National Conference with International Participation
ENGINEERING MECHANICS 2007
Svratka, Czech Republic, May 14 – 17, 2007

MULTI-OBJECTIVE IDENTIFICATION OF MATERIAL PARAMETERS

M. Lepš*

Summary: *The problem of an inverse analysis occurs in many engineering tasks. The task itself has several different forms and can be solved by a variety of methods. In this paper, we present a multi-objective optimization methodology to tackle the identification problem. Practical aspects will be shown on a microplane material model calibration, where parameters of highly non-linear material model are searched.*

1. Introduction

Concrete is one of the most frequently used materials in Civil Engineering. Nevertheless, as a highly heterogeneous system, it shows very complex non-linear behavior, which is extremely difficult to describe by a sound constitutive law. As a consequence, a numerical simulation of complex concrete structures still remains a very challenging and demanding topic in engineering computational modeling.

One of the most promising approaches to modeling of concrete behavior is based on the microplane concept [1]. When applied to concrete, it leads to a fully three-dimensional material law that incorporates tensional and compressive softening, damage of the material, supports different combinations of loading, unloading and cyclic loading along with the development of damage-induced anisotropy of the material. As a result, this material model is fully capable of predicting the behavior of real-world concrete structures once provided with proper input data [2]. The major disadvantages of this model are, however, a large number of phenomenological material parameters and a high computational cost associated with structural analysis even in a parallel environment. Although the authors of the model proposed a heuristic calibration procedure, it is based on the trial-and-error method and provides only a crude guide for the determination of the selected material parameters.

This year, a procedure based on artificial neural networks (ANN's) [3] for the microplane parameter identification that is able to identify reliably all microplane parameters was developed. In particular, an artificial neural network was used to estimate required parameters. As the training procedure, the genetic algorithm-based method was used.

However, the drawback of this methodology is a high computational cost of the identification algorithm. A suite of 30 uniaxial compression tests consumes approximately 25

* Ing. Matěj Lepš, Ph.D.: Katedra mechaniky, Fakulta stavební, České Vysoké Učení Technické v Praze; Thákurova 7; 166 29 Praha 6 – Dejvice; tel: +420 224 354 369, fax: +420 224 310 775;
e-mail: leps@cml.fsv.cvut.cz

days on a single processor PC with the Pentium IV 3400 MHz processor and 3 GB RAM. If we run tests in parallel on 7 computers, the needed time is less than 4 days. The hydrostatic and triaxial compression tests are less demanding, when running in parallel on 7 computers, the required time is less than one day for each test.

In this contribution, the problem is solved by implementing a parallel multi-objective procedure. The numerical analysis is implemented using the OOFEM - free finite element code with object oriented architecture [4]. The optimization procedure utilizes the Global parallel model [5]. More specifically, the program is divided into an optimization and an analysis part and in this way it is implemented in the cluster of PCs. As an optimization algorithm, the method called SADE [6] is used. Management of several objectives is utilized by the Average Ranking procedure [7]. As objectives, errors among experimental and computed stress-strain curves on a uniaxial compression test are used.

2. Forward mode of an inverse analysis

The forward mode of an inverse analysis can be formulated based on the existence of an experiment $\boldsymbol{E}$, which, physically or virtually, connects the known inputs (parameters) $\boldsymbol{X}^{\boldsymbol{E}}$ to the desired outputs (measurements) $\boldsymbol{Y}^{\boldsymbol{E}}$. Formally, this can be written as

$$\boldsymbol{Y}^{\boldsymbol{E}} = \boldsymbol{E}(\boldsymbol{X}^{\boldsymbol{E}}) \,. \tag{1}$$

Then, the problem of an inverse analysis is defined as a search for unknown inputs $\boldsymbol{X}^{\boldsymbol{E}}$ from the known outputs $\boldsymbol{Y}^{\boldsymbol{E}}$, i.e. *inversely* to the experiment $\boldsymbol{E}$. In common engineering applications, the experiment $\boldsymbol{E}$ is usually simulated by some virtual model $\boldsymbol{M}$. Often, the model is a program based on numerical methods such as the finite element method. Here, this work assumes that the model $\boldsymbol{M}$ is sufficiently precise to replace the experiment $\boldsymbol{E}$, and thus, we can put

$$\boldsymbol{E} \equiv \boldsymbol{M} \,. \tag{2}$$

This automatically results to

$$\boldsymbol{Y}^{\boldsymbol{E}} = \boldsymbol{M}(\boldsymbol{X}^{\boldsymbol{E}}) \,. \tag{3}$$

This step is important from the economy point of view, where the cost of the evaluation of the model $\boldsymbol{M}$ is assumed to be by an order of magnitude smaller than the cost of the physical experiment $\boldsymbol{E}$.

Based on the above-mentioned statements, **the forward** (classical) mode/direction of an inverse analysis is defined as a minimization of an error function of a difference between the outputs of the model and the output of the experiment, i.e.

$$\mathbf{min}\ \boldsymbol{f}(\boldsymbol{X}) = ||\boldsymbol{Y}^{\boldsymbol{E}} - \boldsymbol{M}(\boldsymbol{X})|| \,. \tag{4}$$

A solution $\boldsymbol{X}^*$ comes with the minimum of this function, where $\boldsymbol{f}(\boldsymbol{X}^*) = 0$ as well as $\boldsymbol{X}^* \equiv \boldsymbol{X}^{\boldsymbol{E}}$.

The problem (4) has been classically solved by gradient-based optimization methods. Nowadays, the model $\boldsymbol{M}$ is usually hidden in a program which is limited by license conditions, compact code etc. and therefore, the knowledge of derivatives is missing even if

the function is differentiable. Hence, the soft-computing methods can be successfully applied here. Methods like *the simulated annealing method* [8,9] with one solution in time or *evolutionary algorithms* [10,11] with a "population" of solutions are usually used.

The main advantage of this approach is that the forward mode is general in all possible aspects and is able to find an appropriate solution if such exists. This statement is confirmed with special cases like

a) A problem of a same value of outputs (***Y***) for different inputs (***X***), i.e. existence of several global optima. This case leads to a multi-modal optimization [12] but is solvable by an appropriate modification of an optimization algorithm.

b) There are different outputs (***Y***) for one input (***X***). This is the case of stochastic and probability calculations as well as experiments burdened with a noise or an experimental error. This obstacle can be tackled e.g. by introduction of stochastic parameters for outputs.

c) There is more than one experiment for one material.

The novelty within this contribution is the solution of the c) statement above which utilizes a multi-objective optimization algorithm, see Section 3.

The biggest disadvantage of the forward mode is the need for a huge number of error function evaluations. This problem can be managed by parallel decomposition and parallel implementation. The parallel decomposition is based on an idea of the so-called *implicit parallelism*, i.e. the independence of any two solutions ***X***. This is utilized by *a global parallel model* [5], where the main (master, root) processor/computer controls the optimization process while the slave processors compute the expensive evaluations of the model ***M***. Thanks to independency of solutions, nearly linear speed-up is reached until a high number of processors.

3. Optimization algorithm

To obtain more reliable results, the applicability of the global optimization algorithm for the parameter identification problem is examined in the present work. Particularly, the algorithm SADE developed at FCE, CTU in Prague, is used. In this section, each of the steps of the introduced algorithm is described in more detail. For further information, we refer an interested reader to work [6] or to Internet page [13]. The final algorithm has the following form:

```
void SADE ( void )
{
   FIRST_GENERATION ();
   while ( to_continue() )
   {
      MUTATE ();
      LOCAL_MUTATE ();
      CROSS ();
      EVALUATE_GENERATION ();
      SELECT () ;
   }
}
```

Note that in the following text, an attention is paid to constants the algorithm is working with since they have a cardinal impact on the algorithm performance. It should be emphasized that their optimal setting is unfortunately problem-dependent and presents additional non-trivial task that must be dealt with.

3.1 FIRST_GENERATION

In this function, called at the beginning of the process, the population of individuals, or vectors, whose coordinates are random numbers within given margins, is generated. There is only a single constant in this function that defines how large population of the individuals is to be generated. Based on some test problems, it was concluded that it is convenient to take this number linearly dependent on the number of variables of a solved function. In particular, to get the size of the population, the parameter `pool_rate` is used to multiply the number of optimized variables. An essential property of this parameter is the ability to slow down the convergence process, which prevents the algorithm to fall into a local extreme.

3.2 MUTATE

Within the function MUTATE, a certain number of new individuals is created by mutation of the current population. The procedure is as follows: an individual 'A' is randomly chosen from the population, further a new individual 'B' is created with all coordinates as random numbers within given margins. A new individual, which is to be added to the population, is created by translation of the vector 'A' by a random fraction of the distance 'AB' in the direction of the vector 'B'. For such a form of the mutation operator, a new constant describing the number of individuals to be created by this operator is introduced. This constant, called `radiation`, should range from 0 to 30% of the number of individuals at the beginning of the cycle. Often it is chosen to be equal to 10%. The bigger the constant is, the larger scatter of individuals in the population occurs and the convergence slows down. Attention must be paid to the size of this parameter, because for some values the algorithm failed to converge at all.

3.3 LOCAL_MUTATE

The goal of this operator is to increase algorithm's performance for problems where a solution with a high precision is sought. It again introduces a certain number of new individuals in the population by translating all coordinates of a randomly chosen vector from the population by a random small distance. The translation is chosen for each coordinate separately as the fraction of the range for the given variable with the fraction randomly chosen from the interval -0.0025 to 0.0025.

As in the case of the previously described operator MUTATE, it is again necessary to define the number of individuals to be created by local mutation. The constant called localradiation defines this number as the percentage of the total number of individuals in the population at the beginning of the cycle. The constant is set within the range 0 to 30 %, typically 10%.

3.4 CROSS

This operator creates new individuals so that, at the end of one generation, their total number is twice the number of individuals at the beginning of the cycle. The new individual is

created on the principle of differential crossing. Three individuals A, B, C are randomly chosen from the population. New individual D is created according to the following relation,

$$D = \mathrm{A} + \texttt{cross\_rate} \bullet (B - C). \qquad (5)$$

The parameter `cross_rate` has the largest influence on the convergence of the algorithm. Its value is usually taken from the range 0.1 to 0.5. The bigger is its value, the slower is convergence of the algorithm.

3.5 EVALUATE_GENERATION

The function EVALUATE_GENERATION evaluates all new individuals. Firstly, it is necessary to define the manner how to treat the individuals (vectors), which fall outside the user-defined bounds. If the optimized function is defined beyond these margins, it is sometimes advantageous not to penalize these individuals. Such behavior enables to find new solutions if the margins are not well-known in advance. In the opposite case, where the solution outside the margins is not feasible, a suitable form of penalization should be used. The SADE algorithm uses in this case the so-called 'boundary return', which means that the coordinate beyond the margins is replaced with the boundary value.

3.6 SELECT

This operator reduces the number of individuals to one half, i.e. the same number as it was at the very beginning of the cycle. This is done on the basis of natural selection; in particular, the inverse tournament selection is performed: from two randomly selected individuals the worse is disqualified from the population. This selection is random and no additional constant is needed.

3.7 Parameters settings

As a whole, only four constants are defined in the SADE algorithm. For the reported optimization problem, these parameters were set as follows,

```
pool_rate = 2
radiation = 2%
local_radiation = 2%
cross_rate = 0.5
```

3.8 Multi-objectivization

Management of several objectives is utilized by the Average Ranking (AR) procedure [7]. The advantage of the AR is that it can be applied to any population-based optimization method. As proposed by authors in [7]: “The separate fitnesses of every solution are extracted into a list of fitness values for each objective. These lists are then individually sorted into order of fitness, resulting in a set of different ranking positions for every solution for each objective. The average rank of each solution is then calculated, with this value allowing the solutions to be sorted into order of best average rank. Thus, the higher an average rank a solution has, the greater its chance of producing more offspring. Since all objective fitnesses are treated

separately, this method is range-independent". Moreover, the AR method is known to perform well for correlated objective functions, which is exactly the case of our identification procedure, where the utopia point (all errors equal to zero) exists.

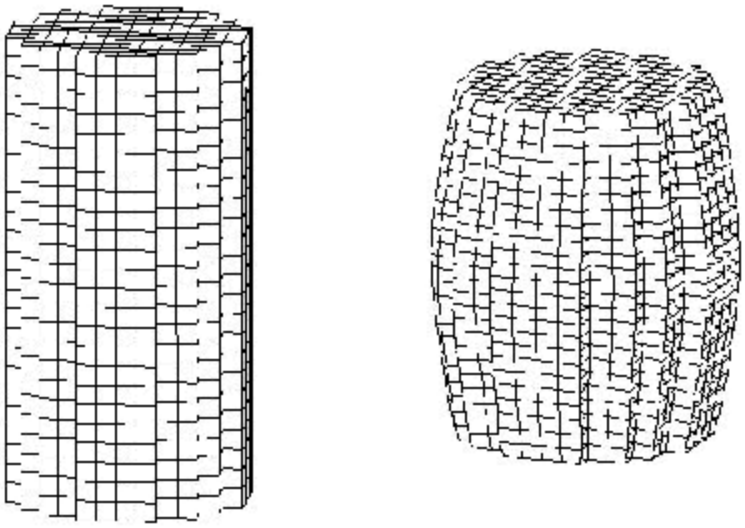

Figure 1: The computational model of a uniaxial compression test a) at the start and b) at the end of loading.

4. Identification of parameters for microplane material model

In the adopted microplane M4 law, a certain type of concrete is described by eight parameters: Young's modulus E, Poisson's ratio ν, and other six parameters (k_1, k_2, k_3, k_4, c_3, c_{20}), which do not have a simple physical interpretation, and therefore it is difficult to determine their values from experiments. The common practice for an experimenter is to employ a trial and error method to tune stress-strain diagrams by varying the model parameters [2, 14]. This is not trivial task because of highly non-linear behavior, however, several limits can be found in the literature for these parameters. In the current implementation, the appropriate boundaries were set to values shown in Table 1.

Table 1: Bounds of microplane parameters

E	$\in$	(20.0, 50.0) GPa
ν	$\in$	(0.1,0.3)
k_1	$\in$	(0.001,0.00001)
k_2	$\in$	(100.0,1000.0)
k_3	$\in$	(5.0,15.0)
k_4	$\in$	(30.0,200.0)
c_3	$\in$	(3.0,5.0)
c_{20}	$\in$	(0.2,5.0).

5. Multi-objective formulation

To define the problem more formally, the optimization goal is to find microplane parameters from the stress-strain diagram of a test specimen in a uniaxial compression, see Figures 1 and 3. In the case of a uniaxial compression test, the global response of a specimen represented by the stress-strain (σ-ε) diagram for the structure cannot be simply related to two-stage material response with an elastic-linear and non-linear (hardening and softening) part (see Figure 3). Nevertheless, we assume that it will be still possible to employ the two-stage approach, where the solution process will be divided into the optimization of elastic and non-linear parameters in the sequential way. Each step is described in detail in the rest of this section. Due to lack of experimental data, a reference simulation with parameters shown in Table 3 will be used to provide the target data.

5.1 Identification of elastic parameters

In the elastic range, Young's modulus and Poisson's ratio are determined using very short simulations describing only the elastic response of a specimen represented by the linear part of a stress-strain diagram. To identify both elastic parameters this information needs to be supplemented with both axial as well as lateral deformations. At this point, the error functions for the differences between reference and searched material can be found:

$$F_j = \sum \left(\frac{h_{i,ref} - h_i}{h_{i,ref}} \right)^2 , \qquad (6)$$

where *i* is the number of steps within elastic regime. This formulation enables to define two objective functions – F_1 for axial and F_2 for lateral case, respectively.

The analysis presented in the elastic identification part is based on the statistics of 50 independent multi-objective optimization processes, executed for both objective functions. The termination criteria were set to: first, the number of objective function evaluations exceeded 1000; second, the values of both objective functions smaller than a stopping precision 10^{-3} was found. A particular optimization process was marked as 'successful', when the latter termination condition was met. Note that since the reference simulations instead of experimental data are used, the optimum for every objective function is equal to zero. Results of this study are summarized in Table 2 showing the minimum and maximum values found. Note that the success rate was equal to 92%. The sketchy history of one evaluation in two-objective case is presented in Figure 2.

Table 2: Results of elastic identification part

Parameters	E [Mpa]	ν
Reference	32036	0,179
Found min	31806	0,177
Found max	32279	0,180

5.2 Identification of non-linear parameters

Once we have successfully determined Young's modulus and Poisson's ratio, we can continue towards the estimate of the k_1 and c_{20} parameters (Other parameters cannot be reliably identified from the uniaxial compression test [3]). The k_1 parameter is related to the peak coordinates of the stress-strain diagram. The c_{20} parameter then governs the slope of the diagram in the softening regime. In our particular case, the slope D is approximated as a secant determined from two distinct points – the peak and the end of the stress-strain diagram. This leads to the definition of the next two objective functions:

$$F_3 = \left(\frac{h_{peak,ref} - h_{peak}}{h_{peak,ref}} \right)^2, \tag{7}$$

$$F_4 = \left(\frac{D_{ref} - D}{D_{ref}} \right)^2. \tag{8}$$

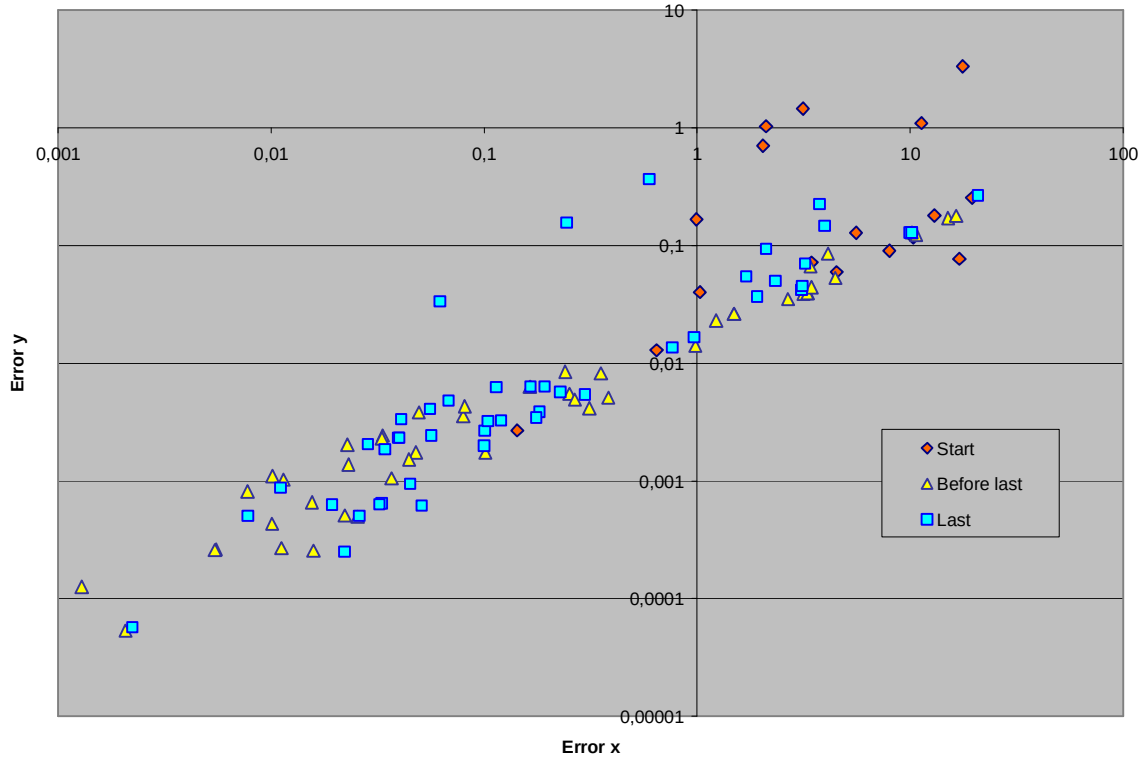

Figure 2: Evolution of errors among population of solutions during elastic identification part.

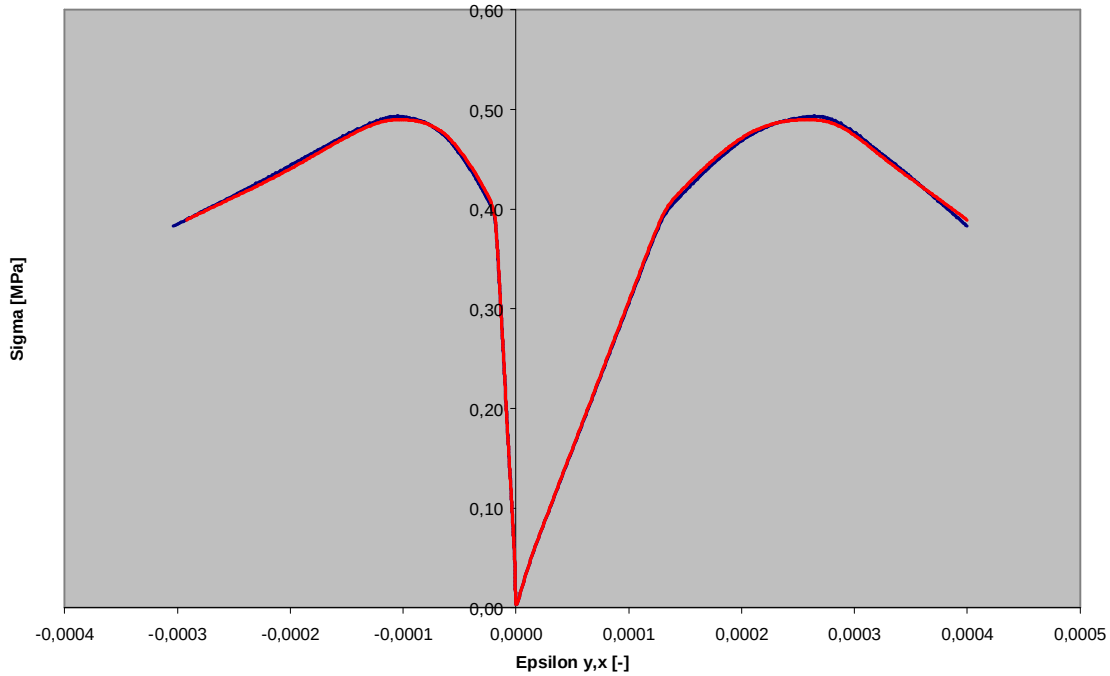

Figure 3: Comparison of load-deflection diagrams: lateral (left) and axial (right) measurements.

Due to computational overheads (more than 14 hours of computation on 8 processors), only one solution is presented, see Table 3. The results show that the errors for the elastic parameters E and ν, and moreover, for parameter k_1, are less than 1%, which is more than sufficient from the practical point of view. This is also documented by Figure 3, where no deviation of the reference and resulting curves is visible in the elastic range. The prediction of the parameter c_{20} shows similar precision, as is shown in Figure and Table 3. Note that the "relative error 1" stems for error according to the reference value, whereas the "relative error 2" stems for difference between parameters divided by the free range of the parameter, i.e. the difference between allowable maximum and minimum of the selected parameter.

Table 3: Results of non-linear identification part

Parameters	E [Mpa]	ν	k1	k2	k3	k4	c3	c20
Found	32220,3	0,180276	0,000106	742,079	7,61354	181,207	4,8279	0,214178
Reference	32035,5	0,178759	0,000107	452,5	7,4167	48,417	4,4167	0,24
Rel. error 1	0,57%	0,84%	0,73%	39,02%	2,59%	73,28%	8,52%	12,06%
Rel. error 2	0,62%	0,76%	0,08%	32,18%	1,97%	78,11%	20,56%	0,54%

7. Conclusions

We have proposed a sound identification procedure for material parameters of the constitutive model for concrete. The most pertinent conclusions can be stated as follows:

i) The sequential identification approach for the uniaxial compression test leads to a sufficient identification of four parameters. Each of two stages uses only a part of the test simulation, which leads to substantial computational time savings.

ii) As the result of a multi-objective identification procedure, the errors in identified parameters do not accumulate. Therefore, the values were identified with higher accuracy then those achieved by neural network-based inverse analysis [3].

iii) The predictive outcome of the present study is the conclusion that the determination of all microplane model parameters needs more test cases rather than a sole uniaxial compression test, which is consistent with findings in [3].

Finally, the parallel solution appears to be an appropriate tool how to tackle with enormous computational demand of the microplane material model. Obtained nearly-linear speedup together with possibility to use much more processors promise new interesting results and potential applications of the presented method in the future.

8. Acknowledgement

The financial support of this work by the research project GAČR 103/07/P554 is gratefully acknowledged.

9. References

[1] Z.P. Bažant, F.C. Caner, I. Carol, M.D. Adley, S.A. Akers, Microplane model M4 for concrete. Part I: Formulation with work-conjugate deviatoric stress, Part II: Algorithm and calibration, *Journal of Engineering Mechanics-ASCE*, 126, (2000), 944-953, 954–961.

[2] J. Němeček, B. Patzák, D. Rypl, Z. Bittnar, Microplane models: Computational aspects and proposed parallel algorithm, *Computers and Structures*, 80, (2002), 2099–2108.

[3] A. Kučerová, M. Lepš, J. Zeman, Back Analysis of Microplane Model Parameters Using Soft Computing Methods, *Computer Assisted Mechanics in Engineering Sciences*, 14(2), 2007, accepted to publication.

[4] B. Patzak and Z. Bittnar. Design of object oriented finite element code. *Advances in Engineering Software*, 32(10-11):759-767, 2001. Web page: http://www.oofem.org

[5] E. Cantú-Paz, *Efficient and Accurate Parallel Genetic Algorithms*, Kluwer Academic Publishers, 2001.

[6] O. Hrstka, A. Kučerová, Improvements of real coded genetic algorithms based on differential operators preventing the premature convergence, *Advances in Engineering Software*, 35, (2004), 237–246.

[7] P. J. Bentley and J. P. Wakefield, *An Analysis of Multiobjective optimization within Genetic Algorithms*, Technical Report ENGPJB96, University of Huddersfield, UK, 1996.

[8] L. Ingber: Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling*, (1993), 18(11):29-57.

[9] R.V.V. Vidal, editor: *Applied Simulated Annealing*, volume 396 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, (1993).

[10] D.E. Goldberg: Genetic algorithms in search, optimization and machine learning, Addison-Wesley, (1989).

[11] Z. Michalewicz: Genetic Algorithms+Data Structures=Evolution Programs, Springer, (1992).

[12] S.W. Mahfoud: Niching methods for genetic algorithms. *PhD thesis*, University of Illinois at Urbana-Champaign, Urbana, IL, USA, (1995).

[13] Homepage of SADE: http://klobouk.fsv.cvut.cz/~ondra/sade/sade.htm.

[14] M. E. Jirasek and Z. P. Bazant, *Inelastic Analysis of Structures*, John Wiley & Sons, 2001.