C. R. Mecanique 347 (2019) 806–816

Contents lists available at ScienceDirect

Comptes Rendus Mecanique

www.sciencedirect.com

Data-Based Engineering Science and Technology / *Sciences et technologies de l'ingénierie basées sur les données*

An efficient Tabu-search optimized regression for data-driven modeling

Chady Ghnatios [a,*], Ré-Mi Hage [b], Ilige Hage [a]

[a] *Notre Dame University–Louaize, Department of Mechanical Engineering, Zouk Mosbeh, PO Box 72, Lebanon*
[b] *Notre Dame University–Louaize, Mathematics Department, Zouk Mosbeh, PO Box 72, Lebanon*

A R T I C L E   I N F O

Article history:
Received 15 July 2019
Accepted 16 October 2019
Available online 14 November 2019

Keywords:
Optimized regression
Tabu-search
Kernel optimization
Data-driven

A B S T R A C T

In the past decade, data science became trendy and in-demand due to the necessity to capture, process, maintain, analyze and communicate data. Multiple regressions and artificial neural networks are both used for the analysis and handling of data. This work explores the use of meta-heuristic optimization to find optimal regression kernel for data fitting. It is shown that optimizing the regression kernel improve both the fitting and predictive ability of the regression. For instance, Tabu-search optimization is used to find the best least-squares regression kernel for different applications of buckling of straight columns and artificially generated data. Four independent parameters were used as input and a large pool of monomial search domain is initially considered. Different input parameters are also tested and the benefits of using of independent input parameters is shown.

1. Introduction

Data sciences and artificial intelligence are nowadays the newly booming combination in research. With the impressive progress in computing power and numerical algorithms, handling big data through regression and machine learning is getting popular. With the ease of use and high accuracy, machine learning-based models are competing with well established constitutive models. Some recent works tend to replace long known constitutive models with data-based identification [1, 2], while others correct the uncertainty or ignorance in the models using regression on error values [3]. However, some other materials constitutive models are highly non linear and not well established. In such case, the data-driven approach constitutes an appealing route.

Artificial neural networks are indeed a good and easy tool to use for fitting, classification, and deriving new models out of data, however their accurate training requires a large number of data points. Regression on the other side requires a lower number of data points to efficiently find a good fitting [4]. Moreover, regressions have the possibility to illustrate the most prominent parameters of the buckling load, allowing therefore the user to plan an optimization scheme for the considered panel. However, a regression should give a form to the fitting model, unlike the artificial neural network. The form of the regression is often named by its "kernel". Different regression optimization schemes exist in the literature, either by using support vector regression to correct any bias in the resulting data [5], or by clustering the data and generating a "neighbor-

* Corresponding author.
E-mail addresses: cghnatios@ndu.edu.lb (C. Ghnatios), rhage@ndu.edu.lb (R.-M. Hage), ilige.hage@ndu.edu.lb (I. Hage).

https://doi.org/10.1016/j.crme.2019.11.006
1631-0721/

hood based" regression [6]. Other optimization techniques tend to identify regression parameters by using unconventional optimization algorithms like the bee colony algorithm using in [7]. However, previously mentioned type of optimization applies a constant regression kernels. In general kernel optimization or parameters learning are treated after a selection of a learning kernel, although kernel optimization may lead to better results[8]. Some other works aim to optimize the selected basis functions for regression, or the selected kernel parameters for the regression, using meta-heuristic algorithms in a large search space [9,10].

Glover [11] was the first to introduce the term meta-heuristic which represents a class of promising algorithms for solving hard optimization problems. Aiming to find near optimal solutions meta-heuristic algorithms work at efficient and comprehensive exploration of the search space, using the governing mechanisms which imitate certain strategies taken from nature, social behavior, physical laws, etc. Some popular global optimization algorithms include: genetic algorithm (GA), simulated annealing (SA), particle swarm optimization (PSO), ant colony optimization (ACO), artificial bee colony (ABC), taboo search (TS), quantum annealing, artificial immune system (AIS), improved harmony search algorithm (IHSA), real coded genetic algorithm (RCGA) and many more... The past 20 years have witnessed the development of numerous meta-heuristic algorithms in several fields, including artificial intelligence, computational intelligence and soft computing [12]. The structure of meta-heuristic algorithms starts with an initial set of independent variables and then evolving to obtain the global minimum/maximum of the objective (fitness) function. Meta-heuristics seeks on improving the solution iteratively starting from an initial one built by some heuristic until a stopping criterion is met. The stopping criterion can be elapsed time, number of iterations, etc. The operation of meta-heuristic algorithms works in such a way to determine the final solution, while only some existing solutions are visited. Each meta-heuristic algorithm under its own specific process attempt to find good solutions with no guarantee to be the greatest [13]. The optimization procedure for any type of meta-heuristic algorithm can be described as follows [14]:

– initializing the population in the search domain $\mathcal{F}$ by seeding the population with random values;
– evaluating the fitness for each individual of the population;
– generating a new population by reproducing selected individuals through evolutionary operations, such as crossover, mutation, and so on;
– looping to step 2 until stopping criteria are satisfied.

Optimization problems arise in various disciplines such as engineering design, manufacturing system, economics etc. Identification of the optimal machining parameters for example is very important for reduction of machining costs, product quality improvement and increased productivity and profit. Therefore, the meta-heuristic algorithms have been increasingly used to further improve the solution of machining optimization problems with complex nature in many applications [15]. It is reported that the meta-heuristic algorithms have been applied in machining because of their ability to deal with highly complex, non-linear, and multi-dimensional machining optimization problems [16].

A review paper dedicated to machining parameters optimization by means of the meta-heuristic algorithms is presented by Yusup et al. [17]. As has been reported in the literature, three types of meta-heuristic-based search algorithms e.g. GA, SA and PSO have been mostly applied in the domain of the machining parameters optimization. However, in recent years there is an increasing trend in the application of other meta-heuristic algorithms such as ACO, ABC, IHSA, and CSA for solving machining optimization problems. In [18] the authors have chosen four meta-heuristic algorithms, namely, RCGA, SA, IHSA and CSA for optimal combinations of different machining parameters for five case studies taken from the literature. IHSA has proved to be the most efficient meta-heuristic algorithm in terms of computational time and solution quality. In [19], the authors applied GA, TS, and SA as Meta-heuristic algorithms for solving the Real life Quadratic Assignment Problem (QAP). Results showed that GA, TS, and SA algorithms have effectively demonstrated the ability to solve QAP optimization problems. Computational results showed that genetic algorithm has a better solution quality than the other meta-heuristic algorithms for solving QAP problems. Tabu-search algorithm has a faster execution time than the other meta-heuristic algorithms for solving real-life QAP problems.

TS has proved remarkably powerful in finding high-quality solutions to computationally difficult combinatorial optimization problems drawn from a wide variety of applications [20–22]. The authors in [23] compared six meta-heuristic optimization algorithms applied to solving the traveling salesman problem: three classical approaches: GA, SA and TS, and compare them with three recently developed ones: QA, PSO and HS. It was shown that simulated annealing and tabu-search outperform newly developed approaches in short simulation runs with respect to solution quality, standard deviation of results and time needed to reach the optimum. SA finds best solutions, yet tabu-search has lower variance of results and faster convergence.

In this work, Tabu-search meta-heuristic optimization algorithm possibilities are explored in the framework of kernel regression optimization. Thus, a pool of variables or a search domain $\mathcal{F}$ is defined, and the best least squares regression kernel would be automatically selected by the algorithm. It is not the first time this technique is used. In [24], the authors combined tabu-search to regression (dubbed TS-REG) which yielded to a robust methodology that was compared to regression alone, artificial neural network and genetic algorithm and found that TS-REG is more reliable when dealing with large number of parameters. While in [10], tool life in turning has been estimated using a novel combination of tabu-search optimization and regression analysis (named TS-REG) and compared against regression analysis and artificial neural networks. The required time to select among 16 independent variables only the significant ones with highest R-square was 3 minutes (max), a minimum amount of time compared to other deterministic, Bayesian, and neural networks approaches. Comparing regression standalone with tabu-search combined with regression TS-REG, the latter select only the significant variables with the lowest p-values, the highest R-square better than deterministic, Bayesian and neural networks approaches, Comparing alternative regression with t-search combined with regression, the R-square was the highest, the p-values were the lowest and the mean absolute percentage error for the latter was the lowest. It is shown that Tabu-search optimized regression outperforms stepwise, backward or forward algorithms used in classical statistical packages offered by different commercial software [9].

The article starts with a review of the used taby search meta-heuristic algorithm and the selected regression methods, then tests the approach on a classical buckling problem. Later on, the optimized regression is tested on different complicated target functions, while exploring the effect of design for experiment selection on the results.

2. Tabu-search optimized regression

In this section we illustrate the optimized regression using the Tabu-search algorithm technique. For the sake of simplicity, linear multiple regression will be considered for this work even though any nonlinear kernel-based regression could be used. We assume a set of inputs $\mathbf{x}$ with known outputs $\mathbf{y}$. A regression tends to find the weights of the inputs such as:

$$\mathbf{X}\mathbf{b} + \boldsymbol{\epsilon} = \mathbf{y} \tag{1}$$

with $\mathbf{X}$ a set of vectors or variables, where each $X_i = \mathbf{f}_i(\mathbf{x}) \in \mathcal{F}$, and $\mathbf{X}$ will be named as the kernel of functions depending on $\mathbf{x}$. $\boldsymbol{\epsilon}$ is the bias of the regression. In our work we are considering a list of possible kernel functions $\mathcal{F}$, consisting of a list of monomial functions, which can take negative and positive exponents. The weights $\mathbf{b}$ are found using the least squares minimization technique:

$$\mathbf{b} = \left(\mathbf{X}^\intercal\mathbf{X}\right)^{-1}\left(\mathbf{X}^\intercal\mathbf{y}\right) \tag{2}$$

and an estimation of the fitted function $\hat{\mathbf{y}}$ can be found using:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{b} \tag{3}$$

The error of the fitting is therefore the bias $\boldsymbol{\epsilon} = \mathbf{y} - \hat{\mathbf{y}}$ which can also be written as:

$$\boldsymbol{\epsilon} = (\mathbf{I} - \mathbf{H})\,\mathbf{y} \tag{4}$$

with $\mathbf{I}$ the identity matrix and $\mathbf{H} = \mathbf{X}\,(\mathbf{X}^\intercal\mathbf{X})^{-1}\,\mathbf{X}^\intercal$ the influence matrix, which is also the derivative of the estimation with respect to the measurements.

The regression results highly depends on the choice of the kernel functions $\mathbf{X}$. Thus, an optimization scheme is used to find the best combination of monomial functions in $\mathbf{X}$ starting from a predefined pool of functions $\mathcal{F}$. The selected optimization scheme is a multi-start Tabu-search algorithm. The Tabu-search algorithm starts from a random position in the domain, and defines an action to take to a neighboring element in the domain. A neighbor of the currently selected solution is defined as:

– adding one element $\mathbf{f}^*(\mathbf{x})$ from the pool of monomial functions $\mathcal{F}$ such as: $\mathbf{X}^* = \mathbf{X} \cup \mathbf{f}^*(\mathbf{x})$;
– substituting one element from $\mathbf{X}$ by another element from the pool $\mathcal{F}$;
– deleting one element X_i from the current kernel solution $\mathbf{X}$.

The coefficient of determination R^2 of the regression is computed for the regression at each step or at each change of kernel neighborhood $\mathbf{X}$. The neighborhood i considered as a better one if the coefficient of determination increased. However, the coefficient of determination increase is not always an indicator of the convergence of the data fitting in the regression. Thus, a composite cost function $\mathcal{C}$ is created to optimize the data fitting:

$$\mathcal{C} = 0.5 \times \left(R^2 + (1 - MAPE)\right) \tag{5}$$

where $MAPE$ is the mean absolute relative error defined by:

$$MAPE = \|\mathbf{y} - \hat{\mathbf{y}}\|^\intercal \cdot \left(\mathbf{y}^{-1}\right) \cdot \frac{1}{N} \tag{6}$$

N being the number of measurements in the $\mathbf{y}$ vector. The algorithm acts by maximizing $\mathcal{C}$ while only conserving the statistically relevant monomial terms in the found regression. T-student tests are thus performed after each iteration to test the relevance of each one of the found monomial terms.

Once a kernel is selected by the Tabu-serach algorithm, the outliers can be removed if having a high leverage. For example a measurement i is considered as an outlier if:

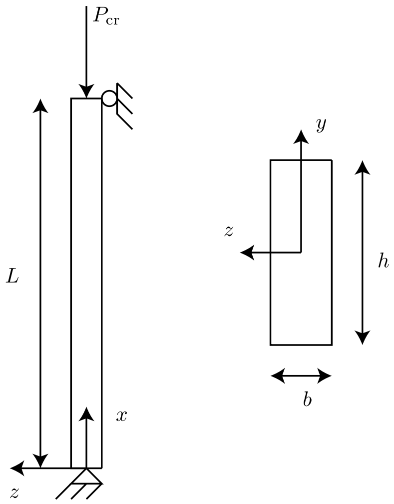

Fig. 1. The column studied for buckling to the left with its cross section to the right.

$$\mathbf{H}_{ii} > \frac{3p}{N} \tag{7}$$

p being the number of selected vectors in the kernel **X**. The outliers are therefore removed before attempting any regression fitting.

To avoid cycling and rechecking tested neighborhoods, the Tabu-search algorithm makes use of the Tabu list $\mathcal{L}$. If a variable X_j taken out of the pool $\mathcal{F}$ decreases the cost function $\mathcal{C}$ instead of increasing it, the variable X_j is added to the Tabu list. If a variable is listed in the Tabu list, it is taken out of the pool of possible kernel functions $\mathcal{F}$. The Tabu list has however a fixed length L. Once the Tabu list reaches its full length, the oldest variable in this list is deleted and enters back the list of possible kernel functions $\mathcal{F}$. The Tabu list helps minimizing the cycling effects of testing some already tested neighborhoods. However, the variables reducing $\mathcal{C}$ are not penalized forever, since they may reenter the list $\mathcal{F}$ after a while, considering that the best currently selected combination of variables **X** might have changed, and the variable X_j can potentially increase the cost function $\mathcal{C}$ with this new combination.

The presented algorithm is first applied on numerically generated data before attempting to fit the experimental data.

3. Numerically generated buckling data

Classical buckling of straight columns is a challenging issue for regression and fitting due to the high non linearities involved with respect to the parameters of the column. Classical mechanics relations gives the critical buckling load around the z axis P_{cr} of straight isotropic columns with rectangular cross section as a function of the column parameters as:

$$P_{\mathrm{cr}} = \frac{\pi^2 E I_z}{L^2} \tag{8}$$

With E being the modulus of elasticity of the column, L the length of the column, I_z its second moment of area around the z axis defined by:

$$I_z = \frac{1}{12} b h^3 \tag{9}$$

With b the dimension of the cross section parallel to the z axis and h the dimension normal to the cross section as illustrated in Fig. 1. The relations clearly illustrates the column length L to the power -2 and its dimension b to the power 3. We compute the value of the buckling load for $E = 200$ GPa and various values of L, b and h, obtaining therefore numerical data for about 2000 points.

The Tabu algorithm illustrated in section 2 is later on used to identify the best polynomial function that fits the obtained data. The initial pool of monomial $\mathcal{F}$ was initiated to all possible combinations of parameters having a power of -3 to $+3$, including the zeros, which leaves us with 194 potential monomials and the constant term. The algorithm converges to obtain therefore the monomial combination defined in Table 1, with $R^2 = 1$ and $MAPE = 1$.

One may note from Table 1 that the identified constant for the correct monomial function is exactly equal to $\pi^2 \times E$. All the other terms are negligible with a weight ratio less that 10^{17} with respect to the correct monomial. These weights can be attributed to numerical errors.

Now considering the absolute critical buckling load in mode 1, the equation of buckling becomes even more complicated even using classical mechanics:

Table 1
Identified monomials for buckling load around the z axis.

Monomial terms	weight
$\frac{bh^3}{L^2}$	1.64493×10^{11}
constant	5.13×10^{-8}
bhL	-7.38×10^{-6}
$\frac{1}{bhL}$	-5.28×10^{-14}
⋮	⋮

$$\begin{cases} P_{\text{cr}_z} = \frac{\pi^2 E I_z}{L^2} & \text{for } h \leq b \\ P_{\text{cr}_y} = \frac{\pi^2 E I_y}{L^2} & \text{for } b \leq h \end{cases} \tag{10}$$

with the second moments of area I_z and I_y defined by:

$$\begin{cases} I_z = \frac{1}{12} bh^3 \\ I_z = \frac{1}{12} hb^3 \end{cases} \tag{11}$$

Eventually the absolute minimum buckling load would lead to the first failure for $P_{\text{cr}_\text{a}} = \min\left(P_{\text{cr}_z}; P_{\text{cr}_y}\right)$. Using the Tabu-search algorithm and attempting to fit P_{cr_a} in one polynomial function is not a good idea since a step function needs theoretically an infinite number of monomials to correctly represent it. However such fitting may lead to a good fitting results at the expense of increasing the number of monomial terms and the computation time. However, fitting P_{cr_z} and P_{cr_y} in two separated polynomial functions and later on taking the solution as the minimum of the two polynomial results $P_{\text{cr}_\text{a}} = \min\left(P_{\text{cr}_z}; P_{\text{cr}_y}\right)$ would lead to the exact solution to the problem. This fact would inspire our next part of the work fitting therefore each mode of buckling with a different polynomial function.

4. Artificially generated data fitting

In this section we illustrate data fittings of three different numerically generated data set using 4 variables x_1, x_2, x_3, and x_4. The data sets are first generated with a polynomial easy functions consisting of four monomials. Later on a more challenging data set is generated with 20 different monomial terms generated using random powers and monomial coefficients. Finally, a challenging function is used with monomial terms taking possibly non-integer powers. The results are shown for different meshing used for the input variables x_i, all the considered input are included in $[0; 10]$. The considered pool of functions $\mathcal{F}$ consists of all possible combination of monomials with a degree lower than 4.

4.1. An easy monomial function

In this section we illustrate the usage of Tabu-search optimized regression to fit the polynomial target function:

$$y_1 = 5x_1^2 + 0.26x_1x_3^3 + 0.53x_2x_3 + 26x_4 \tag{12}$$

The tested function y_1 consists apparently of an easy test for the Tabu-search algorithm. The test is performed on 501 generated data points and split as follows: 80% (401 points) are used for estimation/training while the remaining 20% (100 points) are used for prediction/evaluation of the regression. The test is performed on three different meshes for x_i:

(i) A mesh consisting of random values for each one of the x_i, named mesh 1.
(ii) A mesh consisting of all possible combinations of selected values for each one of the x_i, as usually performed in an experimental plan, named mesh 2.
(iii) A uniform mesh without repetition of nodal values, named mesh 3.

The regression results for mesh 1 using random values for each x_i yields the following least-square fitted equation (13) while enforcing the presence of a constant in the polynomial regression:

$$\hat{y}_1 = 5x_1^2 + 0.26x_1x_3^3 + 0.53x_2x_3 + 26x_4 + 2.98 \times 10^{-8} + 4 \times 10^{-15}x_2^2x_3^2 \tag{13}$$

The difference between y_1 and $\hat{y}_1$ is the presence of two terms, a constant and the interaction between x_2^2 and x_3^2. Both extra terms are weighted many orders of magnitude lower than the other terms originally included in y_1. The resulted regression yielded to an accurate fitting and prediction as shown in the goodness of fitting curve illustrated in Fig. 2. Fig. 2 shows an excellent fitting with a minimal error on both the fitting and the training data. The two extra terms can be associated with numerical errors in the least square polynomial fitting.

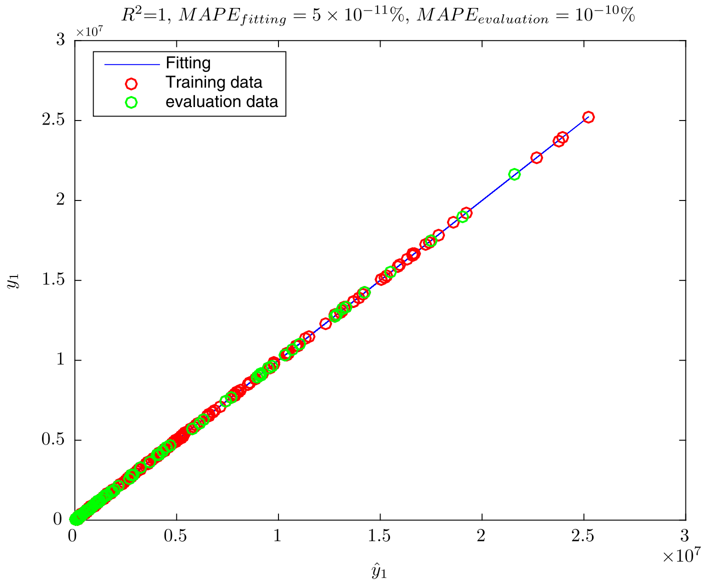

Fig. 2. Goodness of fitting of y_1 as a function of $\hat{y}_1$ for x_i mesh using random values.

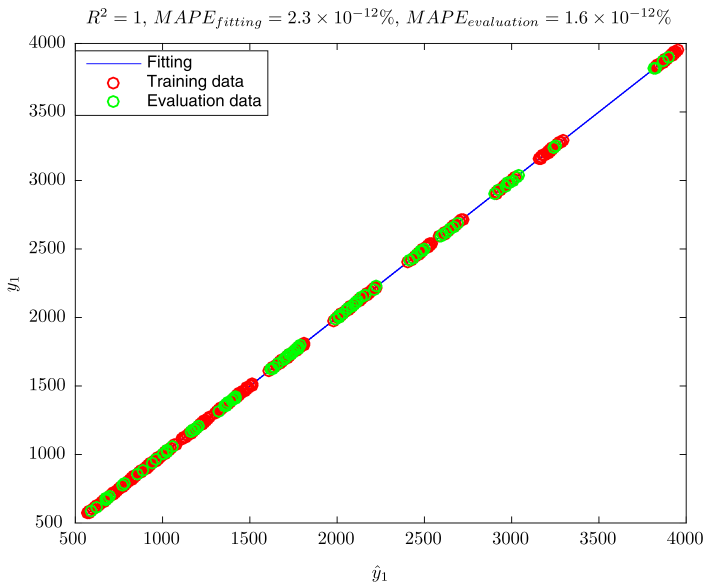

Fig. 3. Goodness of fitting of y_1 as a function of $\hat{y}_1$ for x_i mesh using all possible combination of selected pool values.

The regression results for mesh 2 using all possible combination of values from a selected pool for each x_i yields the following regression when enforcing the presence of a constant in the polynomial regression:

$$\hat{y}_1 = 5x_1^2 + 0.26x_1x_3^3 + 0.53x_2x_3 + 26x_4 + \left(-7.73 - 37.5x_1^2x_2 + 1.4x_2\right) \times 10^{-11} \tag{14}$$

Again, the optimized regression was able to find the terms included in y_1, with an error having 11 orders of magnitude lower than the tested function. The goodness of fitting is illustrated in Fig. 3.

Finally, the regression was also tested for a polynomial regression with the input variables x_i taking uniform mesh values, parallel as what is performed in a unidirectional finite element mesh for example. The regression yields:

$$\hat{y}_1 = 125.68 + 0.13x_2 + 35.93x_3^4 + 1.49x_4^2 \tag{15}$$

Eventually the regression shown in equation (15) is not comparable a priori to the function y_1 shown in equation (12). However, since the meshes are uniform without repetition, a correlation may exist between the x_i and these variables

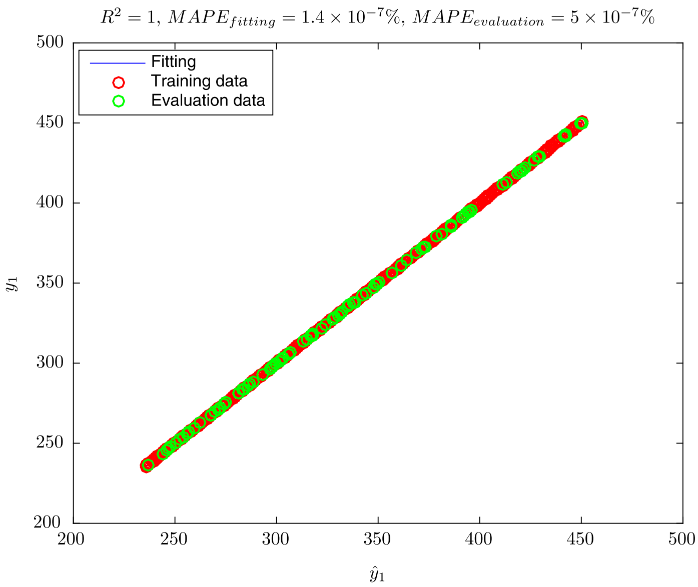

Fig. 4. Goodness of fitting of y_1 as a function of $\hat{y}_1$ using a uniform finite element 1D mesh for x_i values. Excellent fitting is found despite not finding the exact y_1 function.

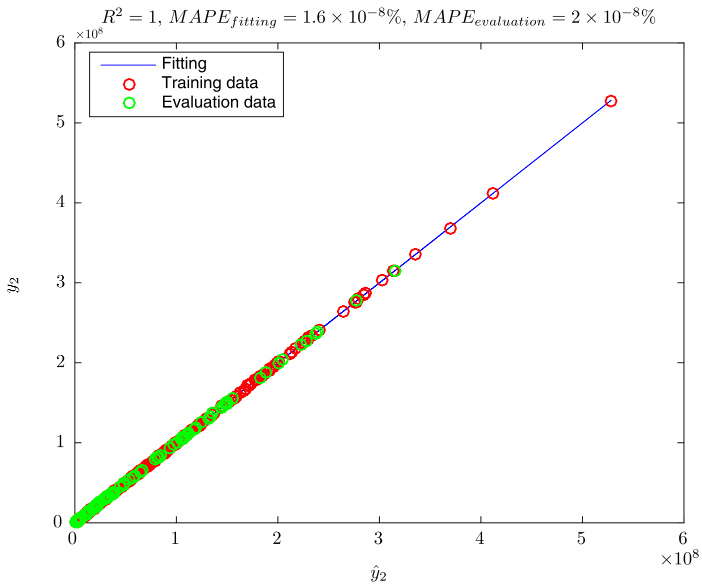

Fig. 5. Goodness of fitting of y_2 as a function of $\hat{y}_2$ for x_i taking random values for $i = 1, \cdots, 4$.

are therefore not totally independent. For instance, x_i may be derived from x_j using $x_i = a \cdot x_j + b$ with a and b are two constants. Thus, the Tabu-search have found the best regression to fit the data, taking into consideration the correlation between the x_i, which cannot be considered as independent variables, since we can get any x_i form scaling and translating the others. The goodness of fitting shown in Fig. 4 shows therefore an excellent fitting results for both the training data set and the testing or evaluation data set.

The trial test in this section on the target function y_1 illustrates the ability of the Tabu-search optimized regression to identify the exact functions y_1 when using independent input variables x_i. The use of correlated input variables would yield good fittings and evaluation, without having the exact initial target function.

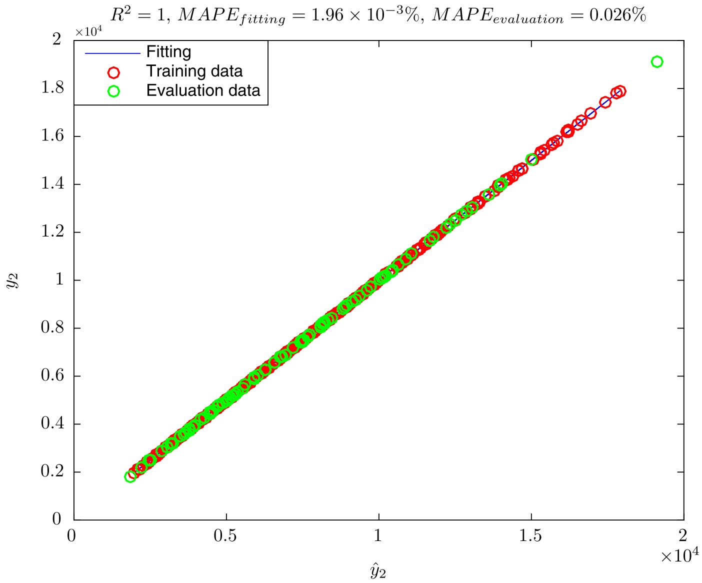

Fig. 6. Goodness of fitting of y_2 as a function of $\hat{y}_2$ for x_i taking all possible combinations of a pool of values, for $i = 1, \cdots, 4$.

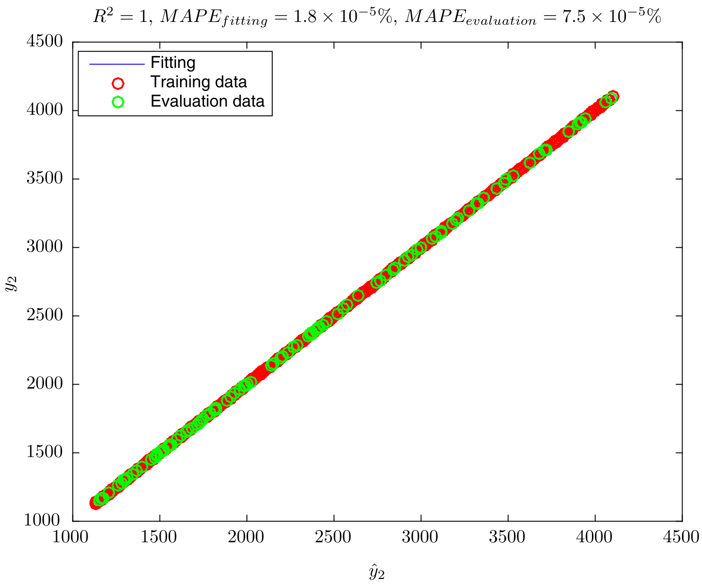

Fig. 7. Goodness of fitting of y_2 as a function of $\hat{y}_2$ for x_i taking uniform meshes values.

4.2. A complicated polynomial function

The target test function in this section y_2 is polynomial, however made of 20 different monomial terms with random integer exponents between 0 and 4, while keeping the degree of each monomial term lower than 4:

$$y_2 = \sum_{j=1}^{20} a_j \times \prod_{i=1}^{4} x_i^{c_{ji}} \tag{16}$$

where the exponents $c_{ji} \in [0; 4]$ are random integer and the polynomial coefficients $a_j \in [0; 10]$ are real constants. The Tabu-search is also performed on the three meshes detailed in section 4.1 with the goodness of fittings illustrated in Figs. 5, 6, and 7 for meshes 1, 2 and 3 respectively.

One may note that for mesh 1 using radom values for x_i, the Tabu-search optimized regression with 20 multi-starts was able to find the exact 20 monomial terms of y_2 6 times out of 20 multi-starts from random initial solutions. The identified

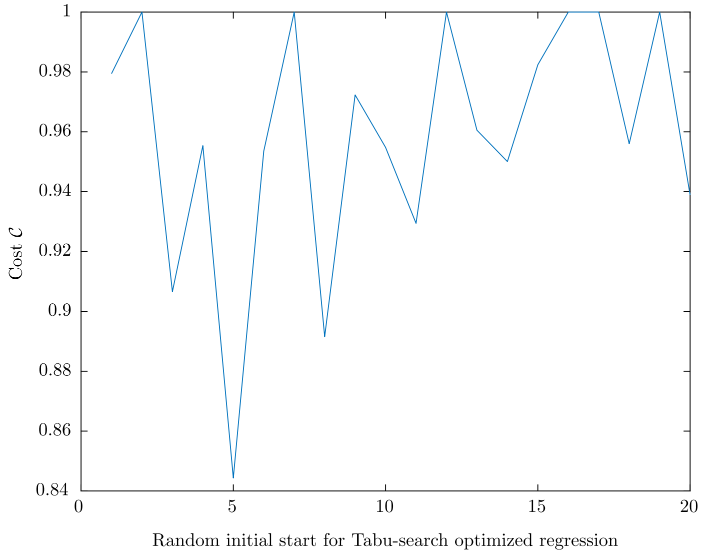

Fig. 8. Cost $\mathcal{C}$ for different starts for the Tabu-searched regression trying to identify y_2 using random values in x_i.

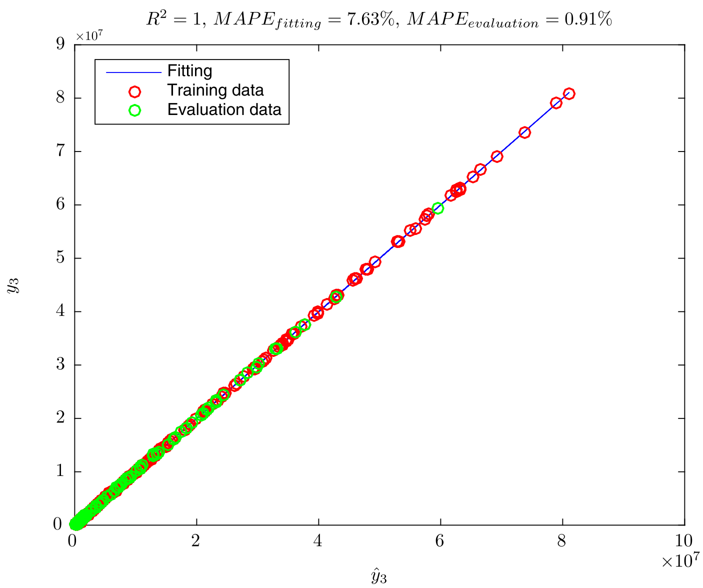

Fig. 9. Goodness of fitting of y_3 as a function of $\hat{y}_3$ for x_i taking random values for $i = 1, \cdots, 4$.

function corresponded to the 20 monomials with their exact coefficients a_j, along with few other monomial terms with coefficients several order of magnitudes lower than a_j. The composite cost function $\mathcal{C}$ is illustrated in Fig. 8 for the 20 multi-starts. Using mesh 2, which is common in experimental testings, the Tabu-search was able to find 14 exact monomial terms only, even after 20 multi-starts, while still finding excellent correlations for both training and testing data sets as shown in Fig. 6. Finally, for a finite element mesh 3, the Tabu-search optimized regression was able to find excellent fittings for both training and evaluation datasets using only 3 monomial terms due to the correlation between the non-independent input variables.

4.3. A challenging target function

In this section, we use a challenging target function y_3 to fit using the Tabu-search optimized regression. y_3 consists of five terms with a mix of integer and non-integer exponents:

$$y_3 = a_1 x_1^{0.5} x_3^2 x_4 + a_2 x_3^{0.1} x_4 + a_3 x_1^{0.3} + a_4 x_1^3 x_2 + a_5 x_1 x_3^{0.2} \tag{17}$$

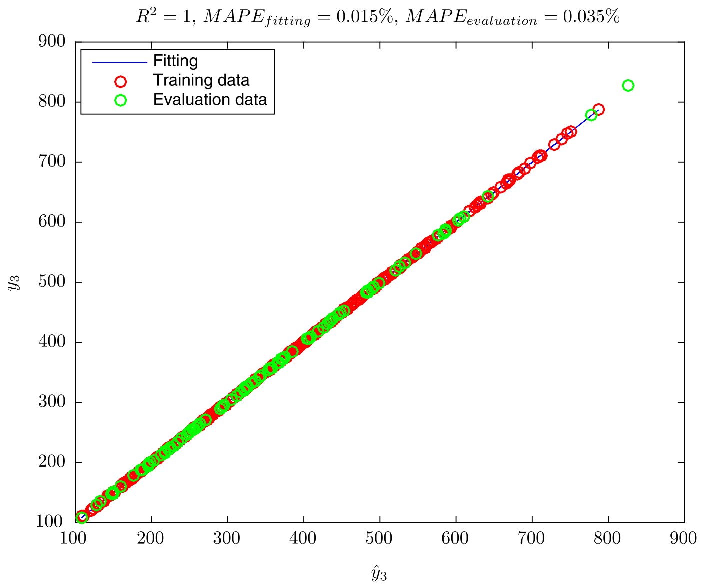

Fig. 10. Goodness of fitting of y_3 as a function of $\hat{y}_3$ for x_i taking all possible collocation values for $i = 1, \cdots, 4$.

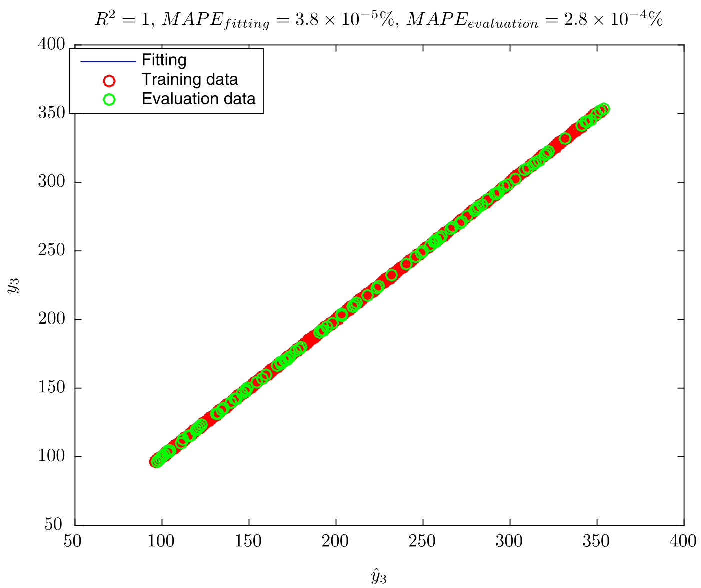

Fig. 11. Goodness of fitting of y_3 as a function of $\hat{y}_3$ for x_i taking uniform meshed values.

with a_j random coefficients of y_3 terms $\in [0; 10]$. y_3 is a challenging target function since it contains only one term from the pool $\mathcal{F}$, while all other terms are taken from outside the pool of functions $\mathcal{F}$. The regression is performed on the three meshes as performed in the previous sections. The goodness of fitting curves for meshes 1, 2 and 3 are respectively illustrated in Figs. 9, 10 and 11.

The illustrated results in Fig. 9 shows accurate fittings even though the considered y_3 terms are not initially found inside the $\mathcal{F}$ pool of variables. The optimized regression was able to find the best representation of the data using only 11 monomials of degrees lower than 4, including the term $x_1^3 x_2$ included in y_3 and available in $\mathcal{F}$.

Fig. 10 as well as 11 both shows excellent fitting results for both training and evaluation data sets. However, the regression using mesh 2 was able to identify the monomial term $x_1^3 x_2$ included in y_3 and $\mathcal{F}$, but the one using mesh 3 never found that term even though Fig. 11 show better results than 10. This is explained by the non-independent nature of x_i while using linear 1D finite element mesh to evaluate their values. The optimal regression using mesh 3 consisted of only 3 monomial terms, while the one found while using mesh 2 consisted of 21 monomial terms.

5. Conclusion

In this work we illustrate the use of Tabu-search optimization as a mean to find the best regression kernel, starting from a large pool of variables $\mathcal{F}$. The ability to optimize the regression kernel improves the fitting abilities as well as the predictive ability of a regression as illustrated in sections 3 and 4. The application of the algorithm is illustrated on buckling of straight columns and on three different sets of artificially generated data. Four parameters were used as regression input with a large pool of monomials is initially considered. Different input parameters are also tested and the benefits of using of independent input parameters is shown. The Tabu-search optimized regression was also able to find the best representation of the data when using correlated input variables x_i, with a lower number of terms than the ones used to create initially the target function y. The final fits illustrate low mean absolute percentage errors, high R^2 and a high predictive ability as illustrated in section 4. It is also important to highlight that only statistically relevant terms are derived by the designed algorithm.

References

[1] R. Ibanez, D. Borzacchiello, J. Vicente Aguado, E. Abisset-Chavanne, E. Cueto, P. Ladeveze, F. Chinesta, Data-driven non-linear elasticity: constitutive manifold construction and problem discretization, Comput. Mech. 60 (5) (2017) 813–826.

[2] R. Ibanez, E. Abisset-Chavanne, J. Vicente Aguado, D. Gonzalez, E. Cueto, F. Chinesta, A manifold learning approach to data driven computational elasticity and inelasticity, Arch. Comput. Methods Eng. 25 (1) (2018) 47–57.

[3] W.C. Moore, S. Balachandar, G. Akiki, A hybrid point-particle force model that combines physical and data-driven approaches, J. Comput. Phys. 385 (May 2019) 187–208.

[4] M. Cerny, Narrow big data in a stream: computational limitation and regression, Inf. Sci. 486 (2019) 379–392.

[5] Z. Zhang, Z. Le, G. Gao, Y. Tian, Bi-sparse optimization-based least squares regression, Appl. Soft Comput. J. 77 (2019) 300–315.

[6] Z. Chen, Y. Zhou, X. He, Handling expensive multi-objective optimization problems with a cluster-based neighborhood regression model, Appl. Soft Comput. 80 (2019) 211–225.

[7] M. Zhu, A. Han, Y.-Q. Wen, W.-Q. Sun, Optimized support vector regression algorithm-based modeling of ship dynamics, Appl. Ocean Res. 90 (2019) 1–17.

[8] M. Strazar, T. Cruk, Approximate multiple kernel learning with least-angle regression, Neurocomputing 7 (2019) 245–258.

[9] J. Pacheco, S. Casado, L. Nunez, A variable selection method based on tabu search for logistic regression models, Eur. J. Oper. Res. 199 (2009) 506–511.

[10] R.M. Hage, I. Hage, C. Ghnatios, I.S. Jawahir, R. Hamadey, Optimized tabu search estimation of wear characteristics and cutting forces in compact core drilling of basalt rock using pcd tool inserts, Comput. Ind. Eng. 136 (2019) 477–493.

[11] F. Glover, Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (5) (1986) 533–540.

[12] S. Gholizadeh, H. Barati, Comparative study of three metaheuristics for optimum design of trusses, Int. J. Optim. Civ. Eng. 3 (2012) 423–441.

[13] F.G. Gomes de Freitas, C.L. Brito Maia, G.A. Lima de Campos, J. Teixeira de Souza, Optimization in software testing using metaheuristics, Rev. Sist. Inf. FSMA 5 (2010) 3–13.

[14] M. Khajehzadeh, M. Raihan-Taha, A. El-Shafie, M. Eslami, A survey on meta-heuristic global optimization algorithms, Res. J. Appl. Sci., Eng. Technol. 3 (6) (2011) 569–578.

[15] A.R. Yildiz, A novel particle swarm optimization approach for product design and manufacturing, Int. J. Adv. Manuf. Technol. 40 (5) (2009) 617–628.

[16] I. Mukherjee, O.K. Ray, A review of optimization techniques in metal cutting processes, Comput. Ind. Eng. 50 (4) (2006) 15–34.

[17] N. Yusup, A.M. Zain, S.Z.M. Hashim, Evolutionary techniques in optimizing machining parameters: review and recent applications (2007-2011), Expert Syst. Appl. 39 (10) (2012) 9909–9927.

[18] M. Madic, D. Markovic, M. Radovanovic, Comparison of meta-heuristic algorithms for solving machining optimization problems, Facta Univ., Mech. Eng. 11 (1) (2013) 29–44.

[19] G. Abd El-Nasser Said, A.M. Mahmoud, E.-S. El-Horbaty, A comparative study of meta-heuristic algorithms for solving quadratic assignment problem, Int. J. Adv. Comput. Sci. Appl. 5 (1) (2014) 1–6.

[20] E.H.L. Aarts, Local Search in Combinatorial Optimization, John Wiley & Sons, Chichester, 1997.

[21] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, Boston, 1998.

[22] Z. Drezner, G. Marcoulides, Tabu search model selection in multiple regression analysis, Commun. Stat. Part B, Simul. Comput. 28 (2) (1999) 349–367.

[23] M. Antosiewicz, G. Koloch, B. Kaminski, Choice of best possible metaheuristic algorithm for the travelling salesman problem with limited computational time: quality, uncertainty and speed, J. Theor. Appl. Comput. Sci. 7 (1) (2013) 46–55.

[24] R. Hage, I. Hage, C. Ghnatios, R. Hamade, Statistically validated and optimized tabu search estimation of cutting tool life in turning, in: ASME 2018 International Mechanical Engineering Congress and Exposition, 2018.