these goals in structural health monitoring system [4, 5]. After 20 years of research, existing structural health monitoring systems at home and abroad have entered the third stage, where the emphasis is on the processing and utilization of data to implement data-based online early warnings and assessments of individual health status [5, 6]. During this stage, the model-based assessment method is simply regarded as a supplementary approach that is used in an offline manner and it has not become the main focus of the online algorithm. Model updating has been studied only to provide a benchmark model for this method [1, 6].

In recent years, the theory and technology of model updating have continued to progress, especially in computational model updating (CMU) and model validation [7, 8] research, some of the techniques used when updating parameter selection [9, 10], uncertainty processing techniques for updating results from continuous monitoring information [11, 12], and alternative technologies that employ small numbers of calculations of complex finite element calculation during a modified iteration step [13–16]. Model updating software has also made rapid progress, while some commercial finite element software systems have expanded their functions to model updating, such as DDS's FEMtools [17], the Balmes' structural dynamics toolbox SDTools [18], and Schedlinski's SysVal [19]. All of these software systems have model updating functions, but these software updating methods are isolated, with poor universality and weak poor optimization abilities. It is difficult to ensure the optimization of the updated results, which need to be programmed completely and automated to reduce human intervention. However, technological progress constantly stimulates structural health monitoring system researchers

to include model updating in the development of the next generation of structural health monitoring systems, thereby making the systems more intelligent with online and real-time updating processes. This will allow the use of physical and mechanical field information to address the monitoring target, thereby expanding the evaluation of sources and the scope of information, as well as overcoming the shortcomings caused by limited measurement points [19].

Thus, we studied online intelligent model update techniques based on monitoring to develop a model updating prototype software framework based on the commercial finite element software system, ANSYS, and the scientific computing software system, MATLAB. We also developed an intelligent algorithms library, which includes particle swarm optimization (PSO), to drive most of the updating tasks. The use of this platform to combine updating tasks, objective functions, and intelligent algorithms facilitates efficient and flexible automatic model updating without manual intervention, thereby bridging the functions of the structural finite element modeling and updating both in the structure and in component level with the present structural health monitoring system. Finally, we consider the key components of the technique developed in this study by discussing the results of a vibration test with a continuous beam and a monitoring system implementation of a cable-stayed bridge, which demonstrated the feasibility of the proposed technique.

2. Descriptions of the Generality of Parametric Model Updates

Based on the differences between the objects updated, model updating methods can be divided into two categories: matrix updating and parametric updating, the essence of which is an optimization algorithm that minimizes the residuals. Matrix updating methods modify the system matrix of structure or submatrices directly. However, the relations among elements may disappear because of the massive volumes of data in the matrix elements that need updating and the drastic manipulations required before and after matrix updating. There may be imaginary elements and negative stiffness values, which means that the elements lose their definite physical meaning, so these methods are not suitable for large structures. Starting in the late 1980s, the research focus shifted gradually to parametric updating methods, which regard the structural design parameters (such as the boundary conditions, physical properties, and geometric features) that constitute system matrixes or the abstract parameters as updating objects, thereby ensuring that the model has a definite physical meaning after updating. Although there are many parametric model updating methods with different characteristics, the core process can be described using the same model.

The number of design parameters in structure finite element models is n, where the first m is the parameters that need to be updated, so the design parameters can be expressed as

$$\mathbf{p} = [p_1, p_2, p_3, \ldots, p_m, \ldots, p_n]. \tag{1}$$

The system matrix of the structure (global stiffness matrix, mass matrix, and damping matrix) can be expressed as a function of the design parameters:

$$\begin{aligned} \mathbf{K} &= f_K(\mathbf{p}), \\ \mathbf{M} &= f_M(\mathbf{p}), \\ \mathbf{C} &= f_C(\mathbf{p}). \end{aligned} \tag{2}$$

In the corresponding finite element model, the experimental model also has the following relationship between the system matrix and design parameter values:

$$\begin{aligned} \mathbf{K}^* &= f_K^*(\mathbf{p}^*), \\ \mathbf{M}^* &= f_M^*(\mathbf{p}^*), \\ \mathbf{C}^* &= f_C^*(\mathbf{p}^*). \end{aligned} \tag{3}$$

The aim of updating is to ensure that the finite element model $(\mathbf{K}, \mathbf{M}, \mathbf{C})$ approximates the experimental model $(\mathbf{K}^*, \mathbf{M}^*, \mathbf{C}^*)$, so it can take advantage of the former to perform inversions or predictions of the mechanical field, or to perform system identification, as well as other tasks. The sources of updating are the differences between the two models, where the magnitude of the differences can be described comprehensively by an objective function

$$\begin{aligned} f &= F(\mathbf{K}, \mathbf{M}, \mathbf{C}, \mathbf{K}^*, \mathbf{M}^*, \mathbf{C}^*) \\ &= F(f_K(\mathbf{p}), f_M(\mathbf{p}), f_C(\mathbf{p}), f_K^*(\mathbf{p}^*), f_M^*(\mathbf{p}^*), f_C^*(\mathbf{p}^*)) \\ &= F_{f,p}(f_K, f_M, f_C, f_K^*, f_M^*, f_C^*, \mathbf{p}, \mathbf{p}^*). \end{aligned} \tag{4}$$

Therefore, model updating can be considered to be an optimization problem:

$$\begin{aligned} &(f_K, f_M, f_C, \mathbf{p})^{\text{opt}} \\ &= \underset{\substack{f_K, f_M, f_C \to f_K^*, f_M^*, f_C^* \\ \mathbf{p} \to \mathbf{p}^*}}{\arg\min} \left(F_{f,p}(f_K, f_M, f_C, f_K^*, f_M^*, f_C^*, \mathbf{p}, \mathbf{p}^*)\right). \end{aligned} \tag{5}$$

As mentioned above, the precondition of design parameter model updating is ensuring the rationality of the finite element analyses model, that is, ensuring a suitable approximation of (f_K, f_M, f_C) using (f_K^*, f_M^*, f_C^*). Otherwise, the results cannot be correct if a wrong finite element model is used for parameter updating. When modeling a finite element, the assumptions and approximations of structural geometry, materials, and boundary conditions have the main effects on rationality and the model structural error. The structural rationality of the finite element model can be guaranteed by ensuring the reasonable modeling of stiffness, mass, and damping and the rational processing of boundary conditions, rational modeling of loads, structural damage, and deterioration of performance. Thus, the objective functions and the optimization updating problem, respectively, can be simplified as

$$\begin{aligned} f &= F_p(\mathbf{p}, \mathbf{p}^*), \\ (f_K, f_M, f_C, \mathbf{p})^{\text{opt}} &= \underset{\mathbf{p} \to \mathbf{p}^*}{\arg\min} \left(F_p(\mathbf{p}, \mathbf{p}^*)\right). \end{aligned} \tag{6}$$

As a result, the problem of design parametric model updating can be described by (6). The general process can be summarized as follows.

(i) We regard the experimental model as the reference to model the finite element model reasonably, thereby ensuring the approximation of (f_K, f_M, f_C) as (f_K^*, f_M^*, f_C^*).

(ii) Depending on the aim of updating and the updating object's characteristics, the design objective function describe the differences in performance (static effect, dynamic response, or structural characteristics) between the finite element model and the experimental model.

(iii) An appropriate optimization algorithm is selected to find the optimal model.

Thus, it is clear that finite element model, the objective function, and the optimization algorithm are three main factors that affected design parameter model updating. Therefore, these three main factors must be standardized first before automating the updating process to ensure their seamless integration and data exchange, which is also the main focus of the present study.

3. Key Techniques for Intelligent Model Updating in an Online Environment

Finite element model updating research has made great progress and has been applied in some fields of engineering, but many problems still need to be solved when model updating is applied to large structures. The major constraints on online model updating are as follows: (1) model updating with large structures entails many degrees of freedom, intensive and complex calculations, and many iterative steps; (2) the limits of the iterative optimization algorithm and the difficulty of appropriate algorithm design, which mean that the optimization process has problems converging on an ideal solution; (3) the improper selection of the design variables or design space often leads to the optimization results losing their physical meaning; (4) single step iteration calculations are large and difficult to perform online and in real time; (5) some optimization steps are not completely programmed and may need manual intervention. These difficulties form a bottleneck that hinders model updating in an online environment. Therefore, we used the following techniques to overcome these difficulties.

3.1. Selection of Model Updating Parameters and Design of the Space Boundary Value. The selection of model parameters has a huge effect on model updating, because the number of parameters selected will affect the scale and speed of the optimization calculations directly, while the parameters selected will affect the physical meaning of the updating results and the pathological degree of the equation governing updating. After updating the parameter selection, the choice of the upper and lower bounds of the design space also needs to be reasonable. The choice should not vary greatly within the definition domain, which is only mathematically reasonable; otherwise the value may be unreasonable from a physical perspective and abnormal in the usual sense of the structure.

Therefore, the design parameters should be selected before the parameter sensitivity analyses based on the actual conditions of the updating object. A sensitivity analyses that considers the statistical properties of updating parameters can then be performed. Given that the updating target function is f and the updating parameter is p, the initial value of the updating parameter is p_0 and δ_p is the statistical variation coefficient of parameter p, so the sensitivity after considering parameter's statistical features can be defined as follows:

$$s = \delta_p \cdot \left.\frac{\partial f}{\partial p}\right|_{p=p_0}. \tag{7}$$

After comparing and selecting various parameters using (7), the parameters with high sensitivity but low dispersion can be ranked at the end of the optional parameter sequence, or even excluded, to ensure the correct physical meaning and the structural rationality of the results. The Monte Carlo method or bootstrap method can be used to make the same choices. Updating the boundaries of the parameters can be achieved using the initial value and the coefficient of variation, that is, $p_0(1 \pm \delta_p)$.

3.2. Hybrid Objective Function. In recent years, research progress in model updating techniques means that dynamic fingerprints have been used increasingly to reflect the dynamic characteristics of the structure, such as the flexibility matrix, strain mode, modal curvature, and frequency response function. These have been applied widely to model updating techniques with good results. However, there are numerous dynamic fingerprints and many different cases, which have created problems when deciding the success or failure of model updating, because it is difficult to choose an appropriate dynamic fingerprint as an objective function. During dynamic model updating, there may be various choices of objective functions, such as frequencies, mode shapes, modal flexibility, modal strain energy, frequency response function, and static displacement, which can all be used as the independent variables of objective functions. Various objective functions perform differently in diverse updating environment, but in hybrid objective functions generally perform better. In the present study, the key objective functions were as follows.

A type of objective function that usually combines natural frequencies and mode shapes is

$$f = \sum_{i=1}^{n} \alpha_i \left(\frac{\omega_i - \omega_i^*}{\omega_i^*} \right)^2 + \sum_{i=1}^{n} \beta_i \left(\frac{1 - \sqrt{\mathrm{MAC}_i}}{\mathrm{MAC}_i} \right)^2. \tag{8}$$

In (8), ω_i and ω_i^* are the ith order of the modal frequencies and the measured modal frequencies of the finite element, respectively, α_i and β_i are the weight coefficients of the ith order, and MAC_i is the ith order modal assurance criterion of the measured model and the finite element model.

Another objective function used in previous study [14] is the cross-model cross-mode (CMCM) model updating method:

$$f = \left\| \boldsymbol{\delta}_{\lambda} - [\mathbf{C} \quad \mathbf{E}] \right\|. \tag{9}$$

In (9), δ_j is the fluctuation ratio of the eigenvalue:

$$\delta_j = \frac{\lambda_j^* - \lambda_j}{\lambda_j} \tag{10}$$

λ_j^* and λ are the ith order eigenvalues of the measured mode and the finite element mode, respectively, and $\mathbf{C}$ and $\mathbf{E}$ are the values of the cross-correlation coefficient of stiffness (COK) and cross-correlation coefficient of mass (COM), which are defined as follows:

$$\mathbf{C} = \begin{bmatrix} C_{1,1} & \cdots & C_{1,N_e} \\ \vdots & C_{k,n} & \vdots \\ C_{nf\times nt,1} & \cdots & C_{nf\times nt,N_e} \end{bmatrix},$$

$$\mathbf{E} = \begin{bmatrix} E_{1,1} & \cdots & E_{1,N_e} \\ \vdots & E_{k,n} & \vdots \\ E_{nf\times nt,1} & \cdots & E_{nf\times nt,N_e} \end{bmatrix}, \tag{11}$$

$$C_{k,n} = \frac{\mathrm{COK}_{i,j^*}^{(n)}}{\mathrm{COK}_{i,j^*}} = \frac{(\boldsymbol{\varphi}_i)^T \mathbf{K}_n \boldsymbol{\varphi}_i^*}{(\boldsymbol{\varphi}_i)^T \mathbf{K} \boldsymbol{\varphi}_i^*},$$

$$E_{k,n} = \left(1+\delta_j\right) \frac{\mathrm{COM}_{i,j^*}^{(n)}}{\mathrm{COM}_{i,j^*}} = \left(1+\delta_j\right) \frac{(\boldsymbol{\varphi}_i)^T \mathbf{M}_n \boldsymbol{\varphi}_i^*}{(\boldsymbol{\varphi}_i)^T \mathbf{M} \boldsymbol{\varphi}_i^*}.$$

In the equations, $k = i \times j = 1, 2, \ldots, nf \times nt$, $n = 1, 2, \ldots, N_e$.

The two objective functions are the representation of the directly designed parametric updating method and the indirectly designed parametric updating method based on phenomenological theory, respectively.

3.3. Alternative Technique for Complex Finite Element Calculation Using Small Amount of Computation Methods in a Single Updating Iteration Step. The model updating process can be transformed into an optimization problem, which generally requires an iterative technique that calculates an objective function many times according to certain rules. For large complex structures, each iterative step requires one (linear) or more (nonlinear) finite element analyses to obtain the independent variables of the objective function, which can include static effects, dynamic responses, the modal parameters and their relevant derived quantities, and frequency response functions. Thus, each calculation has a huge computational load, which means that model updating is not suitable for performing online. To achieve real-time online updating, alternative techniques use small amount of calculations approaches to replace complex finite element calculation. The main methods include the response surface method, influence matrix method, neural network method, support vector machine (SVM) method, and model reduced method.

The response surface method and the influence matrix method determine the relationship between an explicit function formulated as a transformation of the linear matrix and between the structural response and the updating parameter, by simulating the calculations in advance, which replaces the finite element calculation using a function or the product of a matrix. The neural network and SVM methods simulate the complex functional relationship using intelligent algorithms. A common feature of all of these methods is that a large number of finite element calculation rounds are needed in advance, and then an alternative technique of small amounts of calculations is induced in the process of updating by self-learning algorithm or fitting methods mathematically.

The model reduced method does not require a large number of calculations because it replaces the finite element system with many degrees of freedom in the original calculation with the finite element analyses of a reduced model with few degrees of freedom. The transform matrix of the reduced model, T, is a function of the correction parameters, which is concerned with the iterative calculations. Additionally, the process to generate T is also a complex calculation process which needs large amount of calculations. In the present study, therefore, a dynamic reduced method based on Neumann series is proposed to overcome above mentioned troublesome.

3.4. Automation of the Optimization Algorithm. After constructing the objective function, the model updating problem is transformed into a constrained optimization problem. There are three main categories of methods for solving optimization problems: analytical methods, traditional numerical methods, and intelligent optimization algorithms. In general, if an analytical method is used to solve complex optimization problems, it is almost impossible to achieve the solution because the conditions of the problems cannot meet the requirements of the assumed conditions for the analytical solution. Therefore, numerical methods have many advantages when dealing with complex optimization problems. Some traditional numerical iteration algorithms include Newton's method, the conjugate gradient method, linear programming method, and nonlinear programming method. With multiobjective combinatorial optimization problems, however, the traditional numerical optimization algorithms also have difficulties. The intelligent optimization algorithms that have emerged since the 1980s, such as genetic algorithm, simulated annealing algorithm, ant colony algorithm, PSO, artificial immune algorithms, and hybrid optimization strategies, have been developed by simulating certain natural phenomena or processes to provide feasible solutions for combinatorial optimization problems that are difficult to solve using traditional optimization techniques. From the beginning of the intelligent optimization algorithms are presented, the applications cover all the field of civil engineering are done by researchers, Yi et al. used generalized genetic algorithm and modified monkey algorithm to treat the problem of optimal sensor placement for structural health monitoring [20, 21]. Xu et al. used neural networks method to treat the real-time computing problem in the semiactive control of structures [22, 23]. The PSO is also investigated and

applied on the automation FEM model updating problem by our team [24–26].

We compared most of the optimization algorithms for model updating and found that most algorithms were useful in specific conditions [21–23]. Building an intelligent platform for model updating to make full use of the optimization results can be helpful when selecting a traditional optimization algorithm (such as sequential quadratic programming method) or a modern intelligent optimization algorithm (such as particle swarm intelligent algorithm and evolutionary algorithm) to address different issues, which can make the model updating process more flexible and efficient, so it can be applied to engineering applications.

4. Components of an Intelligent Platform for Structural Health Monitoring System Model Updating

At present, most of the large finite element software systems that are used widely in engineering lack support for model updating, while some finite element software systems that have been developed specifically for model updating do not perform well with conventional finite element analyses, which makes it difficult for them to handle complex engineering structures and they have insufficient optimization ability. Based on the sensitivity analyses theory of model parameter updating and by combining the numerical calculation software MATLAB with the finite element modeling ability ANSYS, we developed an intelligent platform for model updating in a structural health monitoring system. Our system facilitates automated model updating for actual engineering structures because there is an urgent need for health monitoring systems at present. By extracting the general steps of the dynamic-based parametric bridge model updating process, we developed a common software architecture and implemented some key techniques, which should allow the subsequent production of large online bridge updating software.

The commonality of our proposed framework is mainly reflected in the following: the optimization algorithm and target function are modular, so different optimization algorithms and target functions can be selected for different problems; the system is extensible, so the optimization algorithm and target function can be added independently depending on needs, according to certain standards.

4.1. Data Exchange Interface and the Process Control between MATLAB and ANSYS. The ANSYS parametric design language (APDL), which is shipped with ANSYS, can be used for general process control, but it may be more difficult and very slow when implementing specific complex physical models. In the proposed framework, therefore, MATLAB is responsible for the optimization and control process, whereas ANSYS is responsible for finite element calculation. The data generated by MATLAB's optimization module are passed to ANSYS for structure analyses and ANSYS then returns the updating data to MATLAB's optimization module. This cycle is repeated until the results converge. During this process, to ensure the collaboration between MATLAB and ANSYS, there needs to be smooth transmissions among three types of information: call instructions, data interactions, and the interactions among running state.

There is no ready-made interface between MATLAB and ANSYS, but MATLAB provides some universal methods for calling external programs, such as "!," the DOS function, and the system function. "!" and the DOS function can call a shell program to execute a command required by the Windows system, while the system function can perform operating system commands and return the results.

There is also no direct data exchange system between MATLAB and ANSYS, so the exchange of data must occur via files. We used text files as the medium of data exchange. Text files are simple but they have a high data storage capacity. Both software systems provide suitable functions for reading and writing text files.

Implementing the transfer of status information between two types of software is the key to process control. We achieved the mutual notification of status information by setting up a state variables identification function (flag M). This function judges whether ANSYS is operating by reading the status value in the MATLAB program. Thus, if ANSYS stops running, the flag will be zero and it notifies MATLAB that the current run is over so the next step of the calculation can be run; otherwise MATLAB will wait. The format uses the system or DOS functions to call ANSYS as follows:

```
[status,result]
=system ("D:\Ansys\v110\ANSYS\bin\intel
  \anss110 –b -i ansys/ansysconcle
  -o output.out")
```

After the ANSYS performs the calculation, the status returns a value of 0. Thus, the returned value of the status can be used to write a flag file, as follows:

```
function flag(status)
if status==0

    out=objfunt4;
    %objfunt4 is the objective function

end
```

4.2. Parametric Finite Element Modeling Solution. The parametric finite element technique refers to the definition of the finite element model structure using a group of undetermined parameters, where the parameters can be the structural geometry size and physical and mechanical properties. During parametric model updating design, the parameters that need to be updated constitute the undetermined parameters of the parameterized finite element model. ANSYS provides a parameter design language (APDL), which is used to complete the finite element analyses automatically. APDL's command is a type of script command, which can provide users with parameters, vectors, cycles, and a series of functions, while a batched analyses technique is also provided by ANSYS.

The basic steps of the parametric finite element analyses method are as follows. First, abstract the characteristic parameters of the described model according to the geometrical structure of the model and simplify them as appropriate. Second, establish the finite element analyses process, including entity modeling, analyses, and results treatment processes, using the command stream file in ANSYS. Third, replace the parameters in the model with abstracted characteristic parameters by APDL, which constitutes the finite element analyses process with variable parameters. Finally, depending on the requirements of the design and analyses, provide specific characteristic values to the parameters and conduct the finite element analyses to obtain the results. The main difference between parametric finite element analyses and general finite element analyses is that the preprocessing geometry model, material property, loading, and boundary conditions are parametric and the parameters must be assigned values before the calculations.

Depending on the basic flow of the parametric finite element analyses, we divide the general finite element model into the corresponding modules shown in Figure 1. Ansysconsole is the main control module of the model, which is responsible for the finite element analyses process control. Text file Inf and Para are the initial parameter files of parametric model, where Inf records the model's calculation type, basic parameters, and so forth and Para records the trial data generated by the optimization algorithm. MCF and MCS are the result files for the finite element analyses, where the former records the modal frequency and the latter records the modal shape. These four files comprise the data module of the model and they are the key components of the interaction with MATLAB, that is, Model, Loading, D&D, and BC form the basic model file, where Model is the initial model, Loading simulates the load conditions, D&D can simulate the damage and performance degradation of the structure, and BC is the model boundary condition. Specific modules can also be added to the fixed format. Finally, Cal_m is the computing core of the model that handles the model analyses.

4.3. Objective Function Library Design. An objective function is a metric used to describe the level of difference between theoretical and experimental model features, which comprised the geometrical and mechanical parameters of the structure. Optimization processes are error-minimizing processes that are applied to the experimental model and the theoretical model. The objective functions used for model updating include static-based objective functions, dynamic-based objective functions, and static and dynamic combined objective functions.

Method for Calling an Objective Function. Because the data generated by the intelligent algorithms are the initial parameters that need to be updated and the data needed by the objective function are the corresponding responses of the structure, an interface function, called parafinder, must be established between the intelligent algorithm and objective function. Its main functions are passing the parameters provided by master program, passing the parameters provided by the optimization algorithm, preliminary processing of the parameters provided by the optimization algorithm, selecting an objective function, and returning objective function values to the optimization algorithm. The method used to call an objective function is shown in Figure 2.

Objective Function Library. Different objective functions are calculated depending on the parameters transferred from parafinder. Each objective function has its own name and the main functions include obtaining the parameters passed by the master program, obtaining the data produced by the optimization algorithm, and calculating the objective function value. Based on the commonly used objective functions for bridge model updating to produce an objective function library with a standard interface, we developed the objective function library table shown in Table 1. The standard input and output of each objective function are shown in Table 2. In the table, f_{ai} and f_{ti} represent the calculated frequency and test frequency, respectively, ϕ_{ai} and ϕ_{ti} represent the calculated modal vector and test modal vector, respectively, and u_{ai} and u_{ti} represent the calculated static displacement and the measured displacement, respectively.

4.4. Intelligent Algorithm Library Design. After establishing the objective function, the model updating problem is transformed into a constrained optimization problem with multiobjective functions. In this section, the main problem that needs to be solved is developing a method that addresses the model updating problem, that is, the optimization algorithm. MATLAB provides intelligent algorithm libraries, which can be used as tools by the model updating software. We used PSO as a possible method for the intelligent optimization algorithm class library.

The PSO Intelligent Algorithm. The PSO algorithm was inspired by population behavior and has been used to solve optimization problems. In the algorithm, each case represents a potential solution to the problem, where each particle has a corresponding fitness value that is determined by s fitness function. The velocity of a particle determines the direction and distance of the particle's movement, where the velocity is adjusted dynamically based on the particle's experience, thereby allowing individual optimization in the solution space. The calling format for the PSO algorithm is as follows:

```
d_shown=[]; %PSO drawing parameters.
D=; % The dimension of the input.
mv=60; % The largest particle velocity.
VR=[];% Hunting zone.
PSO = [10 300 20 3 3 0.9 0.9 100 1e-25
        2000 1e-5 1 1 1e-5 10 10.0];
% Basic parameters of PSO.
seed=;%Original value.
[optOUT,tr,te,bestpos]
=pso_Trelea_vectorized
("parafinder",D,d_shown,mv,VR,0,PSO,
   "goplotpso",seed);
```

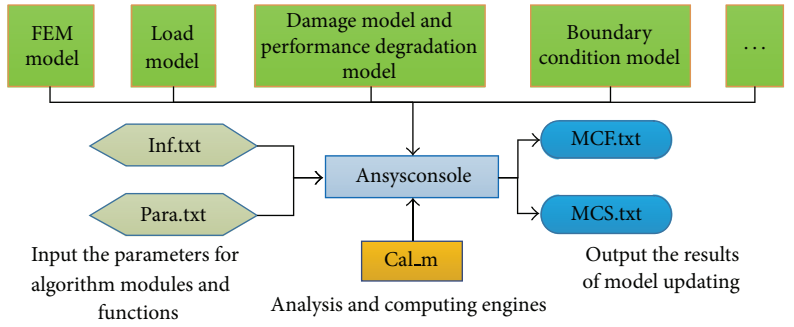

Figure 1: Schematic of the parameterized FE model.

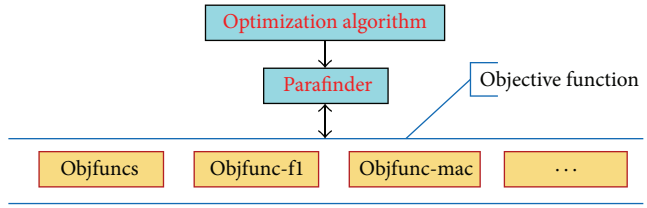

Figure 2: Invoking a routine using the target functions library.

Table 1: Library of target functions.

Objective function type	Features	Name	Function name
Static-based		Static objective function	Objfuns
Static and dynamic based		Static and dynamic objective function	Objfunsd
Dynamic-based	Transmission characteristics	Frequency	Objfund-f1
			Objfund-f2
		Mode of vibration	Objfund-mac1
			Objfund-mac2
		Strain mode	Objfund-strain
		Frequency-response function	Objfund-response
	Transmission curvature	Modal curvature	Objfund-mcurvature
		Flexibility curvature	Objfund-fcurvature
	Characteristic parameter	Frequency band energy spectrum	Objfund-spectrum
		Subband energy spectrum	Objfund-subspectrum
	Complex function	Joint frequency and mode shape	Objfund-fmac
		Flexibility matrix	Objfund-fmatrix
		Modal strain energy	Objfund-starinenergy

Table 2: Input-output target functions library.

Objective function	Remarks	Input	Output
Objfun-df1	Frequency change square ratio	f_{ai}, f_{ti}	Depends on its definition [20]
Objfun-df2	Frequency change square ratio	f_{ai}, f_{ti}	Ditto
Objfun-mac1	Modal assurance criterion	ϕ_{ai}, ϕ_{ti}	Ditto
Objfun-mac2	Modal assurance criterion	ϕ_{ai}, ϕ_{ti}	Ditto
Objfun-fmac	Joint frequency and mode shape	$f_{ai}, f_{ti}, \phi_{ai}, \phi_{ti}$	Ditto
Objfuns	Static objective function	u_{ai}, u_{ti}	Ditto
Objfunsd	Static and dynamic based	$u_{ai}, u_{ti} f_{ai}, f_{ti}, \phi_{ai}, \phi_{ti}$	Ditto

Table 3: Library of intelligent optimization algorithms.

Type	Name	Interface	Input	Output
Traditional algorithm	Sequential quadratic programming	byfmincon	**a, b**	X_N
Intelligent algorithm	Particle swarm optimization	Bypso	**a, b, c, d, e**	$A_{M\times N}$
	Genetic algorithm	Byga	**a, c, f, g, h, i**	$A_{M\times N}$
	Artificial fish-swarm algorithm	Byaf	**a, c, j, k, l, m**	$A_{M\times N}$
	Ant colony algorithm	Byaca	**a, s, n, o, p**	$A_{M\times N}$
	Immune algorithms	Byia	**a, c, h, i, q, r, s**	$A_{M\times N}$
	Annealing algorithm	Bysa	**t, u, v, w**	$A_{M\times N}$

Parafinder is an interface function between the PSO algorithm and the objective function library.

Intelligent Optimization Algorithm Library. Based on the calling method and the method used to establish the interface function with the PSO algorithm, the standard programming library including other commonly used intelligent optimization algorithms is shown in Table 3.

In Table 3, the parameters of the output X_N, which is an N-dimensional trial vector where N is the number of parameters, are as follows: $A_{M\times N}$ represents an $M * N$ trial matrix, M represents the size of the population, and N represents the number of parameters. The input parameters are as follows: a represents the maximum number of iterations, b represents the convergence error, c represents the size of the population, d represents the algorithm type, e represents the disturbed value, f represents the individual length, g represents the generation gap, h is the crossover probability, i is the mutation probability, j represents the largest feed testing number, k represents perception distance, l represents the crowding degree factor, m represents the moving step, n represents the importance factor of a pheromone, o represents the importance factor of an inspiration function, p represents the pheromone volatilization factor, q represents the volume of a memory database, r represents the evaluation parameters of diversity, s represents the number of distribution centers, t represents the cooling rate, u represents the initial temperature, v represents the termination temperature, and w represents the chain length.

4.5. Technical Proposal and Prototype Software for the Model Updating Platform. After choosing appropriate updating parameters, it is necessary to enter the updating module, where the parameters produced by the optimization algorithm are subjected to finite element analyses to determine the physical quantities that correspond to the measured values. The objective function value is calculated to determine whether the model error has been minimized and to determine whether the iterations should continue; otherwise the updating process is over. The basic flowchart of this process is shown in Figure 3.

Based on processes, the prototype of the overall software framework was implemented using MATLAB in a prototype program. ANSYS was readily implemented in the MATLAB environment to combine the model, objective function, and optimization algorithm, thereby implementing the overall process of automated model updating.

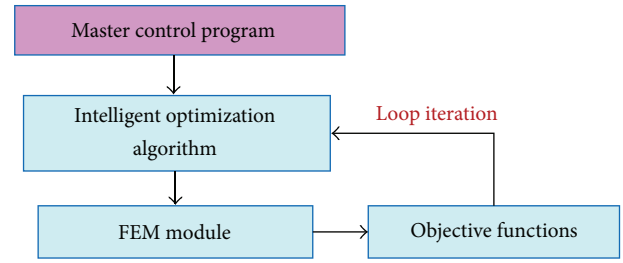

Figure 3: Flowchart of the software system for intelligent model updating.

Table 4: Parameters to be updated.

Num.	Parameter	Initial value	Upper limit	Lower limit
1	(E) Mpa	3.2×10^4	3.6×10^4	2×10^4
2	(D) kg/m^3	2.5×10^3	2.6×10^3	2.2×10^3
3	(K_1) N·m/rad	1000	1×10^6	0
4	(K_2) N·m/rad	1000	1×10^6	0
5	(K_3) N·m/rad	1000	1×10^6	0
6	(K_4) N·m/rad	1000	1×10^6	0

5. Verification of the Validity of the Prototype Software

5.1. Offline Verification of the Prototype Software Intelligent Updating Platform: Continuous Beam Experiments. The test model was a three-span reinforced concrete continuous beam structure, where the three spans were 1000, 1700, and 1700 mm, respectively, as shown in Figure 4. At the bottom of the continuous beam, there were five positions, that is, A, B, C, D, and E, which were installed with acceleration sensors. The test was a single-point excitation method and the sampling frequency was 512 Hz.

Before model updating, the parameters with greater sensitivity should be selected as the parameters for updating. Based on the structure, the parameters selected are shown in Table 4.

In Table 4, E is the elastic modulus, D is the density, and K_i is the rotational stiffness of the ith bearing.

Based on a model test where the physical quantities were measured, the objective function given in (8) was selected as the m function, Objfund-fmac, from the objective function library. After selecting the PSO algorithm from the intelligent optimization algorithms library and configuring

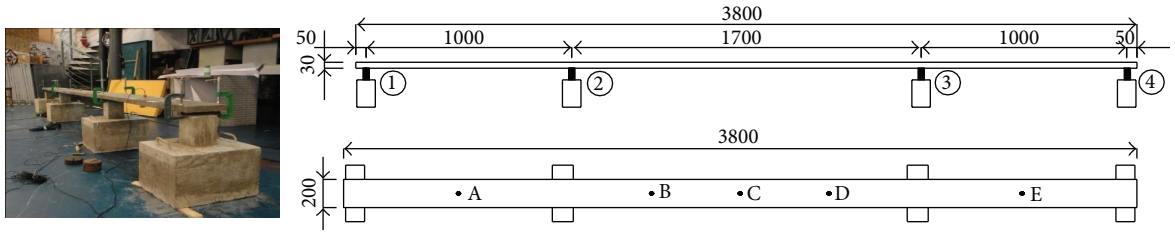

Figure 4: Three-span continuous beam test.

Table 5: Parameters for updating.

Num.	Parameter	Original values	Updating values
1	(E) Mpa	3.2e4	2.67e4
2	(D) kg/m^3	2.5e3	2.53e3
3	(K_1) N·m/rad	1000	48320.33
4	(K_2) N·m/rad	1000	2448.81
5	(K_3) N·m/rad	1000	87109.14
6	(K_4) N·m/rad	1000	233.8

the appropriate operating parameters, the m script automatically performed the task successfully. The objective function approximately reached convergence and the convergence process is shown in Figure 5. The parameters searched are shown in Table 5. A comparison of the model parameters before and after updating is shown in Table 6.

Table 6 shows that the updated frequency was very close to the measured frequency and the testing frequency error was below 1%, with the exception of the third degree. All four MAC values were above 0.9, which shows that the updated model correlated well with the measured model, and this also demonstrated the effectiveness of the updating results. The updating process was simple to write in the MATLAB script where the main contents of the script were model selection, the objective function and optimization algorithm, configuring the input parameters of the calling functions, calling ANSYS, monitoring ANSYS performance, and finally outputting the calculated results file. After launching, the whole process ran automatically and there were no requirements for manual intervention. This example demonstrates the feasibility of our proposed automatic model updating software framework.

5.2. Online Verification of the Prototype Software Intelligent Updating Platform: Cable Force Monitoring. In the previous example, the feasibility of automated model updating was verified offline with measured data. However, it is more challenging and useful to update the structure online using ongoing experimental data. Therefore, we considered the selection of real-time cable monitoring devices on a cable-stayed bridge, which was equipped with structure monitoring system. We connected the prototype software to the monitoring system using a MATLAB program that we developed, which read the data from the remote web-based bridge monitoring system [27], established a web connection to a specific cable acceleration channel, captured the cable

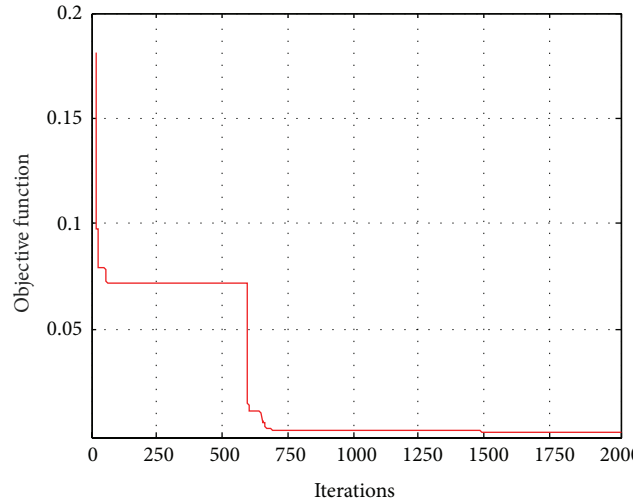

Figure 5: Convergence curve of the objective function.

acceleration records automatically using a compiled m script, identified the cable vibration frequency automatically, called the ANSYS program automatically for model analyses by building a finite element model, inputted the measured frequency and the calculated frequency into the objective function automatically, selected the PSO algorithm from the intelligent optimization algorithm library, automatically updated the cable model online, selected the cable force and bending stiffness as the updating parameters, and finally provided the updated results.

Based on the parameters of the cable obtained from the design data, these parameters were given corresponding input variables when calling the parameterized finite element model. The PSO algorithm was selected for updating and the objective function approximately converged after 20 iteration steps. The results for the cable force and bending stiffness are shown in Table 7, where the reference value of the cable force was obtained using the frequency formula and the error between the identified cable force and the calculated cable force was 3.3%. EI_0 represents the identified bending stiffness, while EI_1 and EI_2 represent the bending stiffness when the steel cable was treated as a fully bonding model and as a no binding model, respectively. A comparison of the frequencies derived from model updating and the identified frequencies is shown in Table 8. The curve of the updating process objective function is shown in Figure 6.

As shown in Table 8, the frequency sequences calculated by the automatic model updating prototype software and

Table 6: Frequencies and modal shape parameters after updating.

Degree	Test frequency (Hz)	Original frequency (Hz)	Updating frequency (Hz)	Error before updating (%)	Error after updating (%)	MAC before updating	MAC after updating
1	26.38	25.02	26.26	4.76	0.5	0.995	0.98
2	57.68	54.75	58.37	5.06	1.2	0.937	0.989
3	65.22	64.54	65.15	1.05	0.1	0.955	0.979
4	88.49	93.11	87.89	5.22	0.7	0.817	0.901

Table 7: Identified cable forces and flexible stiffness.

/	Cable forces (kN)	/	Flexible stiffness (kN·m^2)
Updated value	2900	EI_0	486
Reference value	3000	EI_1	511
Error (%)	3.3	EI_2	3.4

Table 8: Frequencies after updating.

Order	Measured frequencies (Hz)	Updated frequency (Hz)	Error (%)
1	1.27	1.304	2.61
2	2.637	2.609	1.07
3	3.906	3.915	0.23
4	5.273	5.223	0.96
5	6.543	6.533	0.15
6	7.91	7.848	0.79
7	9.18	9.166	0.15

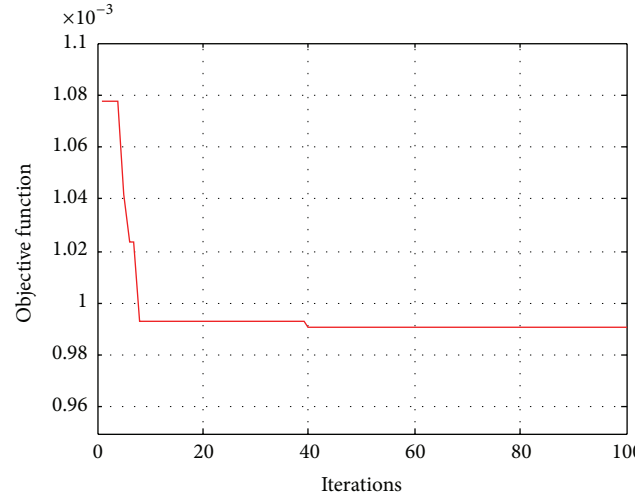

Figure 6: Convergence curve for the target function.

the measured frequency sequences were approximately the same; that is, maximum error was 2.61% and the bending stiffness estimate was also reasonable. These results indicate that the proposed model updating prototype software also performs well and reliably in an online environment, which is important for its further development.

6. Conclusions and Prospects

In this study, we developed an intelligent model updating prototype software framework platform, which is based on the commercial finite element software, ANSYS, and the scientific computing software, MATLAB. The proposed parameterized ANSYS finite element modeling scheme is suitable for updating different bridge structure models. We built a standard objective function library for different updating targets. To automate the overall updating process, we constructed an intelligent algorithms library, which included PSO, for different types of updating tasks. The intelligent algorithms library is the driving force of the automated updating process. After careful planning, it can allow the combination of bridge and component models, objective functions, and intelligent algorithms to provide a flexible, efficient, and universal bridge structure model updating platform for online monitoring environments. To illustrate the technical feasibility of our proposed scheme, we implemented the main technical links, which we tested using data from a continuous beam vibrations test and a cable-stayed bridge monitoring system. The results demonstrated the feasibility of our method for monitoring-based online intelligent model updating.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This project was supported by The Project of National Key Technology R&D Program in the 12th Five Year Plan of China (Grant no. 2012BAJ11B01).

References

[1] C. R. Farrar, F. M. Hemez, D. D. Shunk, D. W. Stinemates, B. R. Nadler, and J. J. Czarnecki, *A Review of Structural Health Monitoring Literature: 1996–2001*, Los Alamos National Laboratory, Los Alamos, NM, USA, 2004.

[2] K. H. Hsieh, M. W. Halling, and P. J. Barr, "Overview of vibrational structural health monitoring with representative case studies," *Journal of Bridge Engineering*, vol. 11, no. 6, pp. 707–715, 2006.

[3] J. Ou, "Research and practice of intelligent sensing technologies in civil structural health monitoring in The Mainland of China," in *Nondestructive Evaulation for Health Monitoring and*

Diagnostics, p. 61761D, International Society for Optics and Photonics, 2006.

[4] L. Sun, D. Dan, Z. Sun, and Q. Yue, "Health monitoring system for a cross-sea bridge in Shanghai," *IABSE Symposium Report*, vol. 92, no. 28, pp. 9–16, 2006.

[5] Z.-D. Xu, M. Liu, Z. Wu, and X. Zeng, "Energy damage detection strategy based on strain responses for long-span bridge structures," *Journal of Bridge Engineering*, vol. 16, no. 5, pp. 644–652, 2011.

[6] T.-H. Yi, H.-N. Li, and M. Gu, "Recent research and applications of GPS-based monitoring technology for high-rise structures," *Structural Control and Health Monitoring*, vol. 20, no. 5, pp. 649–670, 2012.

[7] J. E. Mottershead and M. I. Friswell, "Model updating in structural dynamics: a survey," *Journal of Sound and Vibration*, vol. 167, no. 2, pp. 347–375, 1993.

[8] B. A. Zárate and J. M. Caicedo, "Finite element model updating: multiple alternatives," *Engineering Structures*, vol. 30, no. 12, pp. 3724–3730, 2008.

[9] G.-H. Kim and Y.-S. Park, "An improved updating parameter selection method and finite element model update using multiobjective optimisation technique," *Mechanical Systems and Signal Processing*, vol. 18, no. 1, pp. 59–78, 2004.

[10] N. Abu Husain, H. Haddad Khodaparast, and H. Ouyang, "Parameter selection and stochastic model updating using perturbation methods with parameter weighting matrix assignment," *Mechanical Systems and Signal Processing*, vol. 32, pp. 135–152, 2012.

[11] A. Deraemaeker, P. Ladevèze, and P. Leconte, "Reduced bases for model updating in structural dynamics based on constitutive relation error," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 21-22, pp. 2427–2444, 2002.

[12] D. Göge and M. Link, "Assessment of computational model updating procedures with regard to model validation," *Aerospace Science and Technology*, vol. 7, no. 1, pp. 47–61, 2003.

[13] J. Carvalho, B. N. Datta, A. Gupta, and M. Lagadapati, "A direct method for model updating with incomplete measured data and without spurious modes," *Mechanical Systems and Signal Processing*, vol. 21, no. 7, pp. 2715–2731, 2007.

[14] W.-M. Li and J.-Z. Hong, "New iterative method for model updating based on model reduction," *Mechanical Systems and Signal Processing*, vol. 25, no. 1, pp. 180–192, 2011.

[15] W.-X. Ren and H.-B. Chen, "Finite element model updating in structural dynamics by using the response surface method," *Engineering Structures*, vol. 32, no. 8, pp. 2455–2465, 2010.

[16] Y. Lu and Z. Tu, "A two-level neural network approach for dynamic FE model updating including damping," *Journal of Sound and Vibration*, vol. 275, no. 3–5, pp. 931–952, 2004.

[17] http://www.femtools.com/courses/updating.htm.

[18] http://www.sdtools.com/sdt/index.html.

[19] W. Song and S. Dyke, "Development of a cyber-physical experimental platform for real-time dynamic model updating," *Mechanical Systems and Signal Processing*, vol. 37, no. 1-2, pp. 388–402, 2013.

[20] T.-H. Yi, H.-N. Li, and M. Gu, "Optimal sensor placement for structural health monitoring based on multiple optimization strategies," *Structural Design of Tall and Special Buildings*, vol. 20, no. 7, pp. 881–900, 2011.

[21] T. H. Yi, H. N. Li, and X. D. Zhang, "A modified monkey algorithm for optimal sensor placement in structural health monitoring," *Smart Materials and Structures*, vol. 21, no. 10, Article ID 105033, 2012.

[22] Z.-D. Xu, Y.-P. Shen, and Y.-Q. Guo, "Semi-active control of structures incorporated with magnetorheological dampers using neutral networks," *Smart Materials and Structures*, vol. 12, article 80, 2003.

[23] Z.-D. Xu, "Earthquake mitigation study on viscoelastic dampers for reinforced concrete structures," *Journal of Vibration and Control*, vol. 13, no. 1, pp. 29–43, 2007.

[24] T. Yang, *A study on the model updating techniques under the health monitoring environment. [Dissertation for master degree]*, Tongji University, 2012, (Chinese).

[25] P. Zhang, *PSO based model updating on cables and its tension identification. [Dissertation for master degree]*, Tongji University, 2010, (Chinese).

[26] Y. Chen, *PSO based cable damper system parameter identification. [Dissertation for master degree]*, Tongji University, 2012, (Chinese).

[27] D. Dan, L. Sun, and Y. Wang, "Applying online identification on Donghai Bridge anywhere, anytime, and anyway," in *Proceedings of the 6th World Forum on Smart Materials and Smart Structures Technology (SMSST '07)*, vol. 1, pp. 677–684, May 2007.