Journal of Civil Engineering and Construction Technology Vol. 3(3), pp. 99-107, March 2012
Available online at http://www.academicjournals.org/JCECT
DOI: 10.5897/JCECT11.075
ISSN 2141-2634 ©2012 Academic Journals

Full Length Research Paper

Forecasting the lowest cost and steel ratio of reinforced concrete simple beams using the neural network

Jiin-Po Yeh* and Kun-Yu Chen

Department of Civil and Ecological Engineering, I-Shou University, Kaohsiung City 84001, Taiwan.

Accepted 10 January 2012

Given the compressive strength of concrete, yield strength of steel, span, dead and live loads, singly reinforced concrete simple beams are first optimally designed using genetic algorithms with constraints satisfying the specifications of the ACI code. The objective function is to minimize the total cost of tension steels, stirrups and concrete. A variety of beams are designed for the use of the neural network. To train and test the effectiveness of the neural network, these optimal results are randomly divided into three sets: the training set, validation set and test set. This paper uses a two-layer feed forward neural network: one hidden layer and one output layer. The transfer functions for the hidden layer and output layer are tan-sigmoid and linear functions, respectively. The inputs of the neural network are the compressive strength of concrete, yield strength of steel and span, width and effective depth of the beam, as well as vertical loads the beam is subjected to; the targets of the neural network are the steel ratio and cost of the beam. To evaluate the accuracy, the regression analysis of the target and network output is carried out. Numerical results show good performance of the neural network, which can be used as a model to predict the lowest cost and steel ratio of singly reinforced concrete simple beams.

Key words: Reinforced concrete beams, genetic algorithm, neural network, regression analysis.

INTRODUCTION

There exist many optimization and computational methods which are inspired by the biological evolution. The genetic algorithm (GA) is one of the most popular algorithms belonging to these evolutionary methods. It solves both constrained and unconstrained optimization problems based on natural selection, the process that drives biological evolution. The constraints can be in the form of linear or nonlinear equality or inequality with bounds on the optimization variables. This algorithm is inspired by biological evolution based on Charles Darwin's “survival of the fittest” theorem. It is less susceptible to getting stuck at local optima than gradient search methods. The concept of genetic algorithms was formally introduced in 1970s by Professor John Holland at the University of Michigan, who in 1975 published the ground-breaking book “Adaptation in Natural and Artificial System” (Holland, 1975). From then on, the continuing price/performance improvements of computational systems have made the genetic algorithm more attractive and popular. Genetic algorithms have a number of applications in a wide spectrum of problem areas, including structural designs, such as truss systems (Rajeev and Krishnamoorthy, 1992; Coello and Christiansen, 2000), the plane frame (Jenkins, 1992), a welded beam (Deb, 1991), etc.

The neural network was originated by McCulloch and Pitts (McCulloch and Pitts, 1943), who claimed that neurons with binary inputs and a step-threshold activation function were analogous to first order systems. Hebb (1949) revolutionized the perception of artificial neurons. Rosenblatt (1958), using the McCulloch-Pitts neuron and the findings of Hebb, developed the first perception model of the neuron which is still widely accepted today. Hopfield (1982) and Hopfield et al. (1983) demonstrated from work on the neuronal structure of the common

*Corresponding author. E-mail: jpyeh@isu.edu.tw. Tel: +888-7-6577711, Ext: 3313.

garden slug that ANNs (artificial neural networks) can solve non-separable problems by placing a hidden layer between the input and output layers. Rumelhart and McClelland (1986) developed the most famous learning algorithm in ANN-back propagation, which uses a gradient descent technique to propagate error through a network to adjust the weights in an attempt to find the global error minimum, marking a milestone in the current artificial neural networks. Since then, a huge proliferation in the ANN methodologies has occurred.

BACKGROUNDS ON THE GENETIC ALGORITHM AND NEURAL NETWORK

More knowledge about genetic algorithms and neural networks is illustrated as follows.

Genetic algorithms

The genetic algorithm begins by creating a random initial population, and then creates a sequence of new populations. At each step, the algorithm uses the individuals in the current generation to create the next population. To create the new population, the algorithm performs the following steps: (1) Scores each member of the current population by computing its fitness value; (2) Scales the raw fitness scores to convert them into a more usable range of values; (3) Selects members, called parents, based on their fitness; (4) Some of the individuals in the current population that have lower fitness are chosen as elite. These elite individuals are passed to the next population; (5) Produces children from the parents. Children are produced either by making random changes to a single parent—mutation—or by combining the vector entries of a pair of parents—crossover; and (6) Replaces the current population with the children to form the next generation. The algorithm stops when one of the stopping criteria is met, such as the number of generation, the weighted average change in the fitness function value over some generations less than a specified tolerance, no improvement in the best fitness value for an interval of time in seconds, etc.

The Matlab Toolbox for Genetic Algorithm (The MathWorks, 2010) is employed in this paper, which uses the Augmented Lagrangian Genetic Algorithm (Conn et al., 1991; Conn et al., 1997) to solve nonlinear constraint problems with bounds. The optimization problem of the simple beam is to Minimize $f(\boldsymbol{x})$ (the fitness function) such that:

$$C_i(\boldsymbol{x})\leq 0,\ \ i=1,\ldots,m$$
$$Ci(\boldsymbol{x})=0,\ \ i=m+1,\ldots,mt \qquad (1)$$
$$LB\leq x\leq \mathrm{UB}$$

Where $C_i(\boldsymbol{x})$ represents the nonlinear inequality and equality constraints, m is the number of nonlinear inequality constraints, mt is the number of nonlinear constraints, $f(\boldsymbol{x})$ is the total cost of tension steels, stirrups and concrete, and LB and UB are the vectors of lower and upper bounds of design variables, respectively. A subproblem is formulated by combining the fitness function and nonlinear constraint functions using the Lagrangian and the penalty parameters. A sequence of such optimization problems are approximately minimized using the genetic algorithm such that the bounds are satisfied. A sub-problem formulation is defined as

$$\Phi(\boldsymbol{x},\boldsymbol{\lambda},s,\rho)=f(\boldsymbol{x})-\sum_{i=1}^{m}\lambda_i s_i \log(s_i-C_i(\boldsymbol{x}))+\sum_{i=m+1}^{mt}\lambda_i C_i(\boldsymbol{x})+\frac{\rho}{2}\sum_{i=m+1}^{mt}C_i(\boldsymbol{x})^2 \qquad (2)$$

where the components λ_i of the vector $\boldsymbol{\lambda}$ are nonnegative and known as Lagrange multiplier estimates. The elements s_i of the vector $\boldsymbol{s}$ are nonnegative shifts, and ρ is the positive penalty parameter. The algorithm begins by using initial values for the parameters. The genetic algorithm minimizes a sequence of the subproblem, which is an approximation of the original problem. If the subproblem is minimized to a required accuracy, then the genetic algorithm stops. Otherwise, the Lagrangian estimates are updated or the penalty parameter is increased by a penalty factor. This results in a new subproblem formulation and minimization problem. These steps are repeated until the stopping criteria are met.

Neural networks

The neural network used in this paper is a two-layer feed forward neural network with the back propagation training algorithm, as shown in Figure 1. The transfer function used in the single hidden layer with 2R neurons is the tan-sigmoid function:

$$a_i = f(n_i)=\frac{e^{n_i}-e^{-n_i}}{e^{n_i}+e^{-n_i}} \quad , \quad i=1,2,3,\ldots,2R \qquad (3)$$

where n_i=$w_{i,1}x_1$+ $w_{i,2}x_2$+…+$w_{i,R}x_R$+ b_i, x_1, x_2,…,x_R are the inputs, R is the number of input elements, $w_{i,1}$, $w_{i,2}$,…, $w_{i,R}$ are the weights connecting the input vector and the ith neuron, and b_i is the bias of the ith neuron. The output layer with two neurons uses the linear transfer function

$$\boldsymbol{A}_i = f(\boldsymbol{N_i})=\boldsymbol{N}_i \quad , \quad i=1,\ 2 \qquad (4)$$

where N_i=$W_{i,1}a_1$+ $W_{i,2}a_2$+……+$W_{i,2R}a_{2R}$+ B_i, $W_{i,1}$,$W_{i,2}$,…,$W_{i,2R}$ are the weights connecting the neurons of the hidden layer and the ith neuron of the output layer, and B_i is the bias of the ith output neuron.

There are many variations of the back propagation

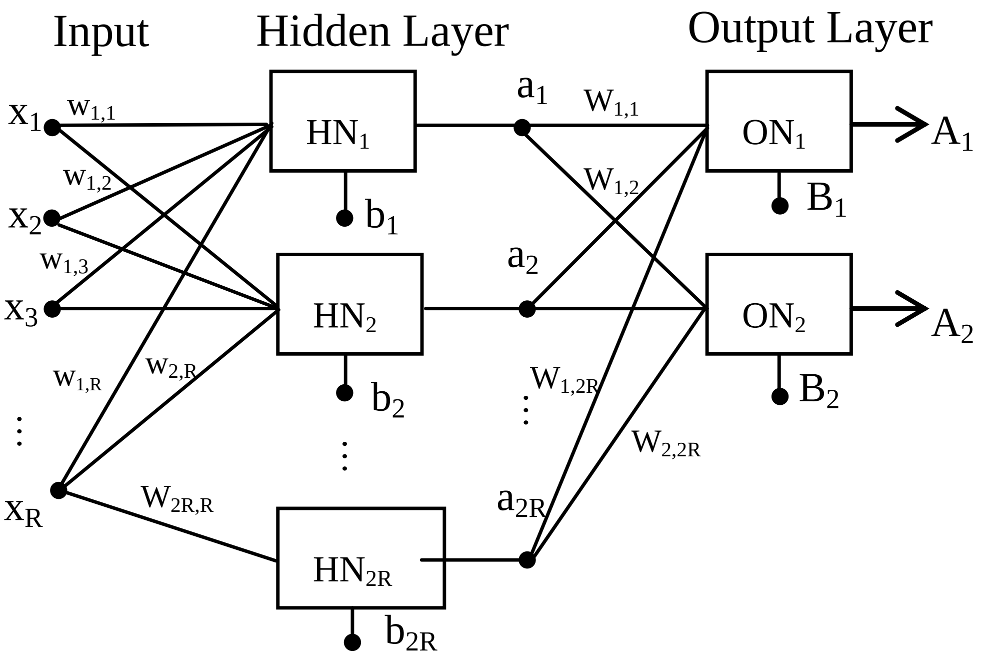


Figure 1. The feed forward neural network with two layers.

algorithm, which is aimed at minimizing the network performance function, i.e., the mean square error between the network outputs and the targets, that is,

$$\text{Minimize} \qquad MSE = \frac{1}{m}\sum_{j=1}^{m}(t_j - a_j)^2 \qquad (5)$$

where t_j and a_j are the *j*th target and network output, respectively. This paper selects the Levenberg-Marquardt algorithm (Hagan and Menhaj, 1994; Levenberg, 1994; Marquardt, 1963) as the training function to minimize the network performance function. This algorithm interpolates between the Newton's algorithm and the gradient descent method. If a tentative step increases the performance function, this algorithm will act like the gradient descent method, while it shifts toward Newton's method if the reduction of the performance function is successful. In this way, the performance function will always be reduced at each iteration of the algorithm.

To improve the network generalization, the error on the validation set is monitored during the training process. When the network begins to overfit the training data, the error on the validation set typically begins to rise. Once the validation error increases for a specified number of iterations (usually set to be 5), the training is stopped and the weights and biases at the minimum of validation error are returned. To evaluate the prediction accuracy, this paper performs the regression analysis between the network outputs and targets. The results are expressed by the parameters of the linear regression, including the correlation coefficient (Wackerly et al., 2008).

METHODOLOGY

Strength and deflection requirements of a simple beam

A number of simple beams with uniformly distributed dead load w_D and live load w_L are optimally designed by the genetic algorithm, based on which the neural network is then trained and tested. The constraints required to design the beam are formulated according to the ultimate-strength design of the ACI Building Code Requirements for Structural Concrete and Commentary (2008), considering the moment, shear force and deflection. Due to the ratio of the clear span to the depth greater than 4, the case of deep beams will not be considered in this paper. The equality and inequality constraints used in this paper when the genetic algorithm is executed are discussed as follows.

The strength requirement for flexure

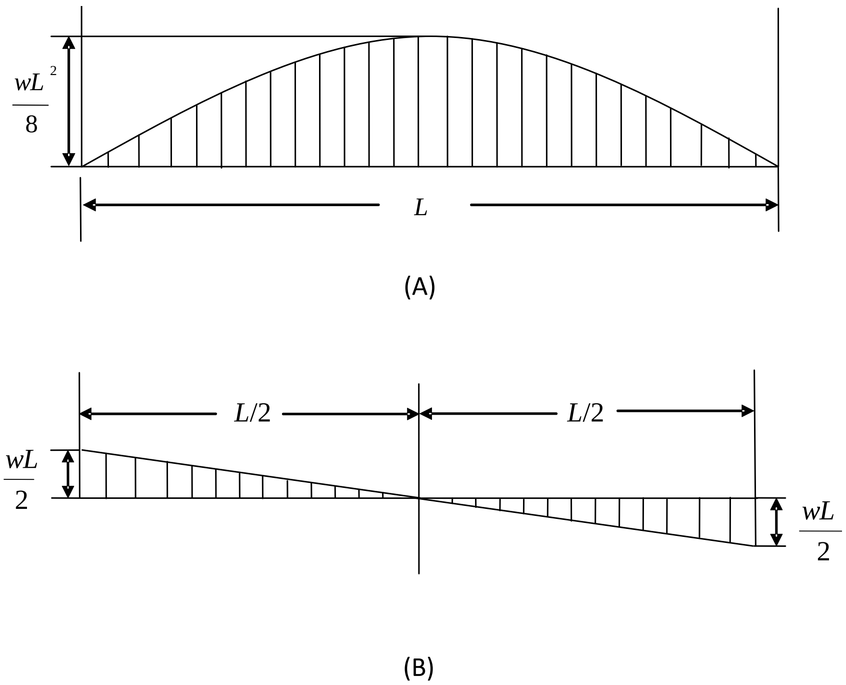
The moment diagram is shown in Figure 2(a). In order to yield optimal results, the strength requirement for flexure takes the equality form of

$$M_u = \phi_m M_n \qquad (6)$$

where $M_u = wL^2/8$ is the factored bending moment and $w = 1.2\, w_D + 1.6\, w_L$ is the factored uniformly distributed load applied to the simple beam, $\phi_m = 0.9$ is the strength reduction factor for flexure and

$$\mathrm{M}_n = \rho b d^2 f_y \left(1 - \frac{\rho f_y}{2(0.85) f_c'}\right) \qquad (7)$$

is the nominal resisting moment, where f_y is the yield strength of the



(A)

(B)

Figure 2(a). The moment diagram and **(b)** shear diagram of a simple beam subjected to the uniformly distributed load *w*.

tension reinforcement, f_c' is the compressive strength of concrete, *b* and *d* are the width and effective depth of the beam, respectively, and ρ is the tension reinforcement ratio. To have reasonable assurance of ductile mode of failure, ACI code limits the amount of tension steel to not more than 75% of that required for a balanced section, that is,

$$\rho \le 0.75\rho_b \tag{8}$$

where ρ_b is the balanced reinforcement ratio. The code also stipulates the minimum steel requirement as

$$\rho \ge \frac{3\sqrt{f_c'}}{f_y} \ge \frac{200}{f_y} \tag{9}$$

Once ρ is decided, the total amount of tension steel can be found, which is ρbdL.

The strength of shear reinforcement

The shear diagram is shown in Figure 2(b). Assuming that vertical stirrups are used, the strength of shear reinforcement

$$V_s = \frac{A_v f_y d}{s} \tag{10}$$

where *s* is the shear reinforcement spacing. If the nominal resistance shear $V_c = 2\sqrt{f_c'}bd$ is less than the nominal vertical shear force $V_u / \phi_s = V_n$, the shear reinforcement has to carry the difference in the two values, but the strength of shear reinforcement cannot be more than $8\sqrt{f_c'}bd$; hence

$$V_s = V_n - V_c = \frac{V_u}{\phi_s} - V_c \le 8\sqrt{f_c'}bd \tag{11}$$

Where V_u is the factored shear force and $\phi_s = 0.85$ is the strength reduction factor for shear. Because # 3 vertical closed stirrups are used, two stirrup bar areas A_v= 0.22 in^2 in Equation (10).

Shear reinforcement spacing

According to the ACI code, the critical section for determining the closest stirrup spacing may be taken at a distance *d* from the face of support. It also stipulates that the maximum stirrup spacing is *d*/2 but no to exceed 24 inch or $\frac{A_v f_y}{50b}$ for $V_s = \frac{V_u}{\phi_s} - V_c \le 4\sqrt{f_c'}bd$, i.e., $V_u \le 3\phi_s V_c$, and *d*/4 but not to exceed 12 inch or $\frac{A_v f_y}{50b}$ for $V_s = \frac{V_u}{\phi_s} - V_c \ge 4\sqrt{f_c'}bd$, that is, $V_u \ge 3\phi_s V_c$. Since the spacing of stirrups cannot be varied

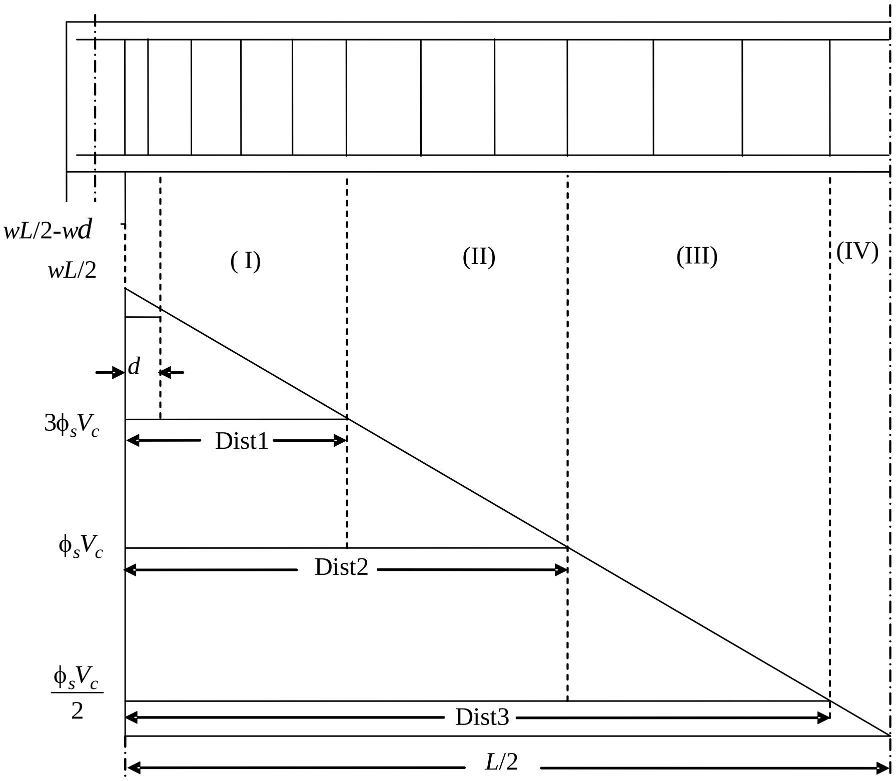


Figure 3. Four regions (I), (II), (III) and (IV) to arrange the stirrups.

continuously, they must change by jumps. Therefore, half the span of the beam is divided into four regions, as shown in Figure 3, where 3 $\phi_s V_c$ is assumed to be less than or equal to (*wL*/2-*wd*). If 3 $\phi_s V_c$ is greater than (*wL*/2-*wd*), then let 3 $\phi_s V_c$ = (*wL*/2-*wd*). Based on the above statements, the range of each region and the maximum spacing *s* of the shear reinforcement required in each region are described as follows.

Region I: The factored shear force is between (*wL*/2-*wd*) to 3 $\phi_s V_c$

$$s = \min\left(\frac{\phi_s A_v f_y d}{0.5wL - wd - \phi_s V_c},\ \frac{A_v f_y}{50b},\ \frac{d}{4},\ 12\right)\ inch \tag{12}$$

Region II: The factored shear force is between 3 $\phi_s V_c$ and $\phi_s V_c$.

$$s = \min\left(\frac{\phi_s A_v f_y d}{3\phi_s V_c - \phi_s V_c},\ \frac{A_v f_y}{50b},\ \frac{d}{2},\ 24\right)\ inch \tag{13}$$

Region III: The factored shear force is between $\phi_s V_c$ and 0.5 $\phi_s V_c$.

$$s = \min\left(\frac{A_v f_y}{50b},\ \frac{d}{2},\ 24\right)\ inch \tag{14}$$

Region IV: The factored shear force is less than 0.5 $\phi_s V_c$. No shear reinforcement is required according to the ACI code.

Once the minimum spacing of stirrups in each region is found, the total number and amount of stirrups required in the beam can be obtained.

Deflection

Serviceability of a structure is determined by its deflection, cracking, extent of corrosion of its reinforcement and surface deterioration of its concrete. This paper only deals with deflection. The maximum instantaneous deflection in an elastic simple beam caused by dead load plus live load can be expressed as

$$\Delta_{iDL} = \frac{5(w_D + w_L)L^4}{384 E_c I_e} \tag{15}$$

If there is only dead load applied to the elastic beam, the maximum instantaneous deflection

$$\Delta_{iD} = \frac{5(w_D)L^4}{384 E_c I_e} \tag{16}$$

and the instantaneous deflection due to live load can then be obtained by subtracting Equation (16) from Equation (15), that is,

$$\Delta_{iL} = \Delta_{iDL} - \Delta_{iD} \tag{17}$$

Table 1. Some optimal results from the genetic algorithm.

f_y (ksi)	f'_c (psi)	*L* (ft)	w_d (lb/ft)	*dist*1/spacing (in/in)	(*dist*2-*dist*1)/*spacing* (in/in)	(*dist*3-*dist*2)/*spacing* (in/in)	*b* (in)	*d* (in)	ρ	*C* (10^3NT$)
40	3000	19.68	1410	24.56/6.14	(43.76-24.56)/12.28	(80.93-43.76)/12.28	9.86	24.56	0.0108	3.723
40	4000	26.24	1545	32.63/8.16	(48.11-32.63)/16.32	(102.8-48.11)/16.32	9.87	32.64	0.0111	6.459
50	4000	26.24	1545	34.48/8.62	(40.94-34.48)/17.24	((99.21-40.94)/17.24	9.96	34.48	0.0078	5.928
50	5000	32.8	1815	46.35/11.59	(55.19-46.35)/23.17	(126.02-55.19)/23.17	8.74	46.35	0.0083	8.761
60	3000	19.68	1410	32.53/8.13	(38.19-32.53)/16.26	(78.15-38.19)/16.26	8.0	32.53	0.0049	3.033
60	3000	32.8	1545	51.42/11.81	(77.76-51.42)/23.62	(137.3-77.76)/23.62	7.87	51.42	0.0059	7.863

In Equations (15) and (16), E_c is the modulus of elasticity of concrete, which is equal to $57000\sqrt{f'_c}$ psi, and I_e is the effective moment of inertia, a smooth transition between the moment of inertia I_{cr} of the cracked section and the moment of inertia I_g of the gross uncracked concrete section. The effective moment of inertia is defined as

$$I_e = \left(\frac{M_{cr}}{M_a}\right)^3 I_g + \left[1-\left(\frac{M_{cr}}{M_a}\right)^3\right] I_{cr} \le I_g \tag{18}$$

Where M_a is the maximum moment along the simple beam and

$$M_{cr} = \frac{I_g f_r}{h/2} = \frac{15 I_g \sqrt{f'_c}}{h} \tag{19}$$

is the cracking moment. Suppose that the simple beam will support or be attached to nonstructural elements likely to be damaged by large deflections. The ACI code stipulates that sum of long-term deflection due to the sustained dead load plus immediate deflection due to the live load has to be less than *L*/480, that is,

$$\Delta_{sum} = \lambda\Delta_{iD} + \Delta_{iL} \le \frac{L}{480} \tag{20}$$

Where λ is a multiplying factor considering long-term loading and shrinkage. According to the ACI code:

$$\lambda = \frac{T}{1+50\rho'} \tag{21}$$

Where ρ' is the compression reinforcement ratio and *T* is a time-dependent factor. Because ρ' is zero for a singly reinforced beam and 5 years or more are considered, the multiplying factor λ = 2.

RESULTS

The width *b* and effective depth *d* of the beam and tension reinforcement ratio ρ are the three variables for genetic algorithms. The fitness function is the total cost in New Taiwan Dollars of the tension reinforcement, stirrups and concrete. The inequality and equality constraints are formulated according to the requirements discussed in Sec. 3. Based on the often-used materials and customs in Taiwan, this paper selects three kinds of yield strength f_y of the tension reinforcement: 40 ksi (2.8 ton/cm^2), 50 ksi (3.5 ton/cm^2) and 60 ksi (4.2 ton/cm^2), three kinds of compressive strength f'_c of the concrete: 3000 psi (0.21 ton/cm^2), 4000 psi (0.28 ton/cm^2)and 5000 psi (0.35 ton/cm^2), three kinds of span *L*: 19.68 ft (6 m), 26.24 ft (8 m) and 32.8 ft (10 m) and four kinds of dead load w_d: 1410 lb/ft (2.1 ton/m), 1545 lb/ft (2.3 ton/m), 1680 lb/ft (2.5 ton/m) and 1815 lb/ft (2.7 ton/m). For simplicity, fix the live load at 1210 lb/ft (1.8 ton/m). Accordingly, there are 108 combinations of beams to be designed.

Genetic algorithms

To run the genetic algorithm of MATLAB, some parameters need to be selected. Here are the values used in this paper: The population size is set to be 20, crossover rate 0.8, and elite number 2. Furthermore, all the individuals are encoded as real numbers; "Rank" is used as the scaling function that scales the fitness values based on the rank of each individual; "Roulette" is the selection function to choose parents for the next generation; The crossover function applies the "Single Point Strategy" to form a new child for the next generation; The "Adaptive Feasible Function" is chosen as the mutation function to make small random changes in the individuals and ensure that linear constraints and bounds are satisfied. Taken as examples, some of the optimal results are listed in Table 1, where *dist1* represents the range of region I, *dist2- dist1* the range of region

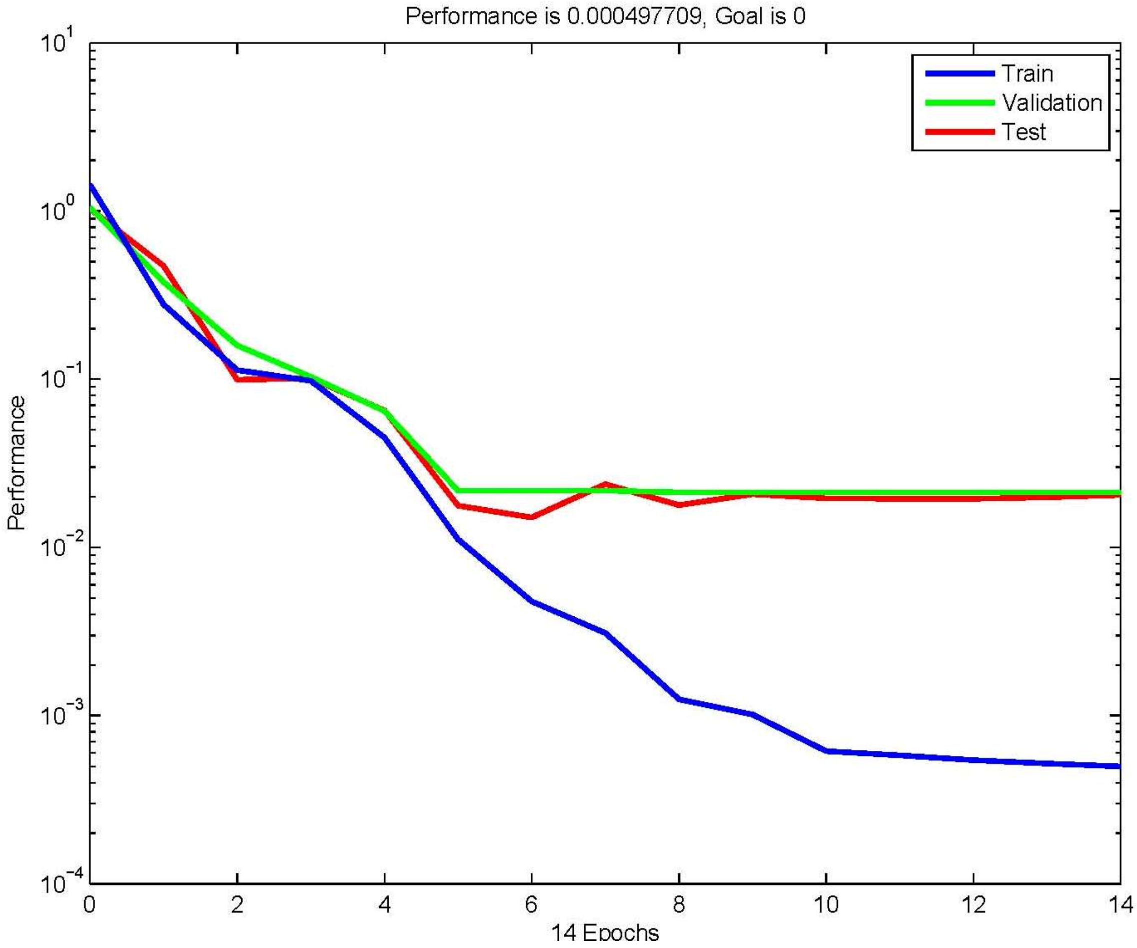


Figure 4. The training process of the neural network.

II, and *dist3-dist2* the range of region III, as indicated in Figure 3, and *spacing* represents the stirrup spacing in each region.

Neural networks

For the purpose of training and testing the neural networks, the optimal results of the 108 beams are divided into three sets: training set (68 data), validation set (20 data) and test set (20 data). The input vector of the neural network consists of six elements: f_y, f'_c, w_d, L, b, and d, and the targets are the minimum price C and tension reinforcement ratio ρ. The process of training is shown in Figure 4, where the training stops at epoch 14 and the performance function is minimized to be 4.9771e-4. After the neural network is trained, the test data is input to it to do simulation. Regression analysis of the network outputs and desired outputs (targets) are then carried out to characterize the network accuracy. The slope *m* and *y*-intercept of the linear regression as well as the correlation coefficient are shown in Table 2. Figures 5 and 6 show the network outputs and targets of the 20 test data for C and ρ, respectively.

Conclusions

Using the optimal results from the genetic algorithm as the data to train and test the neural network, this paper successfully acquires a model to predict the steel ratio and lowest cost of singly reinforced concrete simple beams. As long as the desired width, effective depth, and span of the beam, dead and live load the beam is subjected to, the yield strength of the tension reinforcement as well as the compressive strength of the concrete are provided, the neural network will immediately forecast the minimum cost and the tension reinforcement ratio of a singly reinforced concrete simple beam for quick reference. In addition, by means of inputting those already known values into a computer program written according to the formulas earlier described, the spacing and number of the stirrups can instantly be obtained, which completes the design of the

Table 2. Regression analysis of the network outputs and targets for the test data.

Parameters targets	Slope of the linear regression.	Y-intercept of the linear regression	Correlation Coefficient
Steel Ratio ρ	1.0352	-0.00080854	0.9832
Minimum Cost *C*	0.9092	0.5053	0.9977

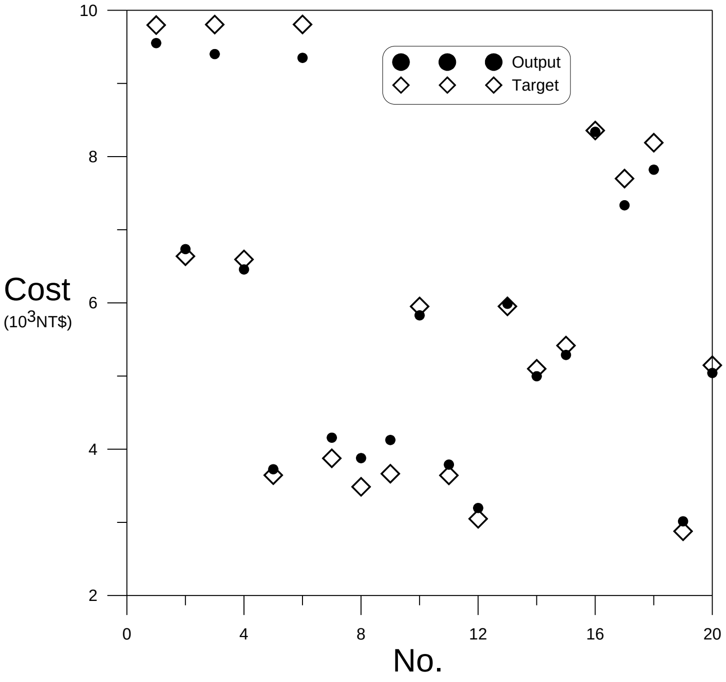


Figure 5. The network outputs and targets of 20 test data for the cost *C.*

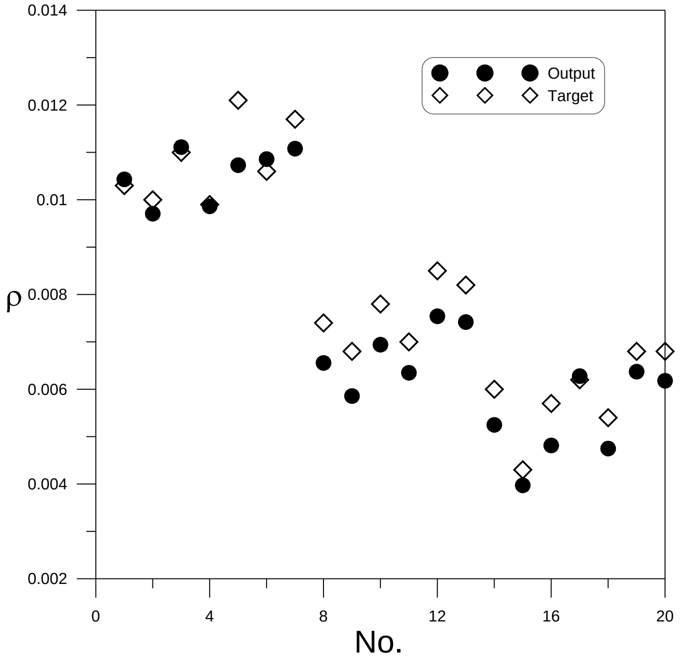


Figure 6. The network outputs and targets of the 20 test data for the tension reinforcement ratio ρ.

simple beam.

REFERENCES

ACI committee 318 (2008). Building Code Requirements for Structural Concrete (ACI 318-08) and Commentary (ACI 318R-08), American Concrete Institute, Farmington Hills, Michigan.

Coello CA, Christiansen AD (2000). Multiobjective Optimization of Trusses Using Genetic Algorithms, Comp. Struct., 75:647-660.

Conn AR, Gould NIM, Toint PhL (1991). A Globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds, SIAM J. Numer. Anal., 28(2):545-572.

Conn AR, Gould NIM, Toint PhL (1997). A Globally Convergent Augmented Lagrangian Barrier Algorithm for Optimization with General Inequality Constraints and Simple Bounds, Math. Comput., 66(217):261-288.

Deb K (1991). Optimal Design of a Welded Beam via Genetic Algorithms, AIAA J., 29:2013-2015.

Hagan MT, Menhaj M (1994). Training Feedforward Networks with the Marquardt Algorithm, IEEE Trans. Neural Netw., 5(6): 989-903.

Hebb DD (1949). The Organization Behavior, Wiley, New Work.

Holland JH (1975). Adaptation in Natural and Artificial Systems, the University of Michigan Press, Ann Arbor, Michigan

Hopfield JJ, Feinstein DI, Palmers RG (1983). Unlearning Has a Stabilizing Effect in Collective Memories, Nature, 304: 158-159.

Hopfield JJ (1982). Neural Networks and Physical Systems with Emergent Collective Computational Abilities, Proc. Natl. Acad. Sci. USA, 79(8): 2554-2558.

Jenkins WM (1992). Plane Frame Optimum Design Environment Based on Genetic Algorithm, ASCE J. Struct. Eng., 118(11):3103-3113.

Levenberg K (1944). A Method for the Solution of Certain Problems in Least Squares, Quarter. Appl. Math., 2:164-168.

Marquardt D (1963). An Algorithm for Least Squares Estimation of Nonlinear Parameters, SIAM J. Appl. Math.,11: 431-441.

McCulloch WS, Pitts W (1943). A Logical Calculus of Ideas Immanent in Nervous Activity, Bull. Math. Biophys., 5:115-133.

Rajeev S, Krishnamoorthy CS (1992). Discrete Optimization of Structures Using Genetic Algorithms, ASCE J. Struct Eng., 118(5): 1233-1250.

Rosenblatt F (1958). The Perception: A Probabilistic Model for Information Storage and Organization in the Brain, Psychol. Rev., 65(6): 386-408.

Rumelhart DE, McClelland JL (1986). Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1 (Foundations), the MIT Press, Cambridge, Massachusetts.

The MathWorks (2010). Global Optimization Toolbox 3 User’s Guide, The MathWorks, Inc., Natick, Massachusetts.

Wackerly DD, Mendenhall W, and Scheaffer RL (2008). Mathematical Statistics with Applications, Cengage Learning, Boston, Massachusetts.