Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 10 (6): 451-461, 2010
ISSN 1812-5654

Calculation of Plate Natural Frequency by Genetic Programming

[1]K. Kumarci, [2]P.K. Dehkordi and [3]I. Mahmodi
[1]Sama Organization (Affiliated With Islamic Azad University), Shahr-e-Kord Branch, Iran
[2]Islamic Azad University, Shahrekord Branch, Iran
[3]Medical Science University of Shahrekord, Shahrekord-Iran, Iran

Abstract: This research emphasizes the study of plate natural frequency by Genetic Programming (GP). In this study we tried to investigate the modeling of plates and modal analysis by finite element method and FEM software. Six parameters were used through plate modeling: length, width, thickness, poison's ratio, modulus of elasticity and density. This study intends to estimate the natural frequency of seven different metals as follow: high strength low alloy steel, gray iron, aluminum by alloy 1100-H14 and 2014-T0 and copper, cold rolling steel and malleable iron, which were different for metals. After frequency estimation by fem software, 100 samples of each metal were produced and GP was used for modal training to choose the best model in metal modeling. The mentioned model specifications have been produced mathematically. Afterwards, 100 extra samples were produced by FEM software. Using obtained models, all of results were tested. The error percentage of GP models and FEM software was compared and the results were produced as diagram. Considering spread of matters, all of stages are indicated only for high strength low alloy steel and the list of the best GP models for another metal is indicated in conclusion. Finally, due to results, modal analysis simulation in plate natural frequency was introduced by GP as a new method by GP.

Key words: Plate, natural frequency, genetic programming, algorithm genetic, modal analysis

INTRODUCTION

The goal of modal analysis in structural mechanics is to determine the natural mode shapes and frequencies of an object or structure during free vibration. It is common to use the Finite Element Method (FEM) to perform this analysis because, like other calculations using the FEM, the object being analyzed can have arbitrary shape and the results of the calculations are acceptable. The types of equations which arise from modal analysis are those seen in Eigen systems. The physical interpretation of the Eigen values and eigenvectors which come from solving the system are that they represent the frequencies and corresponding mode shapes. Sometimes, the only desired modes are the lowest frequencies because they can be the most prominent modes at which the object will vibrate, dominating all the higher frequency modes.

It is also possible to test a physical object to determine its natural frequencies and mode shapes. This is called an Experimental Modal Analysis. The results of the physical test can be used to calibrate a finite element model to determine if the underlying assumptions made were correct (for example, correct material properties and boundary conditions were used).

The study field of plate natural frequency is much spread. Among them, practical natural frequency analysis of elastic plate on water (Hiroaki *et al*., 2001), modal vibration analysis of metal plate by using a laser vibrometer and POD (proper orthogonal decomposition) method (Barrientos *et al*., 2005) the finite element analysis in plate and beam, experimental analysis of modal interactions in the nonlinear vibrations of a plate, application of high frequency barrier discharge for flat plate drag reduction (Shatan, 2009) and calculation of plate natural frequency by neural network (Ziaie *et al*., 2008), which has been used a three layer new elm neural network, by 8 neurons in input layer and 8 neurons in the middle layer, Train cgb training function, learning function of learn P and performance evaluation see in plate frequency calculation. Regard to investigation, it has been tried to calculate plate natural frequency by GP.

Evolutionary algorithms are inspired by nature. The idea is to mimic the natural evolution of the spices in order to create a new kind of search technique that is robust and intelligently seeks solutions in a possibly infinite search space (Mitchell, 1996). Some of the techniques that are part of this branch of computer science are Genetic Algorithm (GA), Genetic Programming (GP) and Evolutionary Programming (EP). Genetic Programming

Corresponding Author: Kaveh Kumarci, Sama Organization (Affiliated With Islamic Azad University), Shahr-e-Kord Branch, Iran

(GP) is an extension to Genetic Algorithms proposed by Koza (1994a), to automatically extract intelligible relationships in a system and has been used in many applications such as symbolic regression (Ong *et al*., 2005) also Koza (1992) defines GP in following steps:

- Generate an initial population of random compositions of the functions and terminals of the problem (computer programs)
- Execute each program in the population and assign it a fitness value according to how well it solves the problem
- Create a new population of computer programs
  - Copy the best existing programs (reproduction)
  - Create new computer programs by mutation
  - Create new computer programs by crossover (sexual reproduction)
  - Select an architecture-altering operation from the programs stored so far
- The best computer program that appeared in any generation, the best-so-far solution, is designated as the result of genetic programming

The GP creates a population of computer programs with a tree structure. Randomly generated programs are general and hierarchical, varying in size and structure. The GPs main goal is to solve a problem by searching highly fit computer programs in the space of all possible solutions. This aspect is the key for finding near global optimum solutions by keeping many solutions that may potentially to be close to minima (local or global). The creation of initial population is a blind random search of the space defined by the problem. The output of the GP is a program rather than a quantity (Asbour *et al*., 2003). The GP has several advantages over the more conventional multivariate analysis or distance-based classifier methods used in data analysis and it has the potential to discover simple rules in data that are both complex and noisy. No assumptions of normality or independence are required about the data to be analyzed. It is possible to combine numerical, ordinal and categorical data in the same analysis and the evolved models can easily be applied to new data. The identification and selection of a small subset of the available variables that has high explanatory power are particularly useful. This research investigates to predicting models with a new approach method called genetic programming.

MATERIALS AND METHODS

Natural angular frequency: This study intends to estimate the natural frequency of seven different metals and after frequency estimation by FEM software, 100 samples of each metal were produced and GP was used for modal training to choose the best model in metal modeling. The mentioned model specifications have been produced mathematically. Afterwards, 100 extra samples were produced by FEM software (Jahed *et al*., 2003). Using obtained models, all of results were tested. For doing of these stages of the study which have been conducted in Sama organization (Shahrekord branch) and also Shahid Bahonar university of Kerman, almost two years (2007 to 2009) have been spent.

Considering a structure, if deformation initiate, it begins to vibrate. If there is no external force, the structure will be under free vibration. For estimation of structure dynamic response, vibration properties (natural frequency and vibration modes) are needed. These properties are provided as below:

$$k\phi = \omega^2 m\phi \quad (1)$$

which is a defined problem. Where, K, M, ω and ϕ are stiffness matrix, structure mass matrix natural angular frequency and vibration mode, respectively. M and K matrixes are estimated by matrix analysis technique. There are many numerical techniques to solve defined problem. Choosing the best computer technique is related to mass and stiffness matrix properties, natural frequency and vibration modes which are necessary in structure analysis.

Structure frequency importance: Structure frequency estimation and attached it to defined quantities (or maximize them) have two advantages; decrease of structure vibration range and prevention of resonance in structure dynamic response. Resonance is a condition in which initiated load is equal to vibration natural frequency. So, in spite of sustaining anticipated efficiency and performance, changing the structure dimensions or shape change, the natural frequency and dynamic excitation will increase and decrease, respectively. This factor decreases stress and rise, therefore structure safety will be provided.

Stiffness matrix and plates mass: A plate bends under vertical loads, so we say it is in flexural state. Here, we will deal shearing and flexural stresses and the strains which are similar to beam stress and strain. Where plate analysis is more complex, because it is 2-D but beam is one-dimensional. On the other hand, metal plate is a three dimensional shell, so metal plate analysis is more difficult because in addition of shells shearing and flexural stresses and strains, we should consider membranous deformation strains.

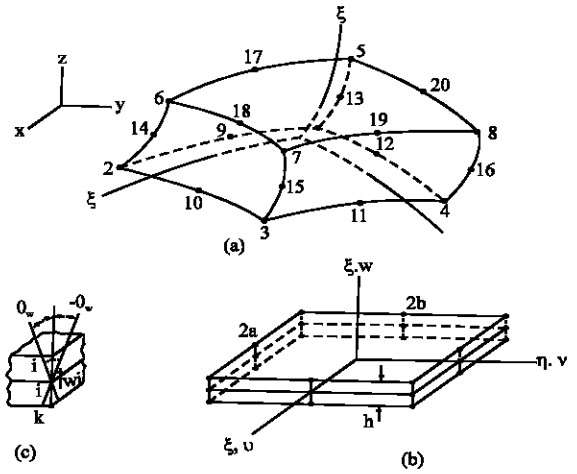

Fig. 1: Special manner of hexahedron, (a) H20 element, (b) PBQ8 rectangular pattern before bracing, (c) restricted nodes displacement

Due to volume, finite elements have axisymetry in plates and shell analysis such as midline plates theory. Automatically, these special specifications include the effects of shearing deformation and torsional stiffness. It is assumed that all of analytical structures are made of linear elasticity material with small strains and displacements.

Elements of plates in flexural state: Figure 1a shows the main member of H20 which is defined by geometrical interpolated quadratics. For perceives of required constraints for change it to flexural elements; we make flat cuboids with natural coordinates. Obtained element is shown in Fig. 1b and is as a rectangular pattern, PQR8, from PBQ8 element before constriction. It must be remembered that three nodal groups are located in angles, but two nodal groups are in the middle of side in PQR8 element. In the Fig. 1c, it is seen that with special constraints instigation, we can change each group and twin nodal to single nodal in the middle of area.

It is probable that isoparameter hexahedron limited, so a dimension is small versus two other dimensions. In this case, hexahedron changed to shell element or plane. For analysis of flat plane, it's necessary to limit reformed dimensions to put them in a single area. This part allocated to isoperimetric hexahedron (H20) that changes to flexural tetrahedral plane, PBQ8. Figure 1a-c show special manner of hexahedron H20 elements, PBQ8 rectangular pattern before bracing and restricted nodes displacement.

Regarding fore mentioned descriptions, we can write stiffness matrix in PBQ8 element as below:

$$K=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}B^{T}EB|J|d\xi d\eta d\zeta=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\left(B_{A}+\zeta\frac{h}{2}B_{B}\right)^{T}E\left(B_{A}+\zeta\frac{h}{2}B_{B}\right)^{T}E\left(B_{A}+\zeta\frac{h}{2}B_{B}\right)|J|d\xi d\eta d\zeta \quad (2)$$

Mass matrix which is suitable for PBQ8 is as below:

$$M=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}f^{T}f|J|d\xi d\eta d\zeta=\rho\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\left(f_{A}+\zeta\frac{h}{2}f_{B}\right)^{T}\left(f_{A}+\zeta\frac{h}{2}f_{B}\right)|j|d\xi d\eta d\zeta \quad (3)$$

Equivalent panel loads which are rise from PBQ8 element loads estimate by this formula:

$$P_{b}(t)=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}f_{A}^{T}b(t)|J|d\xi d\eta d\zeta=2\int_{-1}^{1}\int_{-1}^{1}f_{A}^{T}b(t)|J|d\xi d\eta \quad (4)$$

Modeling by FEM software: Plate modeling and analysis were performed by finite element method using FEM software. Shell93 element is three layered shell which has been used in modeling. It is a 3-D shell element which has 8 nodes and 6 degrees freedom for each of them (3 degrees for transferring and 3 degrees for rotational displacements). It is used in plate and shell modeling (with flexural behavior). The FEM software analysis is modal and linear, so nonlinear properties such as pozzolanicity and contact element are neglect able. In this study, three different supports are used for plate:

- A plate with fixed support in two opposite side
- A plate with fixed support around it
- A plate with fixed support in the corners

Genetic programming: Genetic Programming (Banzhaf *et al*., 1998; Koza, 1990, 1992, 1994a, b) is a recent development which extends classical genetic algorithms (Back *et al*., 2000a, b; Mitchell, 1996) to process nonlinear structures. This optimization technique is based on the principles of natural evolution and is consisted of several genetic operators: selection, crossover and mutation.

The major difference between genetic programming and genetic algorithms is the representation of the solution candidates (Fig. 2).

Node definition: The nodes in the tree structure of genetic programming can be classified into two types. One of them is the terminal set which is consisted of constants or variables. The other one is the function set which is consisted of standard arithmetic operations, standard programming operations, standard mathematical functions, logical functions, or domain-specific functions.

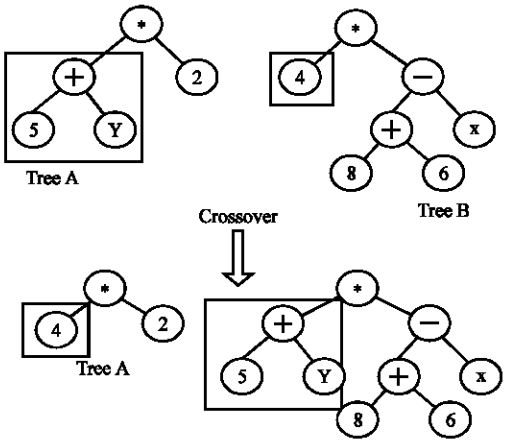

Fig. 2: A tree in genetic programming represents a formula

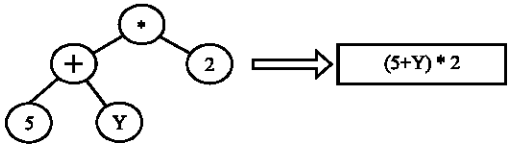

Fig. 3: An example of sub tree crossover

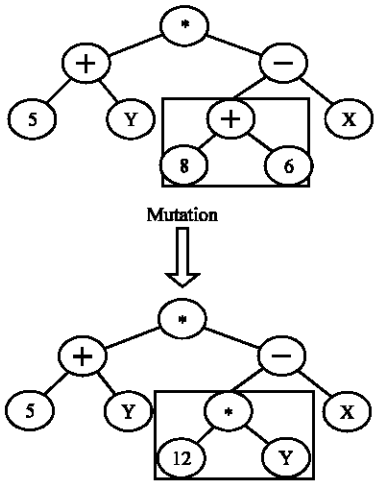

Fig. 4: An example of sub tree replacement mutation

The elements of the terminal set and the function set are used to construct well-formed expression trees, which represent solutions to the problem, according to certain rules.

Initialization: Genetic programming starts with an initial population of expression trees which are randomly generated.

Fitness evaluation: Fitness evaluates how well a tree performs in the problem environment. Fitness values are used by the selection method to select trees for reproduction.

Selection: The selection method determines how trees are selected from the population to be parents for crossover. Better parents are usually selected with the hope that they have a better chance of producing better off springs. The roulette wheel selection is a popular selection method. In Fig. 3 and 4, the most common crossover operator and also an example of sub tree replacement mutation are presented.

RESULTS AND DISCUSSION

In artificial intelligence, Genetic Programming (GP) is an evolutionary algorithm-based methodology ispired by biological programs to find computer programs that perform a user-defined task. It is a specialization of Genetic Algorithm (GA) where each individual is a computer program. Therefore, it is a machine learning technique used to optimize a population of computer programs according to a fitness landscape determined by a program's ability to perform a given computational task.

As it said before, GPs main goal is to solve a problem by searching highly fit computer programs in the space of all possible solutions and finding near global optimum solutions by keeping many solutions. This method several advantages over the more conventional multivariate analysis or distance-based classifier methods used in data analysis and it has the potential to discover simple rules in data that are both complex and noisy. No assumptions of normality or independence are required about the analyzing frequencies data. It is possible to combine numerical, ordinal and categorical data in the same analysis and the evolved models can easily be applied to new data. The identification and selection of a small subset of the available variables that has high explanatory power are particularly useful. Considering of the first steps of applying, precision is the only weakness point of this is method which can be improved by mentioned points.

Genetic programming is a branch of genetic algorithms. The main difference between genetic programming and genetic algorithms is the representation of the solution. Genetic programming creates computer programs in the lisp or scheme computer languages as the solution. Genetic algorithms create a string of numbers that represent the solution. So, genetic programming is much more powerful than genetic algorithms. The output of the genetic algorithm is a quantity, while the output of the genetic programming is a another computer program. In essence, this is the beginning of computer programs

that program themselves genetic programming uses four steps to solve problems:

- Generate an initial population of random compositions of the functions of terminals of the problem (computer programs)
- Execute each program in the population and assign it a fitness value according to how well it solves the problem
  - Create a new population of computer programs
  - Copy the best existing programs
  - Create new computer programs by mutation.
  - Create new computer programs by crossover (sexual reproduction)
- The best computer program that appeared in any generation, the best-so-far solution, is designated as the result of genetic programming (Koza, 1992)

A hierarchical tree structure is used in Fig. 2.

Figure 2 shows a solution candidate in genetic programming while a string of characters with a fixed length represents a solution candidate in genetic algorithms. The genetic programming framework consists of the flowing elements: node definition, initialization, fitness evaluation, selection, crossover, mutation and termination condition.

Node definition: The nodes in the tree structure of genetic programming can be classified into two types. One of them is the terminal set which is consisted of constants or variables. The other one is the function set which is consisted of standard arithmetic operations, standard programming operations, standard mathematical functions, logical functions, or domain-specific functions. The elements of the terminal set and the function set are used to construct well-formed expression trees, which represent solutions to the problem, according to certain rules.

Initialization: Genetic programming starts with an initial population of expression trees which are randomly generated.

Fitness evaluation: The most difficult and most important concept of genetic programming is the fitness function. The fitness function determines how well a program is able to solve the problem. It varies greatly from one type of program to the next. For example, if one were to create a genetic program to set the time of a clock, the fitness function would simply be the amount of time that the clock is wrong. Unfortunately, few problems have such an easy fitness function; most cases require a slight modification of the problem in order to find the fitness.

Selection: The selection method determines how trees are selected from the population to be parents for crossover. Better parents are usually selected with the hope that they have a better chance of producing better off springs. The roulette wheel selection is a popular selection method.

Crossover: Two primary operations exist for modifying structures in genetic programming. The most important one is the crossover operation. In the crossover operation, two solutions are sexually combined to form two new solutions or offspring. The parents are chosen from the population by a function of the fitness of the solutions. Three methods exist for selecting the solutions for the crossover operation.

In GP, sub tree crossover is the most common crossover operator. Sub tree crossover works by replacing a sub tree in one parent with a sub tree from other parent to produce the offspring. Figure 3 presents an example of this sub tree.

Mutation: In GP, mutation is usually achieved by replacing a sub tree with a randomly generated sub tree or by exchanging two randomly selected sub trees. An example of sub tree replacement mutation is presented in Fig. 4.

Termination condition: Common termination conditions include fixed generations, fitness target, fitness convergence and diversity convergence.

Genetic programming pattern: Modeling using GP needs exact pattern. This pattern includes primary ordering and determine of mathematical operator.

Primary ordering: Amounts of data and variables, plate ordering, linking function of gene, determination of fitness function and genetic operator and also constant values are some required related data for Gp primary ordering.

Mathematical operators: The perfect list of mathematical operator in modeling is shown in appendix .

Data modeling: Metal models provided by primary data and GP training, separately. Finally, the best model and the perfect details will be provided. Considering spreading of matters, we provided only the best model for high strength low alloy steel. For the best GP modeling of high strength low alloy steel, C^{++} computer codes is used.

Seven different metals were modeled by FEM and 100 samples were produced for each, then provided to GP for training to choose the best model for modeling. Afterwards, the results defined mathematically. The characteristics of 100 plate samples are used in GP training as data, which are different for each plate. Each plate has 6 parameters: length, width, thickness and modulus of

Table 1: General characteristics in used metal

Plate	Poisson's ratio (v)	Modulus of elasticity (E)	GPA density γ (kg m^{-3})
High strength low alloy steel	0.30	200	7860
Gray iron	0.29	69	7200
Aluminum by alloy 1100-H14	0.33	70	2710
Copper	0.34	120	8910
Cold rolling steel	0.30	190	7920
Malleable iron	0.29	165	7300
Aluminum by alloy 2014-T6	0.33	75	2800

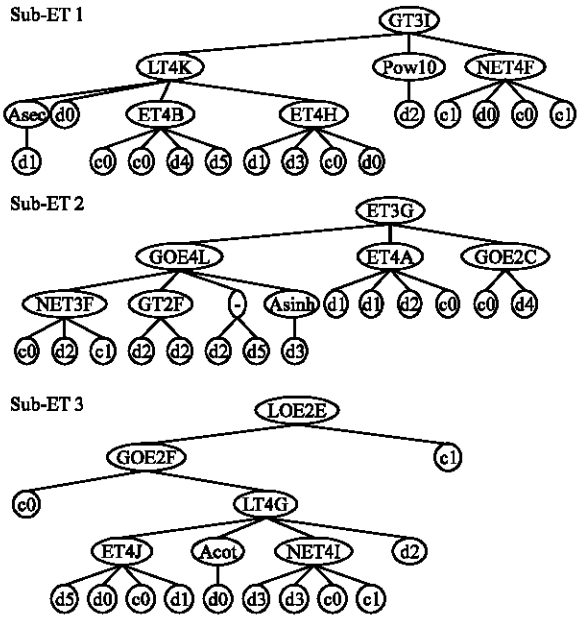

Fig. 5: Terminology tree of the best GP model in high strength low alloy steel

elasticity, density and Poisson's ratio. Altering of plate length, width and thickness are 0.5-12, 0.5-4 and 0.002-0.22 m, respectively. Modulus of elasticity and Poisson's ratio are shown in Table 1.

Genetic programming pattern: Modeling using GP needs exact pattern. This pattern includes primary ordering and determine of mathematical operator.

Primary ordering: Amounts of data and variables, plate ordering includes chromosome and genes number, size of head and tail, size and place of division and linking function of genes, determine of fitness function, determine of genetic operator, determine of constant values are some required related data for GP primary ordering.

Figure 5 shows the terminology tree of the best GP model in high strength low alloy steel.

Mathematical formula details of high strength low alloy steel modeling are represented as follow (Fig. 6).

Model testing: For test of obtained models, another 100 samples are provided. Due to programming in GP model test, 2nd set of data should be provided. Then FEM frequency results were compared with the best GP model. Considering spreading matter, only the best model of high strength low alloy steel modeling is provided.

After the mentioned stages in and also choosing 100 samples and providing 2nd set of data, due to programming in GP model, the results have been tested and then FEM frequency results were compared with the

Generation	Program size	Uterals	Used variables	Training fitness	Testing fitness	Training R^2	Testing R^2
4505	62	26	ara(5), e(5), game(7), nou(4), tol(2), zerkamat(3)	878.616357565559	-	0.992727998034072	-

GT3I.LT4K.Pox10.NET4F.Asec.d0.ET4B.ET4H.d2.c1.d0.c0.c1.d1.c0.c0d4.d5dd1.d3.c0.d0.c1d2.d1.c0.c0.d1.d3.d5.c0.c1.c0
+
ET3G.GOE4L.ET4A.COE2C.NET3F.GT2F.-.Asinh.d1.d1.d2.c0.c0.d4.c0.d4.c0.d2.c1.d2.d2.d2.d5.d3.c1.d1.d2.d1.d1.c1.d3.c0.d4.d2.d5
+
LOE2E.COE2F.c1.c0.LT4G.ET4J.Acot.NET4I.d2.d5.d0.c0.d1.d0.d3.d3.c0.c1.c0.d0.d0.d1.d5.c0.d4.d3.d1.d4.d4.d1.d3.c0.c1

Numerical constanst:

Gene 1
c0 = -4.651337
c1 = 1.226165

Gene 2
c0 = -3.435394
c1 = -8.430298

Gene 3
c0 = 6.808777
c1 = 4.538727

Fig. 6: Mathematical formula details in modelling of low alloy steel with high strength

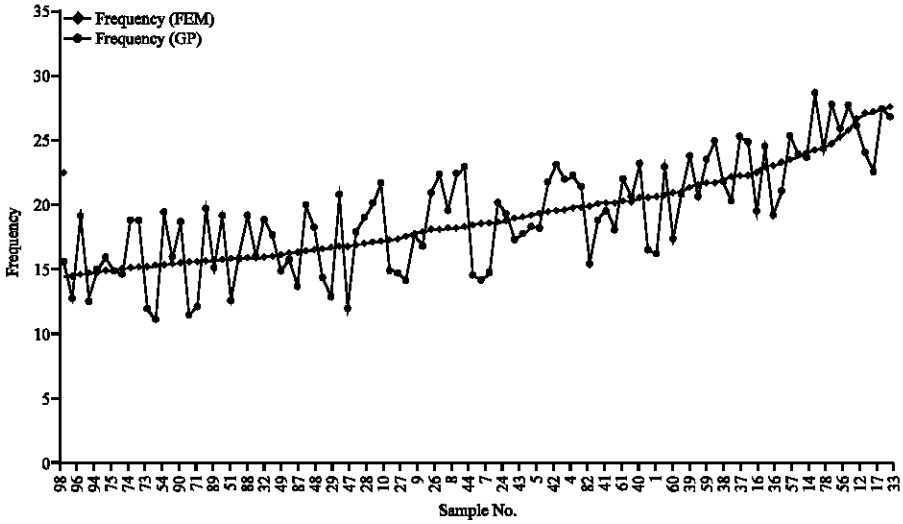

Fig. 7: Comparing frequency of GP and FEM model in high strength low alloy steel

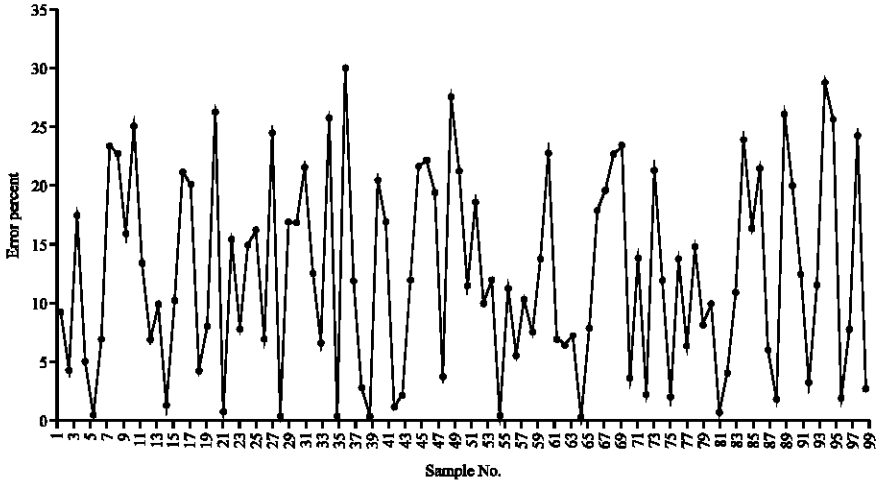

Fig. 8: Comparing the error percentage in high strength low alloy steel modeling

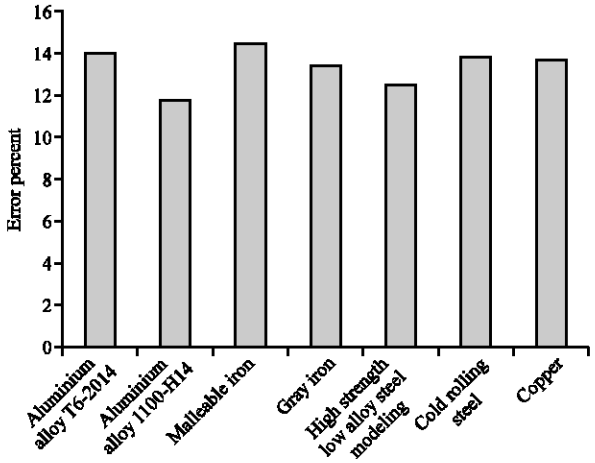

Fig. 9: Comparing error percentage in metals programming model

best GP model. Comparing frequency and Error Percent of GP and FEM model in high strength low alloy steel are shown in Fig. 7 and 8.

Comparing frequency of GP and FEM model in high strength low alloy steel and also comparing of the error percentage in high strength low alloy steel modeling are shown in Fig. 7 and 8. Also Fig. 9 shows the comparing error percentage in metals programming model.

Comparing error percentage in metals programming model it can be seen that frequency of GP and FEM model are close to each other in a good accuracy so that we can use GP model in metal frequency calculating.

CONCLUSION

This research is an investigation to find a new analyzing method for complex and noisy series of data

where Finite Element Method (FEM) is the most common method to perform the analyzing of different series of engineering data. The FEM is a reliable analyzing method because of having the arbitrary shape and also the acceptable results of the calculations, but applying of it in complex and noisy data needs much more time and will be complex to. This study is an investigation to solve a sample problem of engineering problems in this way. Predicting and calculating of metal frequency with a new approach method of genetic programming is the main goal of this research. In this study, a new computer method, which has the potential to discover simple rules in a complex and noisy series of metal plate's frequency data, is used. In other words, this research tries to apply and find a usage of this new specific computer method in civil, mechanic and other related field of engineering.

ACKNOWLEDGMENTS

Thanks are due to Sama Organization, Azad University, Medical Science University of Shahrekord and also author's colleagues because of their kind supports and also all of the persons who gave us permission to made some of their studies, results and illustrations have been reproduced in this Study.

APPENDIX

Appendix: Complete Operator list

Select/Weight	Name	Representation	Arity	Definition
☑4	Addition	+	2	(x+y)
☑4	Subtraction	-	2	(x-y)
☑4	Multiplication	*	2	(x*y)
☑1	Division	/	2	(x/y)
☑1	Floating point remainder	Mod	2	mod(x,y)
☑1	Power	Pow	2	pow(x,y)
☑1	Square root	Sqrt	1	sqrt(x)
☑1	Exponential	Exp	1	exp(x)
☑1	10^x	Pow10	1	pow(10,x)
☑1	Natural logarithm	Ln	1	ln(x)
☑1	Logarithm of base 10	Log	1	log(x)
☑1	Logarithm(x,y)	Log2	2	log(x,y)
☑1	Floor	Floor	1	floor(x)
☑1	Ceiling	Ceil	1	ceil(x)
☑1	Absolute value	Abs	1	abs(x)
☑1	Inverse	Inv	1	1/x
☑1	Negation	Neg	1	-x
☑1	No operation	Nop	1	x
☑1	x to the power of 2	X2	1	x^2
☑1	x to the power of 3	X3	1	x^3
☑1	x to the power of 4	X4	1	x^4
☑1	x to the power of 5	X5	1	x^5
☑1	Cube root	3Rt	1	x^(1/3)
☑1	Quartic root	4Rt	1	x^(1/4)
☑1	Quintic root	5Rt	1	x^(1/5)
☑1	Addition with 3 inputs	Add3	3	(x+y+z)

Select/Weight	Name	Representation	Arity	Definition
☑1	Subtraction with 3 inputs	Sub3	3	(x-y-z)
☑1	Multiplication with 3 inputs	Mul3	3	(x*y*z)
☑1	Division with 3 inputs	Div3	3	(x/y/z)
☑1	Addition with 4 inputs	Add4	4	(a+b+c+d)
☑1	Subtraction with 4 inputs	Sub4	4	(a-b-c-d)
☑1	Multiplication with 4 inputs	Mul4	4	(a*b*c*d)
☑1	Division with 4 inputs	Div4	4	(a/b/c/d)
☑1	Minimum of 2 inputs	Min2	2	min(x,y)
☑1	Minimum of 3 inputs	Min3	3	min(x,y,z)
☑1	Minimum of 4 inputs	Min4	4	min(a,b,c,d)
☑1	Maximum of 2 inputs	Max2	2	max(x,y)
☑1	Maximum of 3 inputs	Max3	3	max(x,y,z)
☑1	Maximum of 4 inputs	Max4	4	max(a,b,c,d)
☑1	Average of 2 inputs	Avg2	2	avg(x,y)
☑1	Average of 3 inputs	Avg3	3	avg(x,y,z)
☑1	Average of 4 inputs	Avg4	4	avg(a,b,c,d)
☑1	Logistic(x)	Log1	1	logistic(x)
☑1	Logistic(x,y)	Log2	2	logistic(x,y)
☑1	Logistic(x,y,z)	Log3	3	logistic(x,y,z)
☑1	Logistic(a,b,c,d)	Log4	4	logistic(a,b,c,d)
☑1	Gaussian(x)	Gau	1	gaussian(x)
☑1	Gaussian(x,y)	Gau2	2	gaussian(x,y)
☑1	Gaussian(x,y,z)	Gau3	3	gaussian(x,y,z)
☑1	Gaussian(a,b,c,d)	Gau4	4	gaussian(a,b,c,d)
☑1	Constant zero function	Zero	1	0(x) = 0
☑1	Constant one function	One	1	1(x) = 1

Appendix: Continued

Select/Weight	Name	Representation	Arity	Definition
☑ 1	0(x,y)	Zero2	2	0(x,y) = 0
☑ 1	1(x,y)	One2	2	1(x,y) = 1
☑ 1	Number Pi	Pi	1	Pi(x) = 3.141592653589793
☑ 1	Euler's number	E	1	E(x) = 2.718281828459045
☑ 1	Sine	Sin	1	sin(x)
☑ 1	Cosine	Cos	1	cos(x)
☑ 1	Tangent	Tan	1	tan(x)
☑ 1	Cosecant	Csc	1	csc(x)
☑ 1	Secant	Sec	1	sec(x)
☑ 1	Cotangent	Cot	1	cot(x)
☑ 1	Arcsine	Asin	1	arcsin(x)
☑ 1	Arccosine	Acos	1	arccos(x)
☑ 1	Arctangent	Atan	1	arctan(x)
☑ 1	Arccosecant	Acsc	1	arccsc(x)
☑ 1	Arcsecant	Asec	1	arcsec(x)
☑ 1	Arccotangent	Acot	1	arccot(x)
☑ 1	Hyperbolic sine	Sinh	1	sinh(x)
☑ 1	Hyperbolic cosine	Cosh	1	cosh(x)
☑ 1	Hyperbolic tangent	Tanh	1	tanh(x)
☑ 1	Hyperbolic cosecant	Csch	1	csch(x)
☑ 1	Hyperbolic secant	Sech	1	sech(x)
☑ 1	Hyperbolic cotangent	Coth	1	coth(x)
☑ 1	Inverse hyperbolic sine	Asinh	1	arcsinh(x)
☑ 1	Inverse hyperbolic cosine	Acosh	1	arccosh(x)
☑ 1	Inverse hyperbolic tangent	Atanh	1	arctanh(x)
☑ 1	Inverse hyperbolic cosecant	Acsch	1	arccsch(x)

Select/Weight	Name	Representation	Arity	Definition
☑ 1	Inverse hyperbolic secant	Asech	1	arcsech(x)
☑ 1	Inverse hyperbolic cotangent	Acoth	1	arccoth(x)
☑ 1	Complement	NOT	1	(1-x)
☑ 1	OR1	OR1	2	if x < 0 OR y < 0, then 1, else 0
☑ 1	OR2	OR2	2	if x >= 0 OR y >= 0, then 1, else 0
☑ 1	OR3	OR3	2	if x <= 0 OR y <= 0, then 1, else 0
☑ 1	OR4	OR4	2	if x < 1 OR y < 1, then 1, else 0
☑ 1	OR5	OR5	2	if x >= 1 OR y >= 1, then 1, else 0
☑ 1	OR6	OR6	2	if x <= 1 OR y <= 1, then 1, else 0
☑ 1	AND1	AND1	2	if x < 0 AND y < 0, then 1, else 0
☑ 1	AND2	AND2	2	if x >= 0 AND y >= 0, then 1, else 0
☑ 1	AND3	AND3	2	if x <= 0 AND y <= 0, then 1, else 0
☑ 1	AND4	AND4	2	if x < 1 AND y < 1, then 1, else 0
☑ 1	AND5	AND5	2	if x >= 1 AND y >= 1, then 1, else 0
☑ 1	AND6	AND6	2	if x <= 1 AND y <= 1, then 1, else 0
☑ 1	Less Than with 2 inputs (A)	LT2A	2	if x < y, then x, else y
☑ 1	Greater Than with 2 inputs (A)	GT2A	2	if x > y, then x, else y
☑ 1	Less Or Equal To with 2 inputs (A)	LOE2A	2	if x <= y, then x, else y
☑ 1	Greater Or Equal To with 2 inputs (A)	GOE2A	2	if x >= y, then x, else y
☑ 1	Equal To with 2 inputs (A)	ET2A	2	if x = y, then x, else y
☑ 1	Not Equal To with 2 inputs (A)	NET2A	2	if x != y, then x, else y
☑ 1	Less Than with 2 inputs (B)	LT2B	2	if x < y, then 1, else 0
☑ 1	Greater Than with 2 inputs (B)	GT2B	2	if x > y, then 1, else 0
☑ 1	Less Or Equal To with 2 inputs (B)	LOE2B	2	if x <= y, then 1, else 0
☑ 1	Greater Or Equal To with 2 inputs (B)	GOE2B	2	if x >= y, then 1, else 0
☑ 1	Equal To with 2 inputs (B)	ET2B	2	if x = y, then 1, else 0

Appendix: Continued

Select/Weight	Name	Representation	Arity	Definition
☑ 1	Not Equal To with 2 inputs (B)	NET2B	2	if x != y, then 1, else 0
☑ 1	Less Than with 2 inputs (C)	LT2C	2	if x < y, then (x+y), else (x-y)
☑ 1	Greater Than with 2 inputs (C)	GT2C	2	if x > y, then (x+y), else (x-y)
☑ 1	Less Or Equal To with 2 inputs (C)	LOE2C	2	if x <= y, then (x+y), else (x-y)
☑ 1	Greater Or Equal To with 2 inputs (C)	GOE2C	2	if x >= y, then (x+y), else (x-y)
☑ 1	Equal To with 2 inputs (C)	ET2C	2	if x = y, then (x+y), else (x-y)
☑ 1	Not Equal To with 2 inputs (C)	NET2C	2	if x != y, then (x+y), else (x-y)
☑ 1	Less Than with 2 inputs (D)	LT2D	2	if x < y, then (x*y), else (x/y)
☑ 1	Greater Than with 2 inputs (D)	GT2D	2	if x > y, then (x*y), else (x/y)
☑ 1	Less Or Equal To with 2 inputs (D)	LOE2D	2	if x <= y, then (x*y), else (x/y)
☑ 1	Greater Or Equal To with 2 inputs (D)	GOE2D	2	if x >= y, then (x*y), else (x/y)
☑ 1	Equal To with 2 inputs (D)	ET2D	2	if x = y, then (x*y), else (x/y)
☑ 1	Not Equal To with 2 inputs (D)	NET2D	2	if x != y, then (x*y), else (x/y)
☑ 1	Less Than with 2 inputs (E)	LT2E	2	if x < y, then (x+y), else (x*y)
☑ 1	Greater Than with 2 inputs (E)	GT2E	2	if x > y, then (x+y), else (x*y)
☑ 1	Less Or Equal To with 2 inputs (E)	LOE2E	2	if x <= y, then (x+y), else (x*y)
☑ 1	Greater Or Equal To with 2 inputs (E)	GOE2E	2	if x >= y, then (x+y), else (x*y)
☑ 1	Equal To with 2 inputs (E)	ET2E	2	if x = y, then (x+y), else (x*y)
☑ 1	Not Equal To with 2 inputs (E)	NET2E	2	if x != y, then (x+y), else (x*y)
☑ 1	Less Than with 2 inputs (F)	LT2F	2	if x < y, then (x+y), else sin(x*y)
☑ 1	Greater Than with 2 inputs (F)	GT2F	2	if x > y, then (x+y), else sin(x*y)
☑ 1	Less Or Equal To with 2 inputs (F)	LOE2F	2	if x <= y, then (x+y), else sin(x*y)
☑ 1	Greater Or Equal To with 2 inputs (F)	GOE2F	2	if x >= y, then (x+y), else sin(x*y)
☑ 1	Equal To with 2 inputs (F)	ET2F	2	if x = y, then (x+y), else sin(x*y)
☑ 1	Not Equal To with 2 inputs (F)	NET2F	2	if x != y, then (x+y), else sin(x*y)
☑ 1	Less Than with 2 inputs (G)	LT2G	2	if x < y, then (x+y), else atan(x*y)

Select/Weight	Name	Representation	Arity	Definition
☑ 1	Greater Than with 2 inputs (G)	GT2G	2	if x > y, then (x+y), else atan(x*y)
☑ 1	Less Or Equal To with 2 inputs (G)	LOE2G	2	if x <= y, then (x+y), else atan(x*y)
☑ 1	Greater Or Equal To with 2 inputs (G)	GOE2G	2	if x >= y, then (x+y), else atan(x*y)
☑ 1	Equal To with 2 inputs (G)	ET2G	2	if x = y, then (x+y), else atan(x*y)
☑ 1	Not Equal To with 2 inputs (G)	NET2G	2	if x != y, then (x+y), else atan(x*y)
☑ 1	Less Than with 3 inputs (A)	LT3A	3	if x < 0, then y, else z
☑ 1	Greater Than with 3 inputs (A)	GT3A	3	if x > 0, then y, else z
☑ 1	Less Or Equal To with 3 inputs (A)	LOE3A	3	if x <= 0, then y, else z
☑ 1	Greater Or Equal To with 3 inputs (A)	GOE3A	3	if x >= 0, then y, else z
☑ 1	Equal To with 3 inputs (A)	ET3A	3	if x = 0, then y, else z
☑ 1	Not Equal To with 3 inputs (A)	NET3A	3	if x != 0, then y, else z
☑ 1	Less Than with 3 inputs (B)	LT3B	3	if (x+y) < z, then (x+y), else z
☑ 1	Greater Than with 3 inputs (B)	GT3B	3	if (x+y) > z, then (x+y), else z
☑ 1	Less Or Equal To with 3 inputs (B)	LOE3B	3	if (x+y) <= z, then (x+y), else z
☑ 1	Greater Or Equal To with 3 inputs (B)	GOE3B	3	if (x+y) >= z, then (x+y), else z
☑ 1	Equal To with 3 inputs (B)	ET3B	3	if (x+y) = z, then (x+y), else z
☑ 1	Not Equal To with 3 inputs (B)	NET3B	3	if (x+y) != z, then (x+y), else z
☑ 1	Less Than with 3 inputs (C)	LT3C	3	if (x+y) < z, then (x+y), else (x+z)
☑ 1	Greater Than with 3 inputs (C)	GT3C	3	if (x+y) > z, then (x+y), else (x+z)
☑ 1	Less Or Equal To with 3 inputs (C)	LOE3C	3	if (x+y) <= z, then (x+y), else (x+z)
☑ 1	Greater Or Equal To with 3 inputs (C)	GOE3C	3	if (x+y) >= z, then (x+y), else (x+z)
☑ 1	Equal To with 3 inputs (C)	ET3C	3	if (x+y) = z, then (x+y), else (x+z)
☑ 1	Not Equal To with 3 inputs (C)	NET3C	3	if (x+y) != z, then (x+y), else (x+z)
☑ 1	Less Than with 3 inputs (D)	LT3D	3	if (x+y) < z, then (x+y), else (x-z)
☑ 1	Greater Than with 3 inputs (D)	GT3D	3	if (x+y) > z, then (x+y), else (x-z)
☑ 1	Less Or Equal To with 3 inputs (D)	LOE3D	3	if (x+y) <= z, then (x+y), else (x-z)

REFERENCES

Asbour, A.F., L.F. Alvarez and V.V. Toropov, 2003. Empirical modelling of shear strength of RC deep beams by genetic programming. Comput. Struct., 81: 331-338.

Back, T., D.B. Fogel and T. Michalewicz, 2000a. Evolutionary Computation 1: Basic Algorithms and Operators. Taylor and Francis, London, France, ISBN: 978-0-7503-0664-5, pp: 378.

Back, T., D.B. Fogel and T. Michalewicz, 2000b. Evolutionary Computation 2: Advanced Algorithms and Operators. Taylor and Francis Group, London, France, ISBN: 0750306653, pp: 270.

Banzhaf, W., P. Nordin, R.E. Keller and F.D. Francone, 1998. Genetic Programming: An Introduction on the Automatic Evolution of Computer Programs and its Applications. Dpunkit-verlag, New York, ISBN-10: 3920993586, pp: 470.

Barrientos, B., D. Moreno, C.P. Lopez and F.M. Santoyo, 2005. Modal vibration analysis of a metal plate by using a laser vibrometer and the POD method. J. Opt. A: Pure Applied Opt., 7: S356-S363.

Hiroaki, E., M. Koichi and O. Saijo, 2001. Practical natural frequency analysis of elastic plate on water Nihon University, Chiba, Japan. Proceedings of the 11th International Offshore and Polar Engineering Conference, June 17-22, Stavanger, Norway, pp: 98-205.

Jahed, H., M.R. Noban and M.A. Eshraghi, 2003. Ansys Finite Element. University of Tehran Press, Tehran, ISBN: 964-03-9929-9.

Koza, J.R., 1990. Genetic programming: A paradigm for genetically breeding populations of computer programs to Solve problems. Stanford University Computer Science Department Technical Report STAN-CS-90-1314.

Koza, J.R., 1992. Genetic Programming: On the Programming of Computers by Means of Nature Selection. MIT Press, Cambridge, MA., ISBN: 0-262-11170-5.

Koza, J.R., 1994a. Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge, ISBN-13: 9-780-262-11189-8.

Koza, J.R., 1994b. Introduction to Genetic Programming. In: Advances in Genetic Programming, Kinnear, K.J. (Ed.). The MIT Press, Cambridge, MA., ISBN: 0-262-11188-8, pp: 21–42.

Mitchell, M., 1996. An Introduction to Genetic Algorithms. 1st Edn., Massachusetts Institute of Technology, A Bradford Book, Cambridge, Boston, ISBN: 0262133164.

Ong, C.S., J.J. Huang and G.H. Tzeng, 2005. Building credit scoring models using genetic programming. Expert Syst. Appl., 29: 41-47.

Shatan, I.N., 2009. Application of high frequency barrier discharge for flat plate drag reduction. Proceeding of the Heat and Mass Transfer Institute of National Academy of Science of Belarus, May. 28-29, Kaunas, Lithuania, pp: 1-1.

Ziaie, A., I. Mahmoudi and A. Kyioumarsi, 2008. Using neural network in plate frequency calculation. Int. J. Mathematics Comput. Simulation, 2: 179-185.