Boosted surrogate models in evolutionary optimization*

Martin Holeňa

Institute of Computer Science, Academy of Sciences of the Czech Republic,
Pod Vodárenskou věží 2, 18207 Praha 8, Czech Republic, martin@cs.cas.cz, web: cs.cas.cz/~martin

Abstract. *The paper deals with surrogate modelling, a modern approach to the optimization of empirical objective functions. The approach leads to a substantial decrease of time and costs of evaluation of the objective function, a property that is particularly attractive in evolutionary optimization. In the paper, an extension of surrogate modelling with regression boosting is proposed. Such an extension increases the accuracy of surrogate models, thus also the agreement between results of surrogate modelling and results of the intended optimization of the original objective function. The proposed extension is illustrated on a case study in the area of searching catalytic materials optimal with respect to their behaviour in a particular chemical reaction. A genetic algorithm developed specifically for this application area is employed for optimization, multilayer perceptrons serve as surrogate models, and a method called AdaBoost.R2 is used for boosting. Results of the case study clearly confirm the usefulness of boosting for surrogate modelling.*

1 Introduction

For more than two decades, *evolutionary algorithms*, especially their most frequently encountered representative – *genetic algorithms*, belong to the most successful methods for solving difficult optimization tasks [3, 11, 31, 32, 42]. The popularity of evolutionary algorithms is to some extent due to their biological inspiration, which increases their comprehensibility outside computer science. Nevertheless, they share several purely mathematical properties of all stochastic optimization methods, most importantly, the valuable ability of to escape a local optimum and continue the search for a global one, and the restriction of the information on which they rely to function values only. Consequently, they do not need information about gradients or second-order partial derivatives, differently to smooth optimization methods (such as steepest descent, conjugate gradient methods, the popular Levenberg-Marquardt method, etc.). This makes them particularly attractive for the optimization of *empirical objective functions*, the values of which cannot be analytically computed, but have to be obtained experimentally, through some measurement or testing. Indeed, the impossibility to compute analytically the function values of such a function makes also an analytical computation of its gradient and second-order derivatives impossible, whereas measurement errors usually hinder obtaining sufficiently accurate estimates of the derivatives.

Like other methods relying solely on function values, evolutionary algorithms need the objective function to be evaluated in quite a large number of points. In the context of optimization of empirical objective functions, this can be quite disadvantageous because the evaluation of such a function in the points forming one generation of an evolutionary algorithm is often costly and time-consuming. Hence, the above mentioned advantages of using evolutionary algorithms for the optimization of empirical objective functions are frequently counterbalanced by considerably high costs and time needed for the evaluation of such functions. An area, where the trade-off between successful optimization and costly objective function evaluations plays a crucial role, is the computer-aided search for new materials and chemicals optimal with respect to certain properties [2]. Here, evolutionary algorithms are used in more than 90 % of optimization tasks, and the rarely encountered alternatives are simulated annealing [9, 22, 23], simplex method [17], and holographic search strategy [37, 38, 41], which also use solely function values, therefore needing a similarly high number of objective function evaluations as evolutionary algorithms. Testing a generation of materials or chemicals typically needs hours to days of time and costs hundreds to thousands euros. Therefore, the evolutionary optimization rarely runs for more than ten generations.

The usual approach to decreasing the cost and time of optimization of empirical objective functions is to evaluate the objective function only sometimes and to evaluate a suitable regression model of that function otherwise. The employed model is termed *surrogate model* of the empirical objective function, and the approach is referred to as *surrogate modelling*. Needless to say, the time and costs needed to evaluate a regression model are negligible compared to an empirical objective function. However, it must not be forgotten that the final optimized function coincides with the original empirical objective function only in some points, whereas in the remaining points it coin-

* The research reported in this paper has been supported by the grant No. 201/08/1744 of the Grant Agency of the Czech Republic and partially supported by the Institutional Research Plan AV0Z10300504.

cides only with its surrogate model. Consequently, the agreement between the results of surrogate modelling and the results of the intended optimization of the original objective function depends on the accuracy of the approximation of the original objective function by the surrogate model.

This paper suggests to increase the accuracy of surrogate models by means of boosting. Boosting is a popular approach to increasing the accuracy of classification, and due to the success of classification boosting, also several methods of regression boosting have already been proposed. However, so far no attempt has been reported to combine regression boosting with surrogate modelling. Hence, the purpose of the research reported in the paper is basically a proof of concept: to extend surrogate modelling through the incorporation of regression boosting, and to validate that extension on several sufficiently complex case studies. One of those case studies is described in the paper.

In the following section, basic principles of surrogate modelling and its strategies in evolutionary optimization are recalled, and important surrogate models are listed. Section 3 recalls the principles of boosting and explains a particular method of regression boosting that will be employed later in a case study in materials science. That case study is sketched and its main results are presented in Section 4.

2 Surrogate modelling

Surrogate modelling is a general approach to the optimization of costly objective functions in which the evaluation of the objective function is restricted to points that are considered to be most important for the progress of the employed optimization method [5, 25, 27, 30, 39, 40]. It is most frequently encountered in connection with the optimization of empirical objective functions, but has been equally successfully applied also to expensive optimization tasks in engineering design in which the objective function is not empirical, but its evaluation is connected with intensive computations [25]. In the context of computer-aided search for new materials and chemicals optimal with respect to certain properties, surrogate modelling can be viewed as replacing real experiments with simulated virtual experiments in a computer: such virtual experiments are sometimes referred to as *virtual screening* [2].

Although surrogate modelling is a general optimization approach (cf. its application in the context of conventional optimization in [5]), it is most frequently encountered in connection with evolutionary algorithms. The reason is that in evolutionary optimization, the approach leads to the approximation of the landscape of the fitness function, i.e., to a method that is known to be useful in general [19, 20, 29]. In evolutionary algorithms, most important for the progress of the method are on the one hand points that best indicate the global optimum (typically through highest values of the fitness function), on the other hand points that most contribute to the diversity of the population.

In the context of evolutionary optimization, surrogate modelling has the following main steps:

(i) Collecting an initial set of points in which the objective function has already been empirically evaluated. This can be the first generation or several first generations of the evolutionary algorithm, but such points are frequently available in advance.

(ii) Approximating the objective function by a surrogate model, with the use of the set of all points in which it has been empirically evaluated.

(iii) Running the evolutionary algorithm for a population considerably larger than is the desired population size, with the empirical objective function replaced by the surrogate model.

(iv) Forming the next generation of the desired size as a subset of the large population obtained in the preceding step that includes points most important according to considered criteria for the progress of optimization (such as indication of global optimum, diversity).

(v) Empirically evaluating the objective function in all points that belong to the next generation of the desired size, and returning to step (ii).

Actually, the above steps (ii)–(v) correspond to only one possible strategy of surrogate modelling in evolutionary optimization: the *individual-based control*, sometimes also referred to as *pre-selection* [40]. An alternative strategy to the steps (ii)–(v) is to run the algorithm for only the desired population size, interleaving one generation/several generations in which the original objective function is empirically evaluated with a certain number of generations in which the surrogate model is evaluated. This is the *generation-based control* of surrogate modelling in evolutionary optimization.

For empirical objective functions, it is typical to be highly nonlinear. Therefore, nonlinear regression models should be used as surrogate models. They can be basically divided into two large groups according to whether the set of functions among which the surrogate model has to be chosen has an explicit finite parametrization.

1. So far, mostly *parametric models* have been used for surrogate modelling. From the point of view of their role in this context and/or their overall importance, the following kinds of parametric nonlinear regression models are most worth mentioning:
   (i) *Multilayer feed-forward neural networks*, more precisely, the nonlinear mappings computed

by such networks. Their attractiveness for nonlinear regression in general and for surrogate modelling in particular [20] is due to their *universal approximation capability*, which actually means that linear spaces of functions computed by certain families of multilayer feed-forward neural networks are dense in some general function spaces [18, 21, 26]. For example, considering the most common representative of such networks – multilayer perceptrons, the linear space formed by all functions computed by the family of perceptrons with one hidden layer and infinitely smooth activation functions is dense in the space $L_p(\mu)$ of functions with the p-th power of absolute value finitely integrable with respect to a finite measure μ, in the space $C(X)$ of functions continuous on a compact X, and in Sobolev spaces generalizing $L_p(\mu)$ to functions that are differentiable up to a given order. In the application domain of catalytic materials, from which the case study presented in Section 4 is taken, nearly all examples of regression analysis published since mid 1990s rely on multilayer feed-forward neural networks, typically on multilayer perceptrons (Figure 1). In the last edition of "Handbook of heterogeneous catalysis", more than 20 such examples are listed, as well as several additional, based on other kinds of such networks – radial basis function networks and piecewise-linear neural networks [16]. Therefore, these three kinds of neural networks are now briefly recalled:

- *Multilayer perceptrons (MLPs)* can have an arbitrary number of hidden layers, and the basis functions of their linear space of computed functions are constructed by means of sigmoidal activation functions, such as logistic sigmoid, hyperbolic tangent, or arctangent [13, 43].
- *Radial basis function (RBF) networks* always have only one hidden layer, and the basis functions of their space of computed functions are radial, i.e., the function value depends only on the distance of the vector of input values from some centre, specific to the function [7].
- *Piecewise-linear neural networks* are simply MLPs with piecewise-linear activation functions. Their linear space of computed functions is dense only in $C(X)$, but on the other hand, they allow a straightforward extraction of logical rules describing the relationships between input and output values of the network [15].

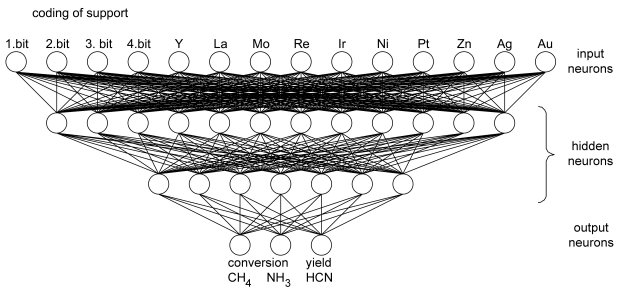

Fig. 1. Example MLP architecture with two hidden layers, used in the case study presented in Section 4.

  - **(ii)** *Support vector regression* based on positive semi-definite kernels [34, 36]. It is worth mentioning that they generalize the above recalled RBF networks, and also the historically first kind of nonlinear regression – polynomial regression.
  - **(iii)** *Gaussian process regression* [28] is listed here also due to a relationship to radial basis function networks, but most importantly due to the fact that it has already been successfully employed in surrogate modelling [6].
2. *Nonparametric regression models* are, in general, more flexible than parametric models, but the flexibility is typically paid for by more extensive computations. Therefore, their importance has been increasing only during the last two decades, following the increasing power of available computers [12, 14]. Nevertheless, there is one noteworthy exception:
  - **(v)** *Regression trees* have been successfully used already since the early 1980s [4]. They are actually a modification of a classification method, therefore the regression function is piecewise-constant. That property accounts for relatively low computational requirements of regression trees, but also decreases their flexibility, otherwise the main advantage of nonparametric methods.

3 Boosting regression models

Boosting is a method of improving classification accuracy that consists in developing the classifier iteratively, and increasing the relative influence of the training data that most contributed to errors in the previous iterations on its development in the subsequent iterations [33]. The usefulness of boosting for classification has incited its extension to regression [8]. Both for classification and for regression, the basic approach to increasing the relative influence of particular

training data is re-sampling the training data according to a distribution that gives them a higher probability of occurrence. This is equivalent to re-weighting the contributions of the individual training pairs (x_j, y_j), with higher weights corresponding to higher values of the error measure.

Since surrogate models are regression models, any method for regression boosting (such as [8, 10, 35]) is suitable for them. In the following, the method *AdaBoost.R2* will be explained in detail, proposed in [8].

Similarly to other adaptive boosting methods, each of the available pairs $(x_1, y_1), \ldots \ldots, (x_p, y_p)$ of input and output data is in the first iteration of AdaBoost.R2 used exactly once. This corresponds to re-sampling them according to the uniform probability distribution P_1 with $P_1(x_1) = \frac{1}{p}$ for $j = 1, \ldots, p$. In addition, the weighted average error of the 1st iteration is set to zero, $\bar{E}_1 = 0$.

In the subsequent iterations ($i \geq 2$), the following sequence of steps is performed:

1. A sample $(\xi_1, \eta_1), \ldots, (\xi_p, \eta_p)$ is obtained through re-sampling $(x_1, y_1), \ldots, (x_p, y_p)$ according to the distribution P_{i-1}.
2. Using $(\xi_1, \eta_1), \ldots, (\xi_p, \eta_p)$ as training data, a regression model F_i is constructed.
3. A [0,1]-valued squared error vector E_i of F_i with respect to $(x_1, y_1), \ldots, (x_p, y_p)$ is calculated as

$$E_i = (E_i(1), \ldots, E_i(p)) = \\ = \frac{((F_i(x_1) - y_1)^2, \ldots, (F_i(x_p) - y_p)^2)}{\max_{k=1,\ldots,p}(F_i(x_k) - y_k)^2}. \quad (1)$$

4. The weighted average error of the i-th iteration is calculated as

$$\bar{E}_i = \frac{1}{p} \sum_{k=1}^{p} P_i(x_k, y_k) E_i(k). \quad (2)$$

5. Provided $\bar{E}_i < 0.5$, the probability distribution for re-sampling $(x_1, y_1), \ldots, (x_p, y_p)$ is for $k = 1, \ldots, p$ updated according to

$$P_i(x_k, y_k) = \\ = \frac{P_{i-1}(x_k, y_k) \left(\frac{\bar{E}_i}{1-\bar{E}_i}\right)^{(1-E_i(k))}}{\sum_{i=1}^{p} P_{i-1}(x_k, y_k) \left(\frac{\bar{E}_i}{1-\bar{E}_i}\right)^{(1-E_i(k))}}. \quad (3)$$

6. The *boosting approximation* in the i-th iteration is set to the median of the approximations $F_1, \ldots, F_i$ with respect to the probability distribution

$$\left(\frac{\bar{E}_1}{1-\bar{E}_1}, \ldots, \frac{\bar{E}_i}{1-\bar{E}_i}\right). \quad (4)$$

The errors used to asses the quality of the boosting approximation are then called *boosting errors*, e.g., *boosting MSE*, or *boosting MAE*, where MSE refers to the mean squared error between the computed and measured values, whereas MAE refers to the mean absolute error, i.e., to the mean Euclidean distance between them. For simplicity, also the approximation in the first iteration, F_1, is called boosting approximation if boosting is performed, and the respective errors are then called boosting errors, although boosting actually does not introduce any modifications in the first iteration.

The above formulation of the method deals only with the case $\bar{E}_i < 0.5$. For $\bar{E}_i \geq 0.5$, the original formulation of the method in [8] proposes to stop the boosting. However, that is not allowed if the stopping criterion should be based on an independent set of validation data. Indeed, the calculation of $\bar{E}_i$ does not rely on any such independent data set, but it relies solely on the data employed to construct the regression model. A possible alternative for the case $\bar{E}_i \geq 0.5$ is reinitialization, i.e., proceeding as in the 1st iteration [1].

In connection with using feed-forward neural networks as surrogate models, it is important to be aware of the difference between the iterations of boosting and the iterations of neural network training. Boosting iterates on a higher level, one iteration of boosting includes a complete training of an ANN, which can proceed for many hundreds of iterations. Nevertheless, both kinds of iterations are similar in the sense that starting with some iteration, over-training is present. Therefore, also over-training due to boosting can be reduced through *stopping* in the iteration after which the *error for an independent set of data first time increases*. Moreover, *cross-validation* can be used to find the iteration most appropriate for stopping.

4 Case study in materials science

The extension of surrogate modelling with boosting will now be illustrated on a case study using data from the investigation of *catalytic materials for the high-temperature synthesis of hydrocyanic acid*. That investigation and its results have been recently described in [24]. It has been performed through *high-throughput experiments* in a circular 48-channel reactor. In most of those experiments, the composition of the materials was designed by means of a *genetic algorithm* developed *specifically for heterogeneous catalysis* [44]. More precisely, the algorithm was running for 7 generations of population size 92, and in addition 52 other catalysts with manually designed composition were investigated. Consequently, data about altogether 696 catalytic materials were gathered.

The composition and preparation of the investigated catalytic materials and the conditions in which they had been tested have been in detail described in [24]. Here, only the independent and dependent variables are recalled, the latter corresponding to the considered possible objective functions:

- *independent variables*: material used as support, and proportions of the 10 metal additives Y, La, Mo, Re, Ir, Ni, Pt, Zn, Ag, Au (an 11th metal, Zr, was left out due to the fact that the proportions of all active compounds sum up to 100 %);
- *dependent variables*, i.e., objective functions: conversions of CH_4 and NH_3, and yield of HCN.

As the surrogate model, MLPs were employed, in accordance with their leading role among nonlinear regression models in the area of catalytic materials [2, 16]. Each considered neural network had *14 input neurons*: 4 of them coding the material used as support, the other 10 corresponding to the proportions of the 10 metal additives belonging to independent variables; output neurons were 3, corresponding to the possible objective functions (Figure 1).

The most appropriate MLP architectures were searched by means of cross-validation, using only data about catalysts from the 1.–6. generation of the genetic algorithm and about the 52 catalysts with manually designed composition, thus altogether data about 604 catalytic materials. Data about catalysts from the 7. generation were completely excluded and left out for validating the search results. To use as much information as possible from the available data, cross-validation was applied as the extreme 604-fold variant, i.e., leave-1-out validation. The set of architectures within which the search was performed was delimited by means of the heuristic *pyramidal condition*: the number of neurons in a subsequent layer must not increase the number of neurons in a previous layer. Denote n_I, n_h and n_O the numbers of input, hidden and output neurons, respectively, and n_{H1} and n_{H2} the numbers of neurons in the first and second hidden layer, respectively. Then the pyramidal condition reads:

(i) for MLPs with 1 hidden layer: $n_I \geq n_H \geq n_O$, in our case $14 \geq n_H \geq 3$ (12 architectures);

(ii) for MLPs with 2 hidden layers: $n_I \geq n_{H1} \geq n_{H2} \geq n_O$, in our case $14 \geq n_{H1} \geq n_{H2} \geq 3$ (78 architectures).

To investigate the usefulness of boosting in our case study, the same data were used and the same set of architectures was considered as for architecture search. In each iteration, a leave-1-out validation was performed, in the way briefly outlined in the preceding section: The mean squared error of the performance of the catalytic materials serving in the individual folds as test data was calculated, and averaged over all the 604 folds. The *criterion* according to which *boosting is considered useful* to an architecture was: the average boosting MSE in the 2nd iteration has to be lower than in the 1st iteration. The iteration till which the average boosting MSE continuously decreased was then taken as the *final iteration of boosting*.

According to that criterion, boosting was useful to 9 from the 12 considered architectures with one hidden layer and to 65 from the 78 considered architectures with two hidden layers. To validate the most promising results of the investigation of the usefulness of boosting in our case study, the data from the 7th generation of the genetic algorithm were used. The validation included the *5 architectures* that were most promising for boosting from the point of view of the *lowest boosting MSE on test data in the final iteration*. These were the architectures (14,10,6,3), (14,14,8,3), (14,13,5,3), (14,10,4,3) and (14,11,3), for which the final iterations of boosting were 32, 29, 31, 19 and 3, respectively. For each of them, the validation proceeded as follows:

1. In each iteration up to the final boosting iteration corresponding to the respective architecture, a single MLP was trained with data about the 604 catalytic materials considered during the architecture search.
2. Each of those MLPs was employed to approximate the conversions of CH_4 and NH_3 and the yield of HCN for the 92 materials from the 7. generation of the genetic algorithm.
3. In each iteration, the medians with respect to the probability distribution (4) of the approximations of the two conversions and of the HCN yield obtained up to that iteration were used as the boosting approximations.
4. From the conversions and the yield predicted by the boosting approximations, and from the measured values, the boosting MSE and MAE were calculated for each MLP.

The boosting errors (MSE and MAE) are summarized in Figure 2, whereas Figure 3 compares the boosting approximations of the conversions of CH_4 and NH_3 and of the yield of HCN in the 1st and final iteration with their measured values. The presented results clearly confirm the usefulness of boosting for the five considered architectures. For each of them, boosting led to an overall decrease of both considered error measures, the MSE and MAE, on new data from the 7th generation of the genetic algorithm. Moreover, the decrease of the MSE (which is the measure employed during the investigation of the usefulness of boosting) is uninterrupted or nearly uninterrupted till the final boosting iteration. On the other hand, the

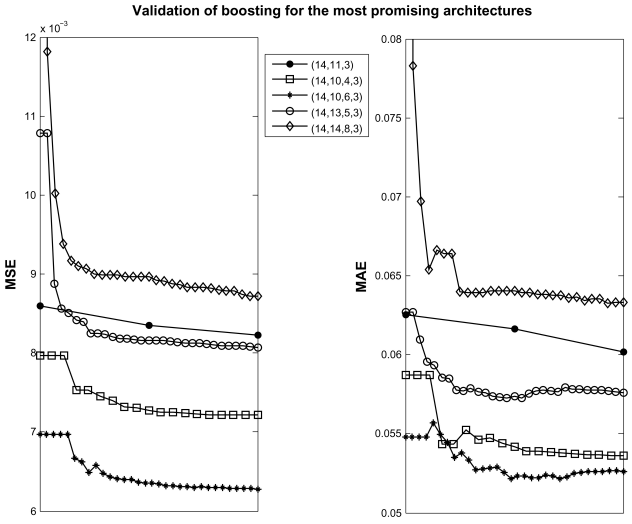

Fig. 2. History of the boosting MSE and MAE on the data from the 7th generation of the genetic algorithm for MLPs with the five architectures included in the validation of boosting.

scatter plots in Figure 3 do not indicate any apparent difference between the effect of boosting on the three properties employed as catalyst performance measures in our case study – conversion of CH_4, conversion of NH_3, and yield of HCN. Hence, the performed validation confirms the usefulness of boosting irrespectively of which of those performance measures is considered.

5 Conclusions

The paper dealt with surrogate modelling, a modern approach to the optimization of empirical objective functions, which is particularly attractive in evolutionary optimization. It proposed to extend surrogate modelling with regression boosting, to increase the accuracy of surrogate models, thus also the agreement between results of surrogate modelling and results of the intended optimization of the original objective function. Needless to say, regression boosting is not new, though it is less common than the popular classification boosting. However, novel is its combination with surrogate models, which adds the advantage of increased accuracy to the main advantage of surrogate modelling – decreasing the time and costs of optimization of empirical objective functions.

Theoretical principles of both surrogate modelling and boosting are known, therefore the main purpose of the reported research was to validate the feasibility of the proposed extension of surrogate modelling on several sufficiently complex case studies, one of which was sketched in this paper. The presented case study results clearly confirm the usefulness of boosting. For the five most promising architectures, boosting leads to an overall decrease of both considered error measures, MSE and MAE, on new data from the 7th generation of the genetic algorithm. Moreover, the decrease of the MSE (which is the boosting error employed during the investigation of the usefulness of boosting) is uninterrupted or nearly uninterrupted till the final boosting iteration. On the other hand, the scatter plots in Figure 3 do not indicate any apparent difference between the effect of boosting on the three catalyst properties considered as possible objective functions in our case study – conversion of CH_4, conversion of NH_3, and yield of HCN. Hence, the performed validation confirms the usefulness of boosting irrespectively of which of these objective functions is selected.

References

1. H. Altinçay: *Optimal resampling and classifier prototype selection in classifier ensembles using genetic algorithms.* Pattern Analysis and Applications 7, 2004, 285–295.
2. M. Baerns and M. Holeňa: *Combinatorial Development of Solid Catalytic Materials. Design of High-Throughput Experiments, Data Analysis, Data Mining.* World Scientific, Singapore, 2009.
3. T. Bartz-Beielstein: *Experimental Research in Evolutionary Computation.* Springer Verlag, Berlin, 2006.
4. L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone: *Classification and Regression Trees.* Wadsworth, Belmont, 1984.
5. A.J. Brooker, J. Dennis, P.D. Frank, D.B. Serafini, V. Torczon, and M. Trosset: *A rigorous framework for optimization by surrogates.* Structural and Multidisciplinary Optimization, 17, 1998, 1–13.
6. D. Büche, N.N. Schraudolph, and P. Koumoutsakos: *Accelerating evolutionary algorithms with gaussian process fitness function models.* IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 35, 2005, 183–194.
7. M.D. Buhmann: *Radial Basis Functions: Theory and Implementations.* Cambridge University Press, Cambridge, 2003.
8. H. Drucker: *Improving regression using boosting techniques.* In A.J.C. Sharkey, editor, Proceedings of the 14th International Conference on Machine Learning, Springer Verlag, London, 1997, 107–115.
9. A. Eftaxias, J. Font, A. Fortuny, J. Giralt, A. Fabregat, and F. Stber: *Kinetic modelling of catalytic wet air oxidation of phenol by simulated annealing.* Applied Catalysis B: Environmental 33, 2001, 175–190.
10. J. Friedman: *Greedy function approximation: A gradient boosting machine.* Annals of Statistics 29, 2001, 1189–1232.
11. D. Goldberg: *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, 1989.

12. L. Györfi, M. Kohler, A. Krzyzak, and H. Walk: *A Distribution-Free Theory of Nonparametric Regression.* Springer Verlag, Berlin, 2002.
13. M.T. Hagan, H.B. Demuth, and M.H. Beale: *Neural Network Design.* PWS Publishing, Boston, 1996.
14. T.J. Hastie and R.J. Tibshirani: *Generalized Additive Models.* Chapman & Hall, Boca Raton, 1990.
15. M. Holeňa: *Piecewise-linear neural networks and their relationship to rule extraction from data.* Neural Computation 18, 2006, 2813–2853.
16. M. Holeňa and M. Baerns: *Computer-aided strategies for catalyst development.* In G. Ertl, H. Knözinger, F. Schüth, and J. Eitkamp, editors, Handbook of Heterogeneous Catalysis, Wiley-VCH, Weinheim, 2008.
17. A. Holzwarth, P. Denton, H. Zanthoff, and C. Mirodatos: *Combinatorial approaches to heterogeneous catalysis: Strategies and perspectives for academic research.* Catalysis Today 67, 2001, 309–318.
18. K. Hornik: *Approximation capabilities of multilayer neural networks.* Neural Networks 4, 1991, 251–257.
19. Y. Jin: *A comprehensive survery of fitness approximation in evolutionary computation.* Soft Computing 9, 2005, 3–12.
20. Y. Jin, M. Hüsken, M. Olhofer, and B. Sendhoff: *Neural networks for fitness approximation in evolutionary optimization.* In Y. Jin, editor, Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 281–306.
21. P.C. Kainen, V. Kůrková, and M. Sanguineti: *Estimates of approximation rates by gaussian radial-basis functions.* In Adaptive and Natural Computing Algorithms, Springer Verlag, Berlin, 2007, 11–18.
22. B. Li, P. Sun, Q. Jin, J. Wang, and D. Ding: *A simulated annealing study of Si, Al distribution in the omega framework.* Journal of Molecular Catalysis A: Chemical 148, 1999, 189–195.
23. A.S. McLeod and L.F. Gladden: *Heterogeneous catalyst design using stochastic optimization algorithms.* m Journal of Chemical Information and Computer Science 40, 2000, 981–987.
24. S. Möhmel, N. Steinfeldt, S. Endgelschalt, M. Holeňa, S. Kolf, U. Dingerdissen, D. Wolf, R. Weber, and M. Bewersdorf: *New catalytic materials for the high-temperature synthesis of hydrocyanic acid from methane and ammonia by high-throughput approach.* Applied Catalysis A: General 334, 2008, 73–83.
25. Y.S. Ong, P.B. Nair, A.J. Keane, and K.W. Wong: *Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems.* In Y. Jin, editor, Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 307–331.
26. A. Pinkus: *Approximation theory of the MPL model in neural networks.* Acta Numerica 8, 1998, 277–283.
27. K. Rasheed, X. Ni, and S. Vattam: *Methods for using surrogate modesl to speed up genetic algorithm oprimization: Informed operators and genetic engineering.* In Y. Jin, editor, Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 103–123.
28. E. Rasmussen and C. Williams: *Gaussian Process for Machine Learning.* MIT Press, Cambridge, 2006.
29. A. Ratle: *Accelerating the convergence of evolutionary algorithms by fitness landscape approximation.* In A.E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, Parallel Problem Solving from Nature, Springer Verlag, Berlin, 1998, 87–96.
30. A. Ratle: *Kriging as a surrogate fitness landscape in evolutionary optimization.* Artificial Intelligence for Engineering Design, Analysis and Manufacturing 15, 2001, 37–49.
31. C.R. Reeves and J.E. Rowe: *Genetic Algorithms: Principles and Perspectives.* Kluwer Academic Publishers, Boston, 2003.
32. R. Schaefer: *Foundation of Global Genetic Optimization.* Springer Verlag, Berlin, 2007.
33. R. Schapire: *The strength of weak learnability.* Machine Learning 5, 1990, 197–227.
34. B. Schölkopf and A.J. Smola: *Learning with Kernels.* MIT Press, Cambridge, 2002.
35. D.L. Shrestha: *Experiments with AdaBoost.RT, an improved boosting scheme for regression.* Neural Computation 18, 2006, 1678–1710.
36. I. Steinwart and A. Christmann: *Support Vector Machines.* Springer Verlag, New York, 2008.
37. A. Tompos, J.L. Margitfalvi, E. Tfirst, L. Végvári, M.A. Jaloull, H.A. Khalfalla, and M.M. Elgarni: *Development of catalyst libraries for total oxidation of methane: A case study for combined application of "holographic research strategy and artificial neural networks" in catalyst library design.* Applied Catalysis A: General 285, 2005, 65–78.
38. A. Tompos, L. Vgvári, E. Tfirst, and J.L. Margitfalvi: *Assessment of predictive ability of artificial neural networks using holographic mapping.* Combinatorial Chemistry and High Throughput Screening 10, 2007, 121–134.
39. H. Ulmer, F. Streichert, and A. Zell: *Model-assisted steady state evolution strategies.* In GECCO 2003: Genetic and Evolutionary Computation, Springer Verlag, Berlin, 2003, 610–621.
40. H. Ulmer, F. Streichert, and A. Zell: *Model assisted evolution strategies.* In Y. Jin, editor, Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 333–355.
41. L. Végvári, A. Tompos, S. Göbölös, and J.F. Margitfalvi: *Holographic research strategy for catalyst library design: Description of a new powerful optimisation method.* Catalysis Today 81, 2003, 517–527.
42. M.D. Vose: *The Simple Genetic Algorithm. Foundations and Theory.* MIT Press, Cambridge, 1999.
43. H. White: *Artificial Neural Networks: Approximation and Learning Theory.* Blackwell Publishers, Cambridge, 1992.
44. D. Wolf, O.V. Buyevskaya, and M. Baerns: *An evolutionary approach in the combinatorial selection and optimization of catalytic materials.* Applied Catalyst A: General 200, 2000, 63–77.

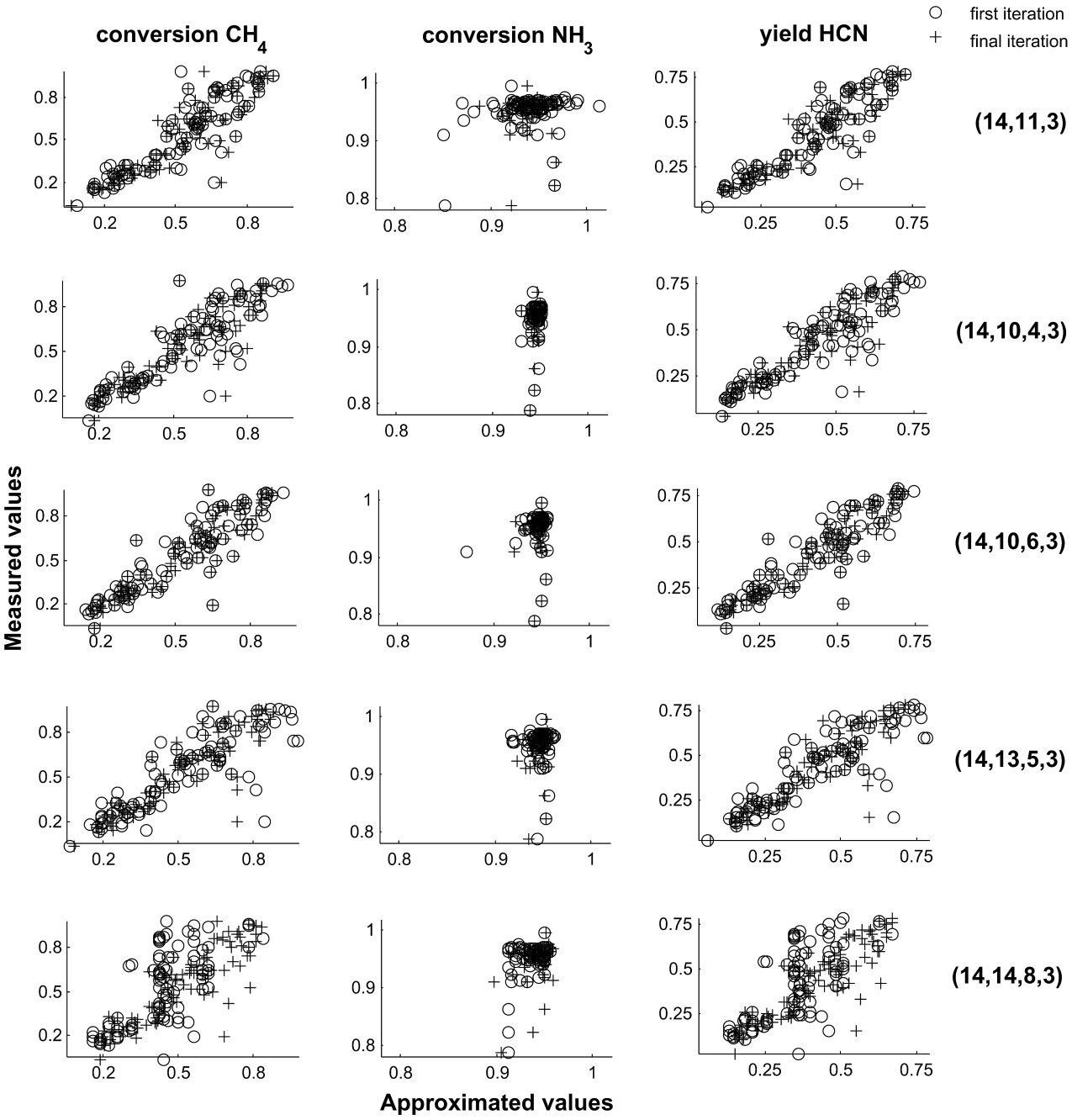

Fig. 3. Comparison of the boosting approximations of the conversions of CH_4 and NH_3 and of the yield of HCN in the 1st and final iteration with their measured values for the 92 catalytic materials from the 7th generation of the genetic algorithm.