ELSEVIER

Computers in Industry 46 (2001) 167–179

COMPUTERS IN INDUSTRY

www.elsevier.com/locate/compind

Weight reduction of aluminum disc wheels under fatigue constraints using a sequential neural network approximation method

Yeh-Liang Hsu, Ming-Sho Hsu*

Department of Mechanical Engineering, Yuan Ze University, 135 Yuan-Tung Rd., Chung-Li 320, Taiwan, ROC

Received 27 January 2000; accepted 12 April 2001

Abstract

This paper describes a weight reduction problem of aluminum disc wheels under cornering fatigue constraints. It is a special structural optimization problem because of the existence of the implicit fatigue constraint. A sequential neural network approximation method is presented to solve this type of discrete–variable engineering optimization problems. First a back-propagation neural network is trained to simulate the feasible domain formed by the implicit constraints using just a few training data. A search algorithm then searches for the ''optimal point'' in the feasible domain simulated by the neural network. This new design point is checked against the true implicit constraints to see whether it is feasible, and the new training data is then added to the training set. This process continues in an iterative manner until we get the same design point repeatedly and no new training point is generated. In each iteration, only one evaluation of the implicit constraints is needed to see whether the current design point is feasible. No precise function value or sensitivity calculation is required. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Structural optimization; Neural network; Implicit constraint; Fatigue

1. Introduction

Disc wheels intended for normal use on passenger cars have to pass three types of tests before going into production: (1) the dynamic cornering fatigue test; (2) the dynamic radial fatigue test; and (3) the impact test. From the statistics of local wheel manufacturers, the dynamic cornering fatigue test has the highest failure rate among the three tests.

On the other hand, to provide better riding comfort, weight reduction is always an important concern to the design of disc wheels for passenger cars. For wheel manufacturers, reduction in weight also means reduction in material cost. Therefore, wheel manufacturers often try to reduce the weight of wheels as much as possible, while the wheels are still able to pass the required structural performance tests.

To the authors' knowledge, few researchers paid special attention on fatigue prediction and weight reduction of aluminum disc wheels. Karandikar and Fuchs [1] developed a computer-based system for predicting the fatigue life of wheels. This system is comprised of a CAD package, a finite element analysis

* Corresponding author. Tel.: +886-3-4638800; fax: +886-3-4558013.
E-mail address: mehsu@saturn.yzu.edu.tw (M.-S. Hsu).

0166-3615/01/$ – see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0166-3615(01)00125-7

program, and a fatigue life computation program. It can successfully simulate the dynamic corner fatigue tests and the development time of new wheels can be shorten using this system.

Kaumle and Schnell [2] also developed a technique for fatigue testing using a suitable rapid-prototyping system. A light metal wheel is used to demonstrate this technique. On the rapid-prototyping wheel the stresses will be measured, areas with high stress can be detected early, and it is possible to eliminate these areas. The fatigue behavior will be tested with the optimized rapid-prototyping wheel.

Marron and Teracher [3] emphasized on reducing the weight of wheel discs. They examined different high-strength sheet steel grades and showed that considerable weight reduction can be obtained by designing the stamping sequence in order to use the steel grade at its maximum without damage.

This paper describes a weight reduction problem of aluminum disc wheels under cornering fatigue constraints. In this paper, finite element models of aluminum wheels are constructed to simulate the dynamic cornering fatigue test. Based on Goodman and Gerber's fatigue criteria, the analysis results are used to predict whether a wheel can pass the cornering fatigue test. The possible failure positions are also identified. This procedure proves to be a practical and reliable tool for predicting whether the wheels can pass the dynamic cornering fatigue test. Matching with physical test results of a local wheel manufacturer in a sample of 26 wheels, this procedure have a 96% success rate in correctly predicting whether the wheels will pass the dynamic cornering fatigue test. Therefore design engineers can decide whether and how to redesign the wheel based on the finite element analysis result.

The design issue for the wheels that are predicted to pass the fatigue test is, ''is there any room to further reduce the weight of the wheel?'' A common approach is to put weight reduction holes on the back of the spokes of the wheels. The position, shape, and size of the weight reduction holes become the design variables that the engineers have to decide.

This is a typical structural optimization problem: to minimize the weight of the wheel, subject to the fatigue constraint. This is also a typical engineering optimization problem that cannot be solved by simply applying existing numerical optimization algorithms. In this optimization problem, the fatigue constraint is the so-called ''implicit constraint''. It cannot be expressed as an analytical function in terms of the design variables. Evaluating the fatigue constraint is quite expensive. Moreover, the constraint is in the form of ''pass-or-fail'', ''feasible-or-infeasible''. The numeric function value of the constraint does not exist, let alone its first derivative.

To handle this type of structural optimization problems, a ''sequential neural-network approximation method (the SNA method)'' is presented in this paper. One important category of numerical optimization algorithms is the sequential approximation methods. The basic idea of the sequential approximation methods is to use a simple sub-problem to approximate the hard, exact problem. By ''simple'' sub-problems, we mean the type of problems that can be readily solved by existing numerical algorithms. For example, linear programming sub-problems are commonly used in the sequential approximation methods. The solution point of the simple sub-problem is then used to form a better approximation to the hard, exact problem for the next iteration. In an iterative manner, it is expected that the solution point of the simple approximate problem will approach the optimum point of the hard exact problem.

The SNA method is also a sequential approximation method. In this method, first a back-propagation neural network is trained to simulate the feasible domain formed by the implicit constraints using just a few training data. A training data consists of two pieces of information: a design point (i.e. a set of values for the design variables) and whether this design point is feasible-or-infeasible. A search algorithm then searches for the ''optimal point'' in the feasible domain simulated by the neural network. This new design point is checked against the true implicit constraints to see whether it is feasible, and the new training data is then added to the training set. The neural network is trained again with this added training data, in the hope that the network will better approximate the boundary of the feasible domain of the exact optimization problem. Then we search for the ''optimal point'' in this approximated feasible domain again. This process continues in an iterative manner until we get the same design point repeatedly and no new training point is generated.

This paper starts by describing the modeling of the cornering fatigue test of disc wheels and the prediction

of fatigue failure using the finite element model. Then the SNA method is presented to solve this structural optimization problem.

2. Modeling the dynamic cornering fatigue test

The dynamic cornering fatigue test simulates the loading condition of the wheels in normal driving. Fig. 1 shows a typical set-up of the 90° loading method of cornering fatigue, according to SAE J32 [4]. In figure, the downside outboard flange of rim of the wheel is clamped securely to the test device, and a rigid load arm shaft is attached to the mounting surface of the wheel. A test load applies on the arm shaft to provide a constant cyclical rotation bending moment. After being subjected to the required number of test cycles, there shall be no evidence of failure of the wheel, as indicated by propagation of a crack existing prior to test, new visible cracks penetrating through a section, or the inability of the wheel sustain load.

The finite element method is used to simulate the dynamic cornering fatigue test. Fig. 2 shows the finite element model of an aluminum wheel. All degrees of freedom of the nodes on the downside outboard flange of the rim are fixed. The dynamic cyclical load is represented by 24 discrete loads, 15° apart. For each node, the maximum and minimum von mises stresses during the load cycle are extracted to obtain the mean stress σ_m and the stress amplitude σ_a of the node, as defined below:

$$\sigma_m = \tfrac{1}{2}(\sigma_{max} + \sigma_{min}) \quad (1)$$

$$\sigma_a = \tfrac{1}{2}(\sigma_{max} - \sigma_{min}) \quad (2)$$

The (σ_m, σ_a) of all 14,000 nodes of the wheel model are plotted in Fig. 3.

Several different criteria are commonly used in predicting fatigue failure. In this study, Goodman and Gerber's criteria are used.

Goodman's criterion:

$$\frac{\sigma_a}{S_e/n} + \frac{\sigma_m}{S_u/n} = 1 \quad (3)$$

Gerber's criterion:

$$\frac{\sigma_a}{S_e/n} + \left(\frac{\sigma_m}{S_u/n}\right)^2 = 1 \quad (4)$$

where S_e is the endurance limit, S_u the ultimate strength of the material, and n is the design factor [5].

Goodman and Gerber's lines of the material of the wheel are plotted in Fig. 4 (design factor $n = 1$). The top 1% of the nodes that are closest to Goodman's line are also left in figure. These nodes have the greatest possibility to fail in the fatigue test. The positions of these nodes on the wheel are also recorded. The average of σ_m and σ_a of these top 1% nodes are calculated, as shown by the solid circle $(\sigma_m, \sigma_a)_{1\%}$ in Fig. 4. This point $(\sigma_m, \sigma_a)_{1\%}$ is used to represent the

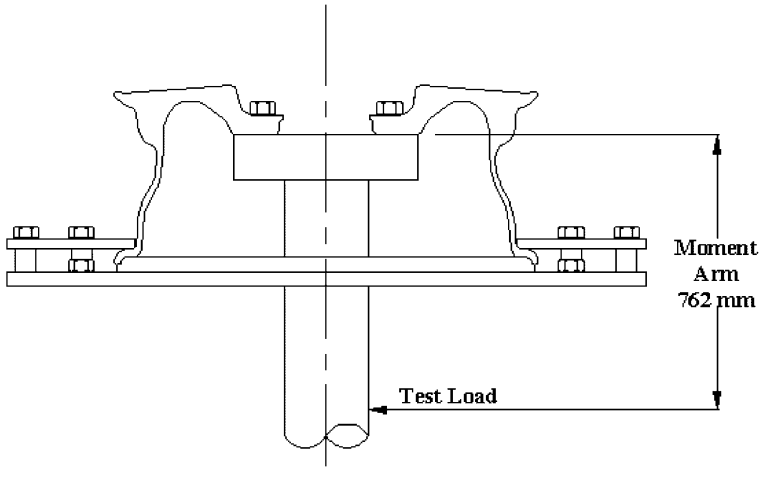


Fig. 1. Typical set-up of 90° loading method of cornering fatigue [4].

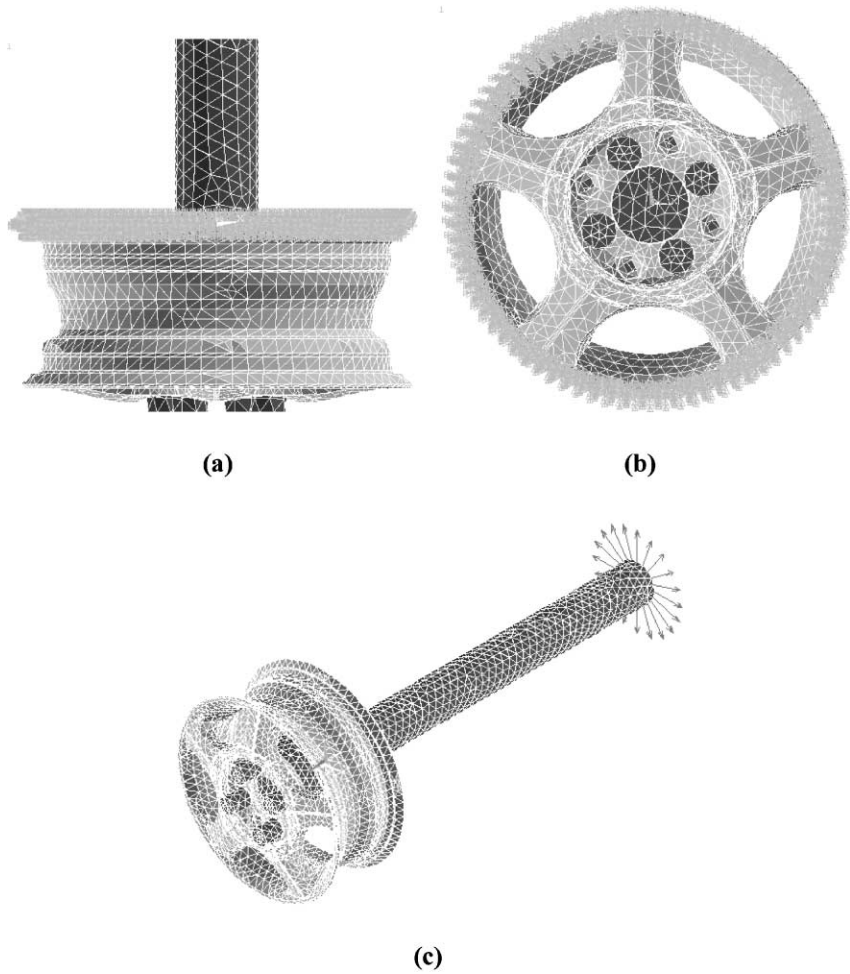


Fig. 2. The finite element model of a wheel: (a) top view; (b) front view; (c) cyclical load.

wheel when checking with Goodman and Gerber's lines to predict whether the wheel will pass the cornering fatigue test.

3. Predicting the fatigue failure of wheels

Finite element models of 28 aluminum wheels from a local manufacturer are constructed to simulate the cornering fatigue tests. The $(\sigma_m, \sigma_a)_{1\%}$ for each wheel is calculated. Fig. 5 shows the $(\sigma_m, \sigma_a)_{1\%}$ of all 28 wheels. These wheels are already physically tested. In Fig. 5, a circle represents the wheels that actually passed the cornering fatigue test, while a cross represents the wheels that did not pass the cornering fatigue test.

These historical test data provide a reference for choosing the design factor n in Eqs. (3) and (4) for this specific case. It is found that for $n = 2.6$, Goodman and Gerber's lines fit the 28 test data well, as shown in Fig. 5. The region inside Goodman's line is the ''safe zone''. Five out of 28 wheels fall in region, and four of them did pass the cornering fatigue test. The region between Goodman and Gerber's lines is the ''dangerous zone''. Fourteen wheels fall in this region, eight of them did not pass the cornering fatigue test. Finally the region outside of Gerber's line is the ''failure zone''. Nine wheels fall in this region, and seven of them did not pass the cornering fatigue test. Note that the two wheels in this zone that passed the test are very close to Gerber's line.

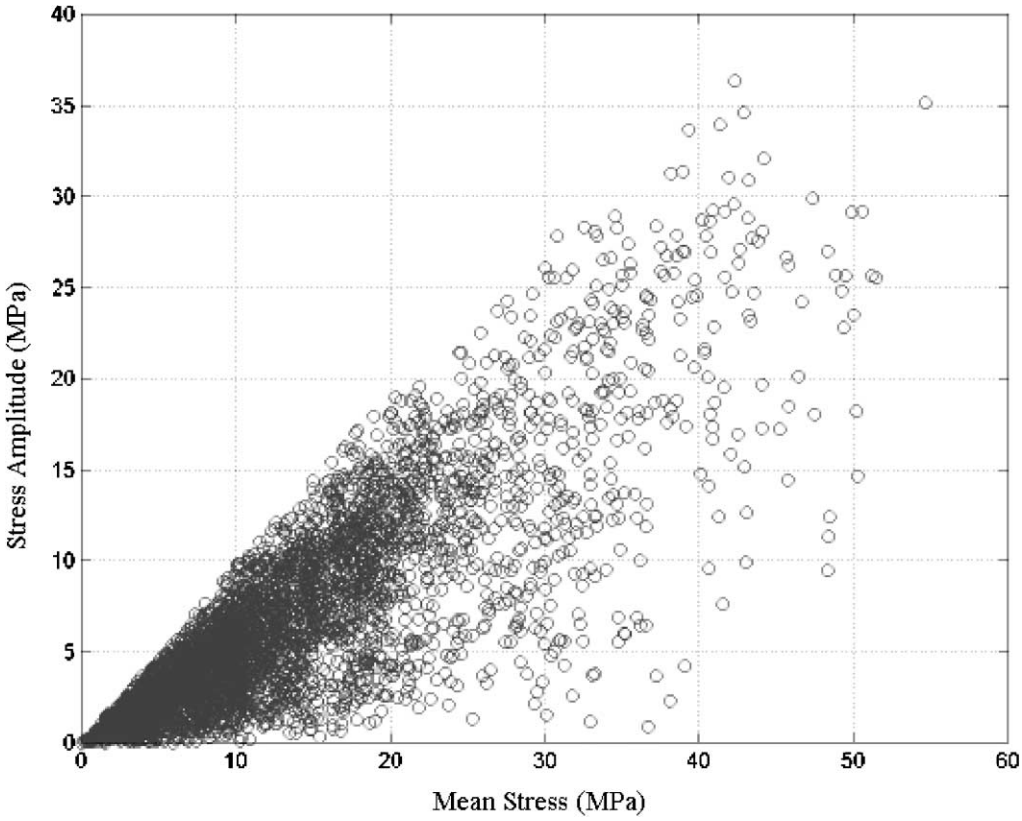


Fig. 3. Mean stress and stress amplitude of all nodes.

This procedure proves to be a practical and reliable tool for predicting whether the wheels can pass the dynamic cornering fatigue test. Following this study, another batch of 26 new aluminum wheels are predicted using the same procedure and the same design factor. The prediction results are then matched with the physical test results. As shown in Fig. 6, all 10 wheels that are in the safe zone did pass the cornering fatigue test. Three wheels fall in the dangerous zone and one of them did not pass the cornering fatigue test. Thirteen wheels fall in the failure zone, and 12 of them did not pass the cornering fatigue test. In this sample of 26 wheels, this procedure has a 96% success rate in correctly predicting whether the wheels will pass the dynamic cornering fatigue test.

For the wheels that are predicted to pass the cornering fatigue test, design engineers may consider to put weight reduction holes on the back of the spokes to further reduce the weight of the wheel. It becomes a typical structural optimization problem: to minimize the weight of the wheel, subject to geometry and fatigue constraints. This problem can be formulated as follows:

$$\text{minimum weight}\,(\boldsymbol{x}), \quad \text{such that}\, \boldsymbol{g}_{\mathrm{e}}(\boldsymbol{x}) \leq 0, \quad \text{fatigue}\,(\boldsymbol{x}) = 0 \tag{5}$$

where $\boldsymbol{x}$ is the vector of design variables, such as the size, shape, depth, and position of the weight reduction holes; and $\boldsymbol{g}_{\mathrm{e}}(\boldsymbol{x})$ are the explicit constraints, that is, the constraints that can be written explicitly in terms of design variables, such as the geometry constraints on the size of the hole.

Constraint fatigue $(\boldsymbol{x})$ is a binary constraint. If the current design $\boldsymbol{x}$ (with the weight reduction holes) falls in the safe zone inside Goodman's line, then fatigue $(\boldsymbol{x}) = 0$, otherwise fatigue $(\boldsymbol{x}) = 1$. The fatigue constraint is the so-called ''implicit constraint''. It cannot be expressed as an analytical function in terms

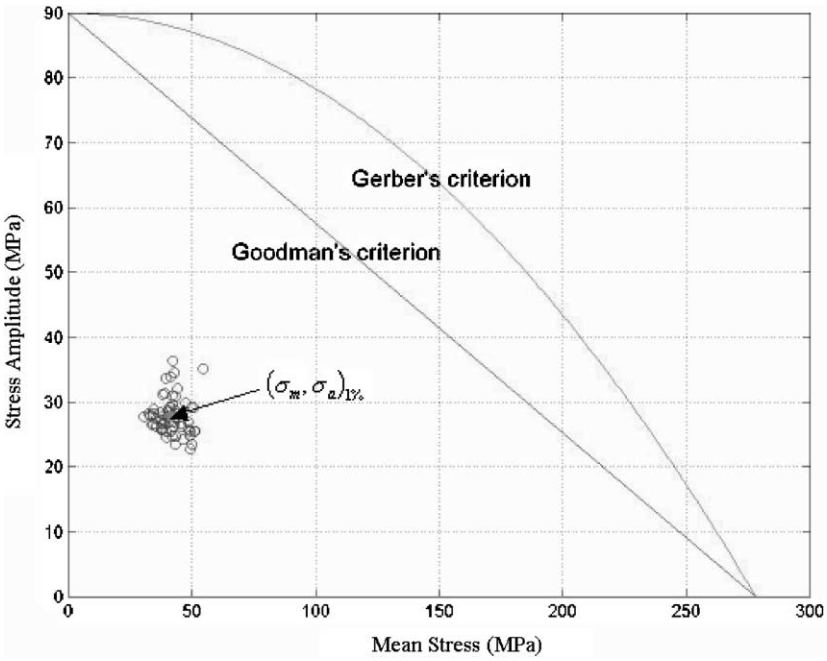


Fig. 4. Goodman and Gerber's lines for design factor $n = 1$.

of design variables. Evaluating the constraint is quite expensive. Moreover, the constraint is in the form of ''pass-or-fail'', ''feasible-or-infeasible''. The numeric function value of the constraint does not exist, let alone its first derivative.

This type of implicit constraints is quite common in engineering optimization problems involving complicated computer simulation or experiments. This type of optimization problems cannot be solved by simply applying certain numerical optimization

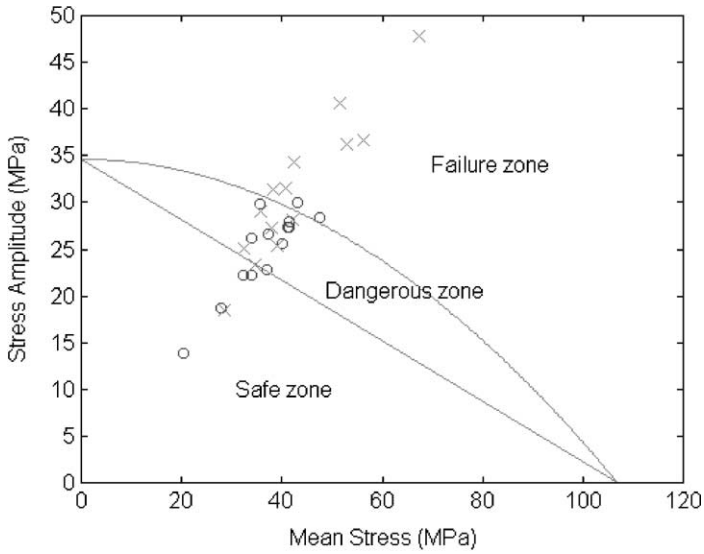


Fig. 5. Goodman and Gerber's lines for design factor $n = 2.6$.

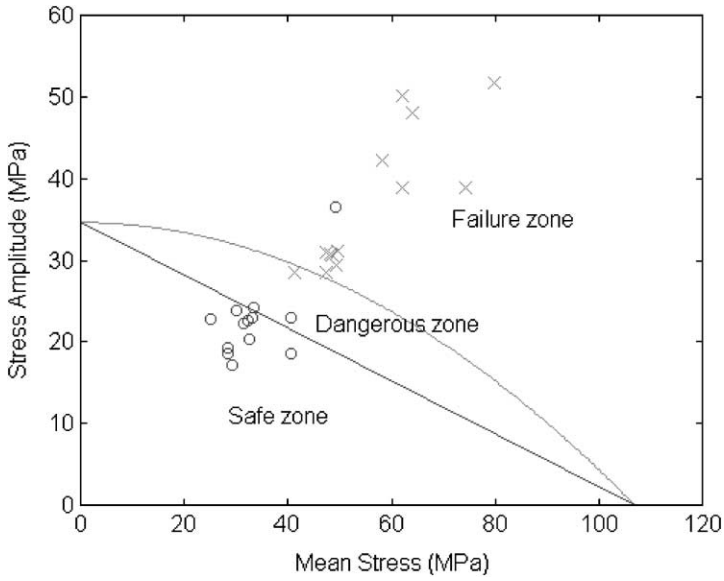


Fig. 6. Prediction results of 26 new aluminum wheels.

algorithms. In the next section, the SNA method is presented to handle this type of structural optimization problem.

4. Using neural networks for optimization

Hopfield and Tank [6] extended neural network applications to optimization problems. In Hopfield's model, the optimization problem is converted into a problem of minimizing an energy function formulated by the design restraints. Lee and Chen [7] further explored the idea of applying neural network technology in design optimization. In their approach, neural networks were used to simulate the constraint functions. Oda and Mizukami [8] developed an application of a ''mutual combination-type'' neural network to the optimal design problems of truss structure. Hopfield's model and the quadratic energy function are used to solve the design problems, and the method is able to obtain feasible solutions for several optimal design problems.

In the SNA method presented in this paper, the neural networks are used to simulate the feasible domain formed by the implicit constraints. In this design case, Eq. (5) can be thought of as a hard, exact problem because of the existence of the fatigue constraint. It is approximated by a ''simple'' approximate problem below:

$$\text{minimum} f(\boldsymbol{x}), \quad \text{such that } \boldsymbol{g}_{\text{e}}(\boldsymbol{x}) \leq 0, \quad \text{NN}(\boldsymbol{x}) = 0 \tag{6}$$

where the binary constraint $\text{NN}(\boldsymbol{x}) = 0$ approximates the feasible domain of the fatigue constraint. If $\text{NN}(\boldsymbol{x}) = 0$, the design point $\boldsymbol{x}$ is feasible; if $\text{NN}(\boldsymbol{x}) = 1$, the design point $\boldsymbol{x}$ is infeasible. For convenience, the optimization model in Eq. (5) will be denoted M_{real} throughout the paper, and the approximate model in Eq. (6) will be denoted M_{NN}. M_{NN} is a ''simple'' model because once the training of the neural networks is completed, the computational cost of evaluating $\text{NN}(\boldsymbol{x})$ is much less than that of evaluating the implicit constraint fatigue $(\boldsymbol{x})$.

To illustrate the SNA method, the wheel design case with circular weight reduction holes is first presented, as shown in Fig. 7. There are two design variables h (depth of the hole) and r (radius of the hole) in this case, and each variable has 11 possible discrete values:

$$h = \{0, 3, 6, 8, 10, 11, 11.5, 11.75, 12, 12.25, 12.5\}$$

$$r = \{0, 5, 10, 15, 18, 20, 21, 22, 22.5, 23, 23.5\}$$

Note that these discrete values are not equally spaced. Higher resolution is used when the radius and the depth of the hole are large. The position of the center

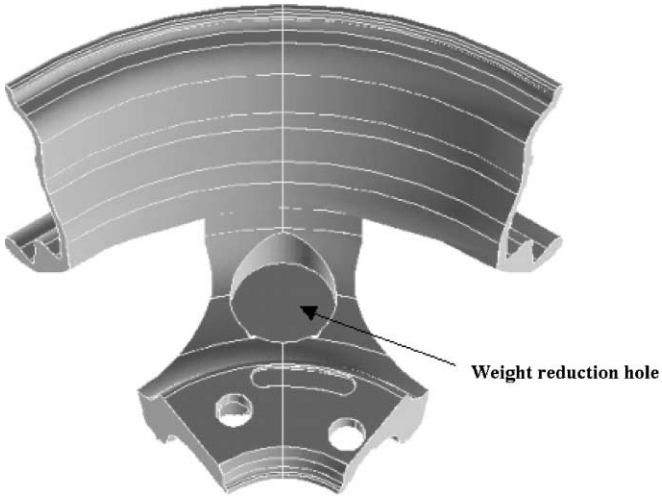


Fig. 7. Weight reduction holes of the wheel with only two variables.

of the hole is fixed. The explicit geometry constraint $\boldsymbol{g}_e(\boldsymbol{x}) \leq 0$ is already reflected in the range of the discrete values. Note that the objective function $f(\boldsymbol{x})$ is also an explicit function. In this case

$$f(\boldsymbol{x}) = W - \pi r^2 h \tag{7}$$

where W is the weight of the wheel without the holes.

Fig. 8 is a flow chart of the SNA method. A back-propagation neural network is trained to simulate the feasible domain of the implicit constraints using just a few training data (the set of initial training data $\boldsymbol{X}_0$). The initial optimization model M_{NN} is formed. After the initial training is completed, a search algorithm is started to search for the solution point $\boldsymbol{x}^j$ of M_{NN}. This

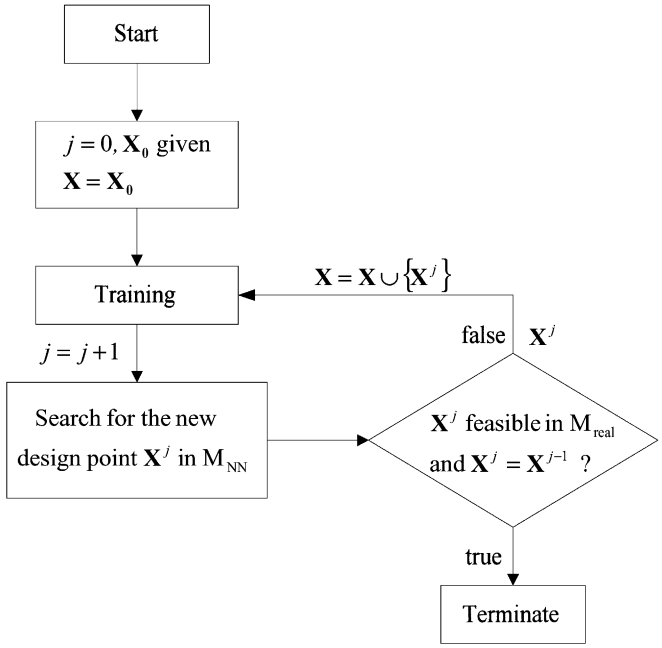


Fig. 8. Flow chart of the SNA method.

solution point is then evaluated to see whether it is feasible in M_{real}, and whether M_{NN} generates the same solution point as in the previous iteration. If the feasible solution point $\boldsymbol{x}^j$ is the same as $\boldsymbol{x}^{j-1}$, no new training data is generated. The feasible domain simulated by the neural networks will not change in the next iteration, so this procedure has to stop. If a new solution point $\boldsymbol{x}^j$ is generated, it is added to the set of training points $\boldsymbol{X}$, and the neural network is trained again. With the increasing number of training points, it is expected that the feasible domain simulated by the neural network in M_{NN} will approach to that of the exact model M_{real}, and the solution point of M_{NN} will approach the true optimum point.

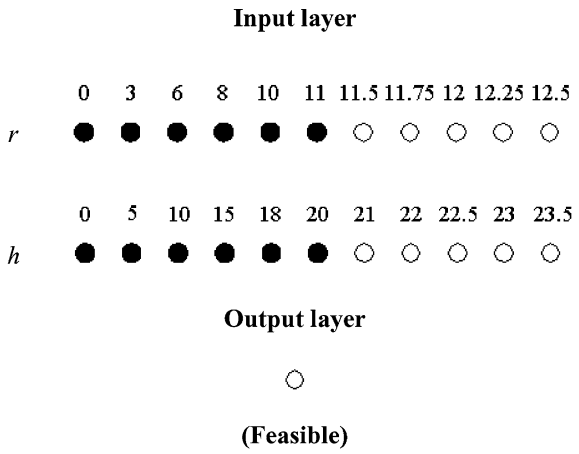


Fig. 9. Representation of a feasible training point $(r, h) = (11, 20)$.

5. Representation of a design point

There are three layers in the neural network, the input layer, the hidden layer, and the output layer. The size of the input layer depends on the number of variables and the number of discrete values of each variable. In this example, there are two design variables and each variable has 11 possible discrete values. Thus a total of 2×11 neurons are used in the input layer. Each neuron in the input layer has value of 0 or 1 to represent a discrete value the corresponding variable takes. There is only a single neuron in the output layer to represent whether this design point is feasible (the output neuron has value 0) or infeasible (the output neuron has value 1).

In the wheel design example, the center point of the discrete search domain $(h, r) = (11, 20)$ is used as one of the initial training points. This training point is represented as shown in Fig. 9, where a blank circle represents a "0" in the node, a solid circle represents a "1" in the node. For illustrative purpose, the 22 input nodes are arranged into two rows in Fig. 9 to represent two design variables. The first six nodes of the first row have value 1 to represent that the first variable r has the discrete value 11. The first six nodes of the second row also have value 1 to represent that the second variable h also has the discrete value 20. Checking with M_{real}, we can determine that $(h, r) = (11, 20)$ is a feasible design point. So the single node in the output layer has the value of 0 to represent that this point is feasible.

The number of neurons in the hidden layer depends on the nature of the problem, and on the number of neurons in the input layer. We decided to put 16 neurons in the hidden layer in this case after a few trial runs. The transfer functions used in the hidden and output layer of the network are both log-sigmoid functions. The neuron in the output layer has a value range (0, 1). After a training is completed, a threshold value is applied to the output layer when simulating the boundary of the feasible domain. In other words, given a discrete design point in the search domain, the network always output 0 (if output neuron's value is less than the given threshold) or 1 (otherwise) to indicate whether this discrete design point is feasible-or-infeasible.

Note that the artificial neural network can be a continuous input model, and could be used with discrete values if needed. However, the representation of a training point in Fig. 9 is designed to have 2×11 input element instead of just two (r and h). In this representation, all training data are in a crisp 0–1 pattern, which makes the training process relatively fast. A conjugate gradient algorithm (Powell–Beale restarts) is used for the training. In this example, the error goal of 1e-5 is usually met within 100 epochs, even for later cases with many training points.

6. Searching in the feasible domain

Using three initial training points $\{(0, 0), \text{feasible}\}$, $\{(11, 20), \text{feasible}\}$, and $\{(12.5, 23.5), \text{infeasible}\}$, Fig. 10 is a rough map of the feasible domain of

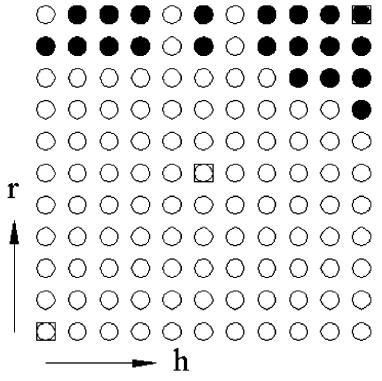


Fig. 10. The feasible domain (represented by blank circles) of M_{NN} by three initial training points.

M_{NN} after the training is completed. As shown in figure, these three initial training points are the lower extreme, the center point, and the upper extreme of the design domain. The mathematical optimization model M_{real} is now transformed into M_{NN}, in which the feasible domain of the optimization model is simulated by NN($\boldsymbol{x}$).

A discrete search algorithm is designed to search for the solution point $\boldsymbol{x}^j$ in M_{NN}. This algorithm has to start from a feasible design point. If the new design point generated from the previous iteration is a feasible design point in M_{real}, it is used as the starting point in the current search. If the new design point is infeasible in M_{real}, then use the same starting point of the previous iteration in the current search.

From the feasible starting point, the search algorithm moves one variable at a time to the neighboring discrete value. At the beginning of the search all design variables form a ''usable set'', which is denoted U. The gradient of the objective function Eq. (7) with respect to the design variables is calculated at the starting point. Then one variable is picked to move to the neighboring discrete value. From all variables $x_i \in U$, the variable x_k that has the largest $|\partial f/\partial x_k|$ is picked. Since we are minimizing the objective function, if $\partial f/\partial x_k < 0$, then x_k is moved to the higher neighboring discrete value; on the contrary, if $\partial f/\partial x_k > 0$, then x_k is moved to the lower neighboring discrete value.

This new design point is checked against NN($\boldsymbol{x}$) to see if it is feasible in M_{NN}. If the new design point is feasible, this move is accepted and the search algorithm is started again from this new design point. If the new design point is infeasible in M_{NN}, this move is discarded, the variable x_k is not usable, so we update the usable set, $U = U - x_k$. Another variable with the largest gradient value is picked from the usable set, and we proceed with the same procedure. This search process terminates when $U = \phi$, since all moves from the current design point for lower objective function values are infeasible in M_{NN}. Therefore the current design point is the optimum design point (at least locally) in M_{NN}.

Back to our wheel design example, the search algorithm described above is then started to search for the optimum point in the feasible domain in Fig. 10. The feasible design point $(h, r) = (11, 20)$ is used as the starting point, and the search algorithm terminates at a new design point $(h, r) = (12, 21)$. This new design point is then evaluated using the true constraints of M_{real} in Eq. (5), and is determined infeasible. Thus a new training point $\{(12, 21), \text{infeasible}\}$ is added to the set of training points, and the neural network is trained again. After the training is completed, NN($\boldsymbol{x}$) is updated, the search algorithm is started again to search for the optimum point in the feasible domain simulated by M_{NN} with four training points.

The iteration continues, with one more training point added into the set of training points in each iteration. The whole process terminates when we get the same design point repeatedly and no new training point is generated. In our wheel design example, the SNA method terminates after six iterations. The final design point obtained by the SNA method is $(h, r) = (11.5, 20)$. The weight of the wheel decreases from 4683 to 4446 g, a 5.1% reduction. Fig. 11 shows the iteration history of this example; the points represented by circles are the feasible design points. Note that only eight training points are used, including three in the initial training, and five during the iterations. That is, only eight evaluations (out of the $11 \times 11 = 221$ possible combinatorial combinations) of the implicit constraints are needed in this case. No gradient calculation is required.

To check the quality of the design point obtained by the SNA method, Fig. 12 compares the feasible domain simulated by NN($\boldsymbol{x}$) at the end of the process and the true feasible domain. The design point obtained by the SNA method is indeed the global

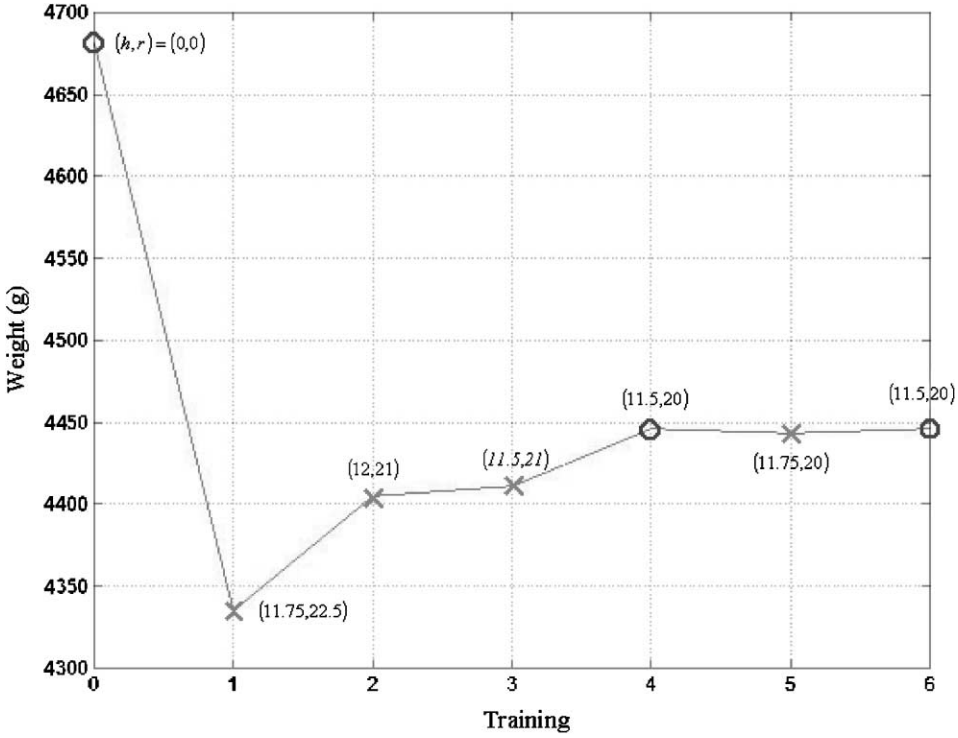


Fig. 11. Iteration history of the wheel design example with two variables.

minimum of M_{real}. Note that in the SNA method, as well as in most numerical optimization algorithm, global optimality is not guaranteed. At the end of the process, the feasible domain of M_{NN} in the neighborhood of the final design point is close to that of the M_{real}, because most training points (nodes enclosed by square brackets in figures) fall in this region. The final design point is a local optimum in this neighborhood.

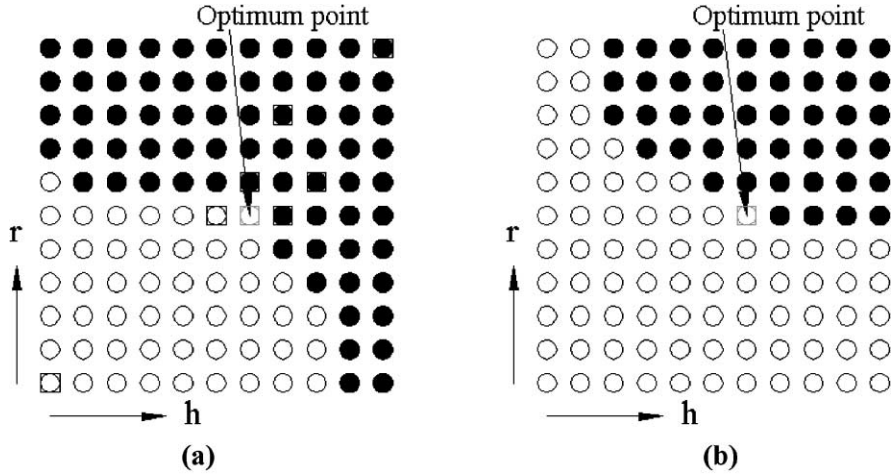


Fig. 12. The feasible domains (represented by blank circles) of (a) M_{NN} and (b) M_{real}.

7. Weight reduction holes described by five design variables

To demonstrate the capability of the SNA method in handling multiple variables, the same weight reduction problem is solved again in this section. As shown in Fig. 13, this time the weight reduction hole is described by five design variables, and the shape of the hole is much more flexible. Each variable has 11 possible discrete values:

$a \in \{4, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19\}$

$b \in \{5, 9, 13, 17, 21, 25, 27, 29, 31, 33, 35\}$

$c \in \{10, 14, 18, 22, 26, 30, 32, 34, 36, 38, 40\}$

$d \in \{10, 18, 26, 34, 42, 50, 54, 58, 62, 66, 70\}$

$w \in \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$

Again, the lower extreme $\{(4, 5, 10, 10, 2), \text{feasible}\}$, the center point $\{(14, 25, 30, 50, 7), \text{infeasible}\}$, and the upper extreme $\{(4, 5, 10, 10, 2), \text{feasible}\}$ of the design domain are used as the initial training points. The SNA method terminates after seven iterations.

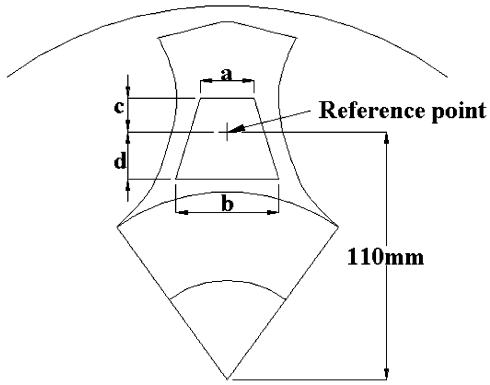


Fig. 13. Weight reduction holes described by five design variables.

The final design is $(a, b, c, d, w) = (19, 35, 40, 10, 12)$. The weight of the wheel decreases from 4683 to 4376 g, a 6.5% reduction, which is better than the 5.1% of the previous case. Fig. 14 shows the iteration history. Note that only nine evaluations (out of the $11^5 = 161{,}051$ possible combinatorial combinations) of the implicit constraints are needed to obtain this design.

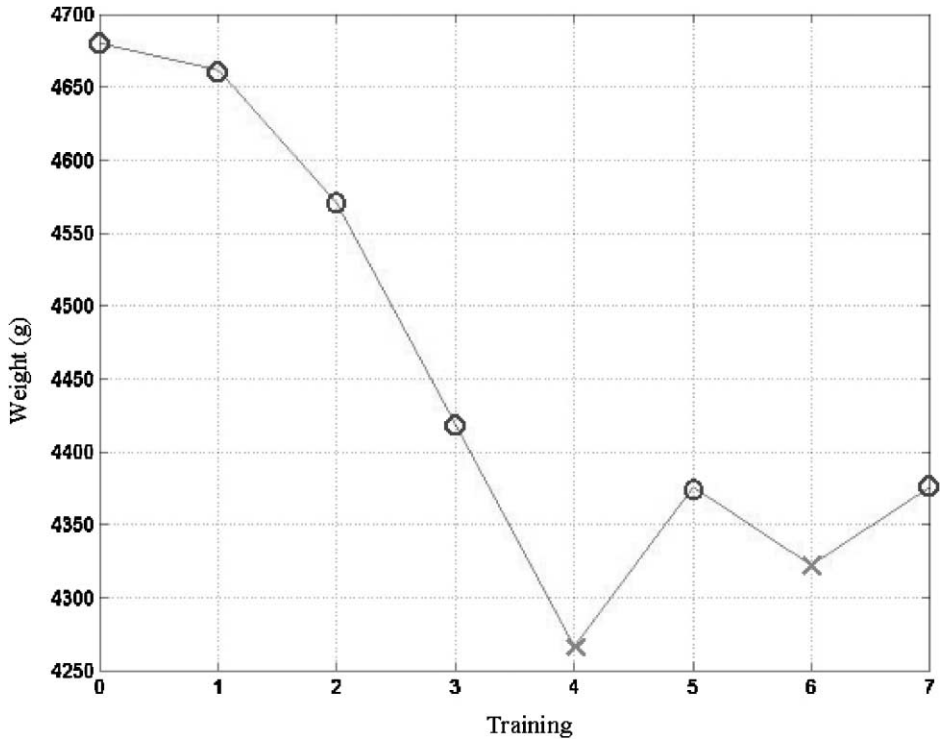


Fig. 14. Iteration history of the wheel design example with five variables.

8. Discussions and conclusions

This paper describes a weight reduction problem of aluminum disc wheels under cornering fatigue constraints. This is a special structural optimization problem because of the existence of the implicit fatigue constraint. Numerical optimization algorithms are usually derivative-based methods that require at least the first derivatives of the implicit constraints with respect to the design variables. Typically, only the local numerical information from the solution point of the previous iteration is used in the current iteration. Knowledge about the whole search domain is not accumulated for formulating better approximation, or for better understanding of the optimization model. Precious function evaluations and sensitivity calculations of the implicit constraints are often not most efficiently utilized.

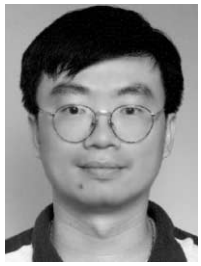
The SNA method presented in this paper is attractive in two ways. First, in each iteration, only one evaluation of the implicit constraints is needed to see whether the current design point is feasible. No precise function value or sensitivity calculation is required. Therefore it is especially suitable for engineering optimization problems with expensive implicit constraints. Secondly, in the SNA method, precious function evaluations of the implicit constraints are accumulated and utilized repeatedly to form better approximation to the feasible domain. In the wheel design example, the SNA method demonstrated the potential of being a more practical and efficient approach than the derivative-based numerical approximation algorithms.

Acknowledgements

This research is partially supported by the National Science Council, Taiwan, Project No. NSC 89-2212-E-155-004, and by Ensure Co. Ltd. This support is gratefully acknowledged.

References

[1] H.M. Karandikar, W. Fuchs, Fatigue life prediction for wheels by simulation of the roating bending test, SAE Transactions 99 (1990) 180–190.
[2] F. Kaumle, R. Schnell, Entwicklungszeit von Leichtmetallradern verkurzen, Materialpruefung/Materials Testing 40 (1998) 394–398.
[3] G. Marron, P. Teracher, Application of high-strength, hot-rolled steels in auto wheels, Minerals Metals & Materials Society of JOM 48 (1996) 16–20.
[4] SAE Handbook, Society of Automotive Engineers Inc., 1992, pp. 4:31.01–4:31.16.
[5] J.A. Bannantine, J.J. Comer, J.L. Handrock, Fundamentals of Metal Fatigue Analysis, Prentice-Hall, Englewood Cliffs, NJ, 1990.
[6] J.J. Hopfield, D.W. Tank, Neural computation of decisions in optimization problems, Biological Cybernetics 52 (1985) 141–152.
[7] S.-J. Lee, H. Chen, Design optimization with back-propagation neural networks, Journal of Intelligent Manufacturing 2 (1991) 293–303.
[8] J. Oda, T. Mizukami, Technique for optimum design of truss structures by neural network, Nippon Kikai Gakkai Ronbunshu, A Hen/Transactions of the Japan Society of Mechanical Engineers. Part A 59 (1993) 273–278.



Yeh-Liang Hsu received his PhD degree from Mechanical Engineering Department, Stanford University, in 1992. He is now associate professor and chairman of Mechanical Engineering Department, Yuan Ze University, Taiwan. He specializes in mechanical design and design optimization.

Ming-Sho Hsu is a PhD student of Mechanical Engineering Department, Yuan Ze University. He also specializes in design optimization, especially in structure optimization.