Order of Non-linearity as a Complexity Measure for Models generated by Symbolic Regression via Pareto Genetic Programming

Ekaterina J. Vladislavleva, Member, IEEE, Guido F. Smits, Member, IEEE, Dick den Hertog

Abstract—This paper presents a novel approach to generate data-driven regression models that not only give reliable prediction of the observed data but also have smoother response surfaces and extra generalization capabilities with respect to extrapolation. These models are obtained as solutions of a genetic programming process, where selection is guided by a tradeoff between two competing objectives - numerical accuracy and the order of nonlinearity. The latter is a novel complexity measure that adopts the notion of the minimal degree of the best-fit polynomial, approximating an analytical function with a certain precision.

Using nine regression problems, the paper presents and illustrates two different strategies for the use of the order of nonlinearity in symbolic regression via genetic programming. The combination of optimization of the order of non-linearity together with the numerical accuracy strongly outperforms "conventional" optimization of a size-related expressional complexity and the accuracy with respect to extrapolative capabilities of solutions on all nine test problems.

In addition to exploiting the new complexity measure, the paper also introduces a novel heuristic of alternating several optimization objectives in a two-dimensional optimization framework. Alternating the objectives at each generation in such a way allows us to exploit the effectiveness of two-dimensional optimization when more than two objectives are of interest (in this paper, these are accuracy, expressional complexity and the order of non-linearity). Results of the experiments on all test problems suggest that alternating the order of non-linearity of GP individuals with their structural complexity produces solutions that are both compact and have smoother response surfaces, and, hence, contributes to better interpretability and understanding.

Index Terms—complexity, model selection, evolutionary multi-objective optimization, genetic programming, extrapolation, industrial data analysis.

I. INTRODUCTION

Fundamental model building is a time- and labour-consuming process in industrial engineering – particularly in polymer research. When time is crucial, empirical estimation of fundamental relationships in process variables is certainly preferable to discovering first-principle models and developing mathematical apparatus to operate with them. Outputs (e.g. system control variables) are usually difficult to monitor, can be measured only in a lab, or require expensive hardware for analysis. The measurements are taken within a limited range of operating conditions and are usually available off-line. However, models built upon these measurements are expected to be used to control dynamic processes on-line.

Since industrial data is always corrupted by noise and is driven by a combination of measured and unmeasured process variables, models should not only accurately predict the observed output but also have some extra generalization capabilities. Examples of such capabilities are insensitivity to a certain amount of noise in the inputs or a capability to extrapolate the output outside the observed region. In addition, generated models should be interpretable, in order to provide additional understanding of the underlying process. The requirements for empirical models in industrial settings are defined as follows (see also [1]):

- Capability for on-line reliable prediction of process outputs within the given range of operating conditions and outside this range.
- Interpretability and the possibility of integrating information from first principles.
- Low maintenance and development costs with no (or negligible) operator interference.
- Robustness with respect to the variability in process inputs.
- The ability to detect novelties in data to attune itself toward changes in the process.

There is no single technique producing models that fulfil all of the requirements listed above. The creation of an input-output model the main emphasis is usually put on the goodness of fit in the given range of inputs. For real-life problems, however, the other requirements are at least equally valuable.

Several empirical modeling techniques are used for constructing input-output regression models including linear regression, nonlinear regression, [2], kriging, [3], radial-basis functions, [4], neural networks, [5], support vector machines (SVMs), [6], [7], and genetic programming, [8]. These techniques are used together to complement each other in modeling complex chemical processes. Neural networks, SVMs and genetic programming have advantages over classical statistical methods in the following cases (see [1]):

- Where no or negligible a priori information is known about the process and no assumptions on models can be made.
- Where modeling problems are multi-dimensional with either too much or too little data.
Symbolic regression via genetic programming has incontestable advantages over neural networks and support vector machines for problems where models have to be simple and interpretable analytic expressions and have a reasonable generalizing behavior ([1], [9]).

Symbolic regression via genetic programming is a methodology to automatically generate symbolic models that describe functional relationships on given data. Inspired by principles of natural selection, symbolic regression via GP sets up an artificial evolution of models, optimizing some of them, evaluating how well they fit the observed data, and then using this information to decide which models to use as parents for the next generation ([8], [10–12]). Such evolutionary search continues until a perfect solution is found or the allotted computation time is exceeded. If no a priori information is given about the relationship between inputs and outputs, the search space is a set of all possible symbolic models representing valid operations from the fixed set on the given input variables. Classical numerical optimization techniques applied in an empirical modeling framework can be less effective, if they make certain assumptions about the structure of input-output models a priori and, hence, limit the search space or make the search biased.

The benefits of symbolic regression via genetic programming include the following:

- There are no prior assumptions on model structure.
- The final predicting model or model ensemble is chosen from a rich set of nonlinear empirical models that is generated automatically.
- Sensitivity analysis of the inputs and variable selection is implicitly performed with no extra cost, which reduces the dimensionality of the problem.
- No assumptions are made on independence of input variables.
- Insight can be provided due to the symbolic representation of models (e.g. in a form of low-order variable transformations that can be used as ‘natural’ meta-variables).

Since the class of potential nonlinear solutions is broad, the main problem is to choose the model or a set of models of optimal complexity ([1], [7], [13]). The model selection problem, together with the requirements for a reliable model, lead us to the following questions: How can we measure and control the complexity of models? How can we quickly assess their dissimilarities in interpretability, flexibility to fit the data, and ability to predict the outputs outside the training range?

This paper introduces a new complexity measure reflecting the features of the response surfaces related to symbolic models. The process of model selection under complexity control with this measure, called the order of non-linearity, improves the extrapolation capabilities of solutions generated by symbolic regression.

In addition to the new complexity measure, a novel heuristic is also introduced, aimed at producing accurate solutions that are genotypically and phenotypically simple. The process of two-objective optimization of model error and expression complexity alternated at each generation with the two-objective optimization of model error and the order of non-linearity produces competitive solutions that have low expressive complexity and low orders of non-linearity.

The paper is organized as follows. Section II briefly describes the concept of symbolic regression via genetic programming and touches on various issues of the complexity control. Section III introduces a new complexity measure reflecting the order of non-linearity of the evolved models. Section IV describes the results of testing the new measure empirically. Section V concludes by discussing the applicability of the order of non-linearity in a genetic programming framework.

II. SYMBOLIC REGRESSION VIA GENETIC PROGRAMMING

Construction of an unknown function in a high-dimensional space from a finite number of samples bears the risk of over-fitting. Care should be also taken to avoid the process noise. Therefore, from models approximating the noisy data, the ones that have minimal (optimal) complexity should be chosen. Okkam’s razor principle states: “No more things should be presumed to exist than are absolutely necessary.” Following this principle, we should limit and control the complexity of models we create, and favor the simplest ones to take part in the evolution. The structural complexity is not the only measure to minimize, in order to produce simple and interpretable models. Also the extrapolative capabilities of models are important, as is the presence of good local properties, which can be related to first-principles.

New models for the next iteration are generated from the current population of models by means of crossover, mutation, and copying. Models gain the right to propagate their good features to successive generations. The current implementation of symbolic regression via GP uses an elite-based selection strategy to select good models and endow them with improved propagation rights. This strategy consists of determination and preservation of a representative set of high quality models, obtained by a given step of evolution [14]. This elite set is stored in an archive [1], [14]. It is built based upon the concept of dominance in a space of selected optimization objectives. Propagation rights are granted to all archive members irrespective of their relative numerical goodness of fit. Archive members generate offspring for the next evolutionary step 1; and these new models are then used to optimize the archive. When the iteration process is terminated, the set of final GP solutions is determined by the archive at the last iteration step. An excellent description of the elite-based strategy for multi-objective evolutionary computation is given in [15], [16].

The process of updating the archive at each generation employs the concept of Pareto-optimality in a set of selected optimization objectives \( \Theta = C_1, C_2, \ldots, C_k \) (see [14], [15] and Figure 1). We use the non-dominated sorting in the objective space to create an archive of a constant size (see the example in Figure 2).

The fitness measure used during the evolution is the normalized mean-squared error \( N M S E \). It is bounded by 0 and 1 with a perfect fit corresponding to 1 (see equations (21)-(23)).

1In case of crossover we always choose one parent from the archive, and the other from a population.
A. Complexity control

In classical GP applications, [8], [10], [11], survival of the fittest is the only criterion to find the optimal model. However, besides numerical closeness to the observed data there are certainly more characteristics that reflect the quality of the generated models. In applications the measured data is almost always with noise, so that perfectly accurate models are not the goal. We presume that ‘the’ solution does not exist. Often models with high goodness of fit look so obscure that it becomes infeasible to convince process engineers to implement them for controlling real on-line processes. In these cases, simpler credible models with a lower level of fitness will be preferred over complex ones. Moreover, limiting the complexity of models may be vital in avoiding over-fitting of data and also modeling the process noise. Too complex models are difficult to use, whereas too simple models may give poor prediction. For classical modeling techniques, model complexity is controlled by a priori knowledge of the process and the true underlying relationship, [7]. In these cases, parsimony pressure is introduced in the fitness function, and a resulting composite fitness function is defined as a linear combination of prediction error and a complexity term (the latter is called a regularization coefficient).

Due to the fact that penalizing models for high complexity is a natural way to control bloat (i.e. excessive growth in the size of GP individuals without improvements in fitness), [8], [17–26], GP researchers have quite extensively analyzed evolutions under parsimony pressure and their relationship to bloat, [27–29]. The definition of the parsimony pressure as a linear term, added to the fitness function, causes GP to perform well on some problems, [20], [27], [30], and less well on others, [8], [18]. T. Soule and J. Foster showed in [28] that the linear coefficients in a composite fitness function, relating numerical fitness and the structural complexity, can be used as a good indicator of the performance of a GP population. However, the search for a good combination of these coefficients requires some intuition and empirical testing.

We firmly believe that when no a priori information about the problem is known, the measure for parsimony pressure has to be optimized individually and simultaneously with numerical fitness. This fosters an intelligent trade-off between model simplicity and model accuracy. Optimizing complexity and accuracy in a truly multi-objective way will exempt us from making risky assumptions about the exact relationship between complexity and accuracy, and will therefore not bias the search. With such bi-objective selection, the GP system is pushed to produce both accurate and simple individuals, from which the best ones form a Pareto front — a set of optimal trade-offs in the two-dimensional performance space of complexity and accuracy.

For the complexity definition, one can think of two directions in determining the qualitative complexity of the GP model: complexity of the model expression (compactness of the genotype) and behavior of the associated response surface (smoothness of the phenotype).

Two-dimensional optimization with striving for compact and accurate GP solutions has been shown to systematically outperform the standard single-objective GP on a variety of regression and classification problems [1], [31], [32]. Similar results have been reported for GA in [33], [34]. This paper aims to explore the feasibility of producing smooth and accurate GP solutions that do not necessarily have ‘simple’ structures, but have ‘simple’ response surfaces, and, hence, generalize better. If producing smooth and accurate solutions proves feasible, this will indicate the possibility of combining
the structural and behavioral complexity measures for creating both smooth and simple GP models, without a risk of either bloat or over-fitting.

B. Structural Complexity of an Expression

Until now, almost all complexity measures considered by the GP community have addressed the structural complexity of an individual.

An individual is a regression model – a functional relationship between system inputs and outputs built on a limited set of observations (measurements or other models). In our case, it is represented as a valid mathematical expression based on a set of given basic functions, input variables and random constants sampled from a certain range. Model expressions may vary in size and in this paper are represented as tree structures. All subtrees in a tree representation of an expression are considered as separate models during the selection process. Thus, more search space is effectively covered by a population of a fixed number of models.

A set of basic operations contains functions that have one or two arguments. Typical representatives of the set are the standard arithmetical functions: addition, subtraction, multiplication and division. They may also include power, trigonometric, logarithmic, exponential, and logical functions.

There is a variety of intuitive measures for determining the size of structures operated by a tree-based GP: 1) The number of nodes in a tree; 2) The number of layers in a tree; 3) Path length, etc.

We have been using a so-called expressional complexity (see [11]). This measure is determined by the sum of the number of nodes in all subtrees of a given tree. It favors the flatter trees (i.e., trees with fewer layers and, hence, with fewer nested functions) over deep unbalanced trees (in the case of an equal number of nodes). An example of such a case is shown in Figure 3. The expressional complexity can be interpreted as a size of the model obtained by substituting all inner functions of the model by their function bodies. M. Keijzer and J. Foster call this complexity measure a visitation length, show that it is a close relative of the path length, and provide a thorough review of its mathematical properties in [36].

The expressional complexity measure is successfully used throughout implementations in MatLab and Mathematica (DataModeler add-in). Optimizing the structural complexity of evolving models together with the goodness of fit produces compact solutions that are interpretable and reliable within the training range. After extrapolation, these solutions often demonstrate unwanted behavior, caused by over-fitting.

III. ORDER OF NON-LINEARITY OF AN EXPRESSION

A. Motivation

With respect to the quality of the function determined by the tree expression and the behavior of its response surface, there is a range of complexity measures:

1. The number of variables in the tree representation (the total number of variables present at the leaves is an indicator of model complexity; the number of unique variables reflects the dimensionality of the model); 2) The number of binary and unary functions present at inner nodes; 3) Some component-wise non-linearity of functions present at inner nodes (e.g., addition is less nonlinear, and, hence, simpler than exponentiation), etc.

Our main objective in measuring the non-linearity of a model is to favor smooth and extrapolative behavior of the response surface and to discourage highly nonlinear behavior (which is unstable towards minor changes in inputs and is dangerous for extrapolation). It would, moreover, be desirable to find a non-linearity measure that agrees with an intuitive impression of the complexity of an elementary function. In other words, we want exponentiation to be more complex than taking a square, summation to be simpler than multiplication, and taking a square root and division to be very complex in the neighborhood of zero.

In [37] we introduced a complexity measure that reflects the order of non-linearity of a model. It is a quantitative measure reflecting the nonlinear growth of the response function determined by a labeled tree. The definition is based on the minimum degree of a polynomial approximating the function on a certain interval with a certain precision.

The reasoning is simple: An obvious measure for the complexity of a multivariate polynomial is its degree. Any tabulated multivariate function can be associated with its unique best-fit approximating polynomial. The degree of this polynomial can be considered as a measure for the order of non-linearity of the response surface of the original function. This order can be seen as a numeric value of the deviation of the response surface from a linear hyperplane. To make this idea suitable for an evolutionary optimization framework, and in particular for our implementation of symbolic regression, we had to make several simplifications in its realization.

The best-fit polynomials are difficult to find [38]. Even though we settle for a polynomial giving a good approximation of the function, instead of the best-fit polynomial, we still need a procedure that is computationally efficient. Since our goal is to compare the behavior of response surfaces of GP models at each step of the evolution, we strove to:

1) calculate the order of non-linearity iteratively for a given
model, starting from the terminals;  
2) consider an efficient Chebyshev polynomial approximation of a function ([39], [40]), instead of a labor-intensive search for an optimal best-fit polynomial for a given precision;  
3) determine the order of non-linearity as the degree of the approximating polynomial only for univariate operators, and use another definition of non-linearity for bivariate functions and compositions.

The first implementation described in [37] used a least-squares polynomial approximation for complexity determination. Given a set of points \((x_1, y_1), \ldots, (x_n, y_n)\) and a maximum order \(p, (p < n)\), least-squares fitting produces a polynomial \(P_{LS} = \sum_{k=1}^{p} a_k x^k\) of degree \(p\) that minimizes the error \(E_{LS} = \sum_{i=1}^{n} (y_i - \sum_{k} a_k x_i^k)^2\) with respect to the coefficients \(a_k\). Certain conditions for points \(x_1, \ldots, x_p\) must hold for a least-squares polynomial to exist and be unique. For high degrees, the problem of finding a unique polynomial often becomes ill-defined. For better treatment of steep response surfaces, we therefore made good use of a more stable and reliable framework – Chebyshev approximations (see also [41]).

Our current implementation for a given accuracy constructs a Chebyshev polynomial approximation of a function, and takes the degree of the resulting polynomial as a basis for the measure of non-linearity of a univariate function. An exact definition of the order of non-linearity appears in the next subsection.

Before giving the definition, we would first like to comment on “approximation of a given accuracy”. It is said that \(P(x)\) approximates a continuous function \(f(x)\) on interval \([a, b]\) with accuracy \(\epsilon\), if

\[
\max_{x \in [a, b]} |f(x) - P(x)| \leq \epsilon. \tag{1}
\]

We change the above definition for error evaluation and consider a finite number of samples \(x \in S \subset [a, b]\). The way in which we determine the test set \(S\) indeed affects the true quality of the approximation. If \(S\) has too few points, then condition (1) is superficial.

If the test set consists of too many points, then error estimation can require excessive computational resources. The choice of \(S\) is dictated by a trade-off between the efficiency of computation and the desired accuracy. In the current implementation, the test set \(S\) consists of equidistant points whose number changes dynamically depending on the length of the interval \([a, b]\).

Further on, by a polynomial approximating a univariate function \(f\) on interval \([a, b]\) with a certain precision \(\epsilon\), we will denote an approximation in a class of Chebyshev polynomials; more precisely, a polynomial \(P_f(x) = \sum_{i=0}^{n-1} c_i T_i(x; a, b)\) of a minimal order, such that

\[
\max_{x \in S \subset [a, b]} |f(x) - P_f(x)| \leq \epsilon. \tag{2}
\]

Here, \(T_i(x; a, b) = T_i\left(\frac{2x - (b-a)}{b-a}\right), \ i = 1, \ldots, n - 1\), and \(T_i(z)\) is an \(i\)-th Chebyshev polynomial on \([-1, 1]\).

For a univariate function given analytically the degree of the approximating polynomial depends on the interval, in which the function is being approximated. This implies the necessity of including scaling into the definition of complexity and calculating the ranges for every inner node that corresponds to a univariate operation. In order to be able to treat bivariate functions as univariate (for polynomial approximations), we also need to estimate the ranges of inner nodes, corresponding to bivariate operations.

We would like to emphasize that the function ranges corresponding to inner nodes cannot be determined accurately from the ranges of terminals by using simple interval computations.

In general, the range evaluation of a function defined on a real interval should take into account the monotonicity and extrema of this function and may involve unwanted computation time.

There is a price to pay, however, to avoid these computations. Every subexpression of a symbolic model is explicitly evaluated to obtain the goodness of fit of predicted output to the original output. Therefore, the ranges of subexpressions can be estimated by simply taking the minimum and maximum of the predicted outputs in the fitness evaluation routine. Such evaluation of ranges of all subfunctions of the given model can indeed introduce some inaccuracy if the extrema of subfunctions are not at the sampling points. Since this is the best we can do without doing extra calculations, it will be good enough for an efficient comparison of the non-linearity of the GP models.

Once the ranges for all nodes in the tree-based symbolic model are found, then the order of non-linearity of this model can be computed according to the following inductive definition.

**B. Definition**

**Inductive definition of the order of non-linearity of a symbolic model**

Let a tree structure represent a valid analytical model over a set of variables \(\mathcal{V} = \{x_1, x_2, \ldots, x_{\text{var}}\}\), and a set of constants \(\mathcal{C} \subset \mathbb{R}\) with functions from a set \(\Phi = \Phi_1 \cup \Phi_2\), where \(\Phi_1 = \{\text{sqrt}(x), \ln(x), \exp(x), \exp(-x), \sin(x), \cos(x), x^{\text{const}}, \text{const}^\text{t}\}\), \(\Phi_2 = \{x + y, x \cdot y, x/y, x^y\}\). Assuming that the precision \(\epsilon\) is given, the complexity of the tree structure is calculated from the leaves to the root according to the following definition:

(A) The complexity of a single node referring to a constant \(\text{const} \in \mathcal{C}\) is zero:

\[
\text{comp}(\text{const}) = 0. \tag{3}
\]

(B) The complexity of a single node referring to a variable from \(x_i \in \mathcal{V}\) is one:

\[
\text{comp}(x_i) = 1. \tag{4}
\]

(C) The complexity of an inner node referring to unary function \(f \in \Phi_1\), is related to the complexity of the child node.
referring to a function, variable, or constant \( g, g \in \Phi \cup \mathcal{V} \cup \mathcal{C} \) and the range of the child node \([a, b]\) by the following formula:

\[
\text{comp}(f \circ g) = \text{comp}(g) \cdot n_f,
\]

where \( n_f \) is the minimal degree of \( P_f \), a Chebyshev approximation of function \( f(x), x \in [a, b] \) with approximation error \( \epsilon \) at most \( \epsilon \).

NB: The complexity of an inner node referring to a unary function \( f \in \Phi_1 \) is related to the complexity of the child node referring to a function, or to a variable \( g \in \Phi \cup \mathcal{V} \) and the range of the child node \([a, b]\) by the following formula:

\[
\text{comp}(f \circ g) = \text{comp}(e^{g \ln \text{const}}) = \text{comp}(g) \cdot n_{\text{power}},
\]

where \( n_{\text{power}} \) is the minimal degree of a Chebyshev approximation of function \( e^{g \ln \text{const}}, x \in [a, b] \) with the approximation error \( \epsilon \) at most \( \epsilon \).

(D) The complexity of an inner node referring to summation and subtraction \( \{+,-\} \in \Phi_2 \) is related to the complexities of child nodes referring to \( g_1, g_2 \) from \( \Phi \cup \mathcal{V} \cup \mathcal{C} \) by the formula:

\[
\text{comp}(g_1 + g_2) = \max\{\text{comp}(g_1), \text{comp}(g_2)\},
\]

\[
\text{comp}(g_1 - g_2) = \max\{\text{comp}(g_1), \text{comp}(g_2)\}.
\]

(E) The complexity of an inner node referring to multiplication \( \{\times\} \in \Phi_2 \) is related to the complexities of child nodes referring to \( g_1, g_2 \) from \( \Phi \cup \mathcal{V} \cup \mathcal{C} \) by the formula:

\[
\text{comp}(g_1 \cdot g_2) = \text{comp}(g_1) + \text{comp}(g_2).
\]

(F) The complexity of an inner node referring to division \( \{/\} \in \Phi_2 \) is related to the complexities of child nodes referring to \( g_1, g_2 \) from \( \Phi \cup \mathcal{V} \cup \mathcal{C} \) with ranges \([a, b]\) and \([c, d]\) by the formula:

\[
\text{comp}(g_1 / g_2) = \text{comp}(g_1) + \text{comp}(g_2) \cdot n_{\text{div}},
\]

where \( n_{\text{div}} \) is a minimal degree of the Chebyshev approximation of a function \( 1/x \) on interval \( x \in [c, d] \) with the approximation error \( \epsilon \) at most \( \epsilon \).

(G) The complexity of the root node determines the complexity of the tree structure.

The inductive definition described above is an algorithm to calculate the order of non-linearity.

In the current implementation, the maximum admissible degree of the Chebyshev approximation is limited to 100. If the precision of the approximation by an 100-order polynomial still exceeds \( \epsilon \), then the complexity value is set to a predefined limit. This limit value is 10,000 for the current implementation. The value for precision \( \epsilon \) is fixed to 0.0001. The fixed precision used for estimating the degree of Chebyshev approximation will imply higher degrees for wider domain ranges. This penalizes the GP system for constructing solutions with too much diversity in the ranges of the inner nodes.

The number of points at which the approximation error is evaluated is dynamic, and depends on the length of the interval of approximation. Currently, we take \( \max\{20 \cdot |b - a|, 500\} \) equidistant points on \([a, b]\). This number should vary, depending on the problem difficulty and the descriptiveness of the input data file.

An example of the non-linearity calculation for a simple two-variable model for \( \epsilon = 10^{-6} \) is given in Figure 4.

C. Discussion

The definition of the order of non-linearity implies that the complexity of a parent model is never less than the non-linearity of any of its submodels. This definition allows us to implicitly take the complexity of the representation into account, and make the order of non-linearity a characteristic of a genotype. Often this causes over-estimation of the true order of non-linearity of the simplified expression. We do this deliberately to push the system towards creating simplified expressions, and to penalize possible precision errors, caused by unnecessary scaling. For example, let trees \( T_1, T_2, T_3 \) represent models \( x^2/x^2, x/x, 1 \), for \( x \in [1, 2] \). Whereas the response surfaces of the models are identical, the orders of non-linearity are different.

Fig. 4. Example of non-linearity calculation for a two-variable model. If \( x_1 \in [0, 1] \) and \( x_2 \in [2, 4] \), then "\( x_1 \times x_2 \)" takes values from the interval \([0, 4] \). Non-linearities of the terminal nodes are one. The non-linearity of the \( \text{Sin} \) node is \( 1 + 1 = 2 \). Therefore, the non-linearity of the \( \text{Sin} \) node is two times the degree of the Chebyshev approximation of function \( \sin x \) on the interval \([0, 4] \). If the chosen approximation accuracy is \( 10^{-6} \) then the order of non-linearity of the root node is 2·9 = 18. The expression complexity (visitation length) of the model is 9.

Fig. 5. Genotypic nature of the order of non-linearity. The order of non-linearity depends on the representation, since interval arithmetics is taken into account: Trees \( T_1, T_2, T_3 \) determine models \( x^2/x^2, x/x, 1 \), for \( x \in [1, 2] \). Whereas the response surfaces of the models are identical, the orders of non-linearity are different.
explains why, in our inductive definition, we treat symbolic
variables.

The Chebyshev approximation of degree \( n \) is our two-dimensional version of the Salustowicz function, which we call \( \text{Salustowicz2D} \). The second function originates from [43]; the third equation (13) is usually unknown or very complex, and cannot be expressed in one equation. To demonstrate the order of non-linearity control as a mechanism for preventing over-fitting, we selected a suit of synthetic regression problems, which allowed us to generate reliable noise-free test data for inter- and extrapolation. The target equations for chosen problems are given below:

\[
f_1(x_1, x_2) = \frac{e^{-(x_1-1)^2}}{1.2 + (x_2 - 2.5)^2}
\]

\[
f_2(x) = e^{-x^3} \cos x \sin(x \cos x \sin^2 x - 1)
\]

\[
f_3(x_1, x_2) = f_2(x_1)(x_2 - 5)
\]

\[
f_4(x_1, x_2, x_3, x_4, x_5) = \frac{10}{5 + \sum_{i=1}^{5} (x_i - 3)^2}
\]

\[
f_5(x_1, x_2, x_3) = 30 \left( \frac{(x_1 - 1)(x_3 - 1)}{x_2^3(x_1 - 10)} \right)
\]

\[
f_6(x_1, x_2) = 6 \sin x_1 \cos x_2
\]

\[
f_7(x_1, x_2) = (x_1 - 3)(x_2 - 3) + 2 \sin((x_1 - 4)(x_2 - 4))
\]

\[
f_8(x_1, x_2) = \frac{(x_1 - 3)^4 + (x_2 - 3)^3 - (x_2 - 3)}{(x_2 - 2)^4 + 10}
\]

The choice of several target expressions was inspired by the paper of M. Keijzer [42], which introduced thirteen functions to analyze the performance of scaled GP. We selected the most difficult problems of that set, and still modified most of them to make the regression process more challenging for our ParetoGP system. The first equation defines the Kotanchek function first used in [1]. The second function originates from [43]; we call it the Salustowicz function. The third equation (13) is our two-dimensional version of the Salustowicz function, which we call Salustowicz2D. The function defined in (14) is our favorite problem. This five-dimensional equation, which
we call the \( \text{UBall5D} \) function, was inspired by a simpler two-dimensional problem from [42], [44]. Despite having a simple and harmonious underlying relationship, it appears to be quite difficult for GP. Target expressions for \( \text{RatPol3D} \) (15), \( \text{SineCosine} \) (16), and \( \text{Ripple} \) (17) problems are adopted from [44], with a linear transformation of variables: \( x_i \mapsto x_i - 3 \), and a few other modifications. The \( \text{RatPol2D} \) problem, defined by equation (18), represents another rational polynomial that is challenging for GP. The contour plots of nine target functions (or projections onto 2D intervals for functions (14) and (15)) shown in Figure 8 confirm the high non-linearity of the underlying response surfaces.

The data for the ninth test problem come from an industrial problem on modeling gas chromatography measurements of the composition of a distillation tower. This Tower problem contains 5000 records and 23 potential input variables. The estimated parameter is propylene concentration at the top of the distillation tower. The samples are from a gas chromatograph and are taken every 15 minutes. The 23 potential inputs are temperatures, flows, and pressures related to the distillation tower. The actual sampling rate is one minute, but 15 minutes averages of the inputs are used for model development to synchronize with the output measurement. The measurements (5000 for each variable) are not treated as time series, but simply used as samples for a regression model. The propylene concentration needs to modeled as a function of relevant inputs only. The range of the measured propylene concentration is very broad and covers most of the expected operating conditions in the distillation tower.

B. Experimental Setup

To compare the effects of optimizing different complexity measures in symbolic regression via GP we devised experiments for three optimization schemes:

**CASE I:** Pareto-optimization of the sum of squared errors and expressional complexity;

**CASE II:** Pareto-optimization of the sum of squared errors and the order of non-linearity;

**CASE III:** Pareto-optimization of the sum of squared errors and expressional complexity, alternated with the Pareto-optimization of the sum of squared errors and the order of non-linearity at every generation.

Note that the optimization of the goodness of fit is present in all cases, since constructing accurate models is our first priority. Comparison of CASE I with CASE II will show that creating accurate and 'structurally simple' (i.e., more compact) equations may still lead to highly nonlinear pathological predictions, compared with creating accurate equations of a low or reduced order of non-linearity.

The presence of CASE III experiments is our attempt to blend optimization of the structural complexity and the non-linearity with accuracy optimization in a 'multi-objective' fashion. We are wary about using a composite objective function, which uses a linear combination of objectives of interest, due to its sensitivity to the particular linear coefficients. Instead of limiting the search by using a composite objective function, one should pursue a true multi-objective search and use a vector of objective functions, whose components are optimized individually and simultaneously. However, the multi-objective approach scales badly when the number of objectives increases. We propose a novel heuristic of overcoming the curse of dimensionality of the objective space by alternating two two-objective optimizations of CASE III.

Since the focus of this paper is non-linearity control, the formalization and generalization of the approach of CASE III for a general multi-objective optimization problem would overburden the reader. Instead, we illustrate the idea using a particular case, in which one accuracy measure and two complexity measures are defined, where the priority is on the accuracy.

Three objectives need to be minimized during the model development cycle: prediction error of the model’s phenotype \( \text{(Error} = 1 - \text{NMSE}) \), the expressional complexity of the model’s genotype \( \text{(Complexity}) \), and the order of non-linearity of the model’s genotype \( \text{(Non} - \text{linearity}) \). The order of importance of these objectives is as follows: error minimization is the primary objective, expressional complexity and non-linearity are equally important secondary objectives. Taking this into account, the original multi-objective optimization problem can be substituted by a different one, according to the scheme below:

\[
\min_{\text{all generations}} \left( \text{Error, Complexity, Non} - \text{linearity} \right) \quad (19)
\]

\[
\begin{align*}
\min_{\text{odd generations}} & \left( \text{Error, Complexity} \right) \\
\min_{\text{even generations}} & \left( \text{Error, Non} - \text{linearity} \right)
\end{align*}
\]

\[
(20)
\]

Computational complexity of such substitution drops down from \( O(n^3) \) to \( O(n^2) \), where \( n \) is the total number of models in the population and the archive\(^5\).

The experiments of CASE III are formulated to combine the best properties of solutions of CASE I and CASE II. Accurate models are thus expected to be both compact and 'smooth' (i.e., generalize well), while not producing pathologies in the unseen areas of the input space.

The results of the experiments are compared with respect to the number of pathologies produced on test data with extrapolation, average order of non-linearity and expressional complexity and the area percentages under the convex hulls of archives plotted in expressional complexity versus model error and the order of non-linearity versus model error spaces.

The detailed results of CASES I-III follow immediately after descriptions of the settings for GP parameters and the choice of training and test data.

C. Data sampling and GP settings

As mentioned above, we focused on the synthetic data in this paper so that we could generate a sufficient amount of

\(^5\)We use the non-dominated sorting algorithm to select a fixed number of least-dominated models at the Pareto front to update the archive.
reliable, outlier-free test samples in the regions outside the training regions. The details of sampling procedures used for generation of training and test data are given in Table I.

The Tower problem contains real-life data for which the true input-output relationship is unknown. To assess extrapolative capabilities of GP solutions for the Tower problem, we decided to select significant input variables at a pre-processing step, and then used only those to divide the data into training and test sets. The driving variables identified at initial screening using the fitness inheritance approach (see [9]) were \( x_1, x_4, x_6, x_{12} \) and \( x_{23} \). The 5000 data records corresponding to these inputs were scaled into the five-dimensional cube \([0, 1]^5\). All records belonging to the interval \([0.02, 0.98]^5\) were selected into the training set, and the remaining records formed a test set for extrapolation.

In all experiments of this paper, one optimization measure is always the numerical fitness, determined as a normalized mean-squared error between observed output vector \( y \) and the predicted output vector \( py \):

\[
NMSE(y, py) = \frac{1 - MSE(scale(y), scale(py))}{1 + MSE(scale(y), scale(py))},
\]

\[
MSE(y, py) = \frac{1}{n} \sum_{i=1}^{n} (y_i - py_i)^2,
\]

\[
scale(y) = \frac{y - \min y}{\max y - \min y}
\]

We conducted 50 independent GP runs for each approach and for each problem. All GP settings except for the optimization complexity (expressional, order of non-linearity, or "both, but alternating") are the same for each test problem. The number of generations is fixed to 250 for all problems except SineCosine and UBAll5D problems. These two had to
be modeled over 500 generations in order to get an appropriate goodness of fit. Other parameter setting are given in Table II. Point mutation and balanced crossovers were used as genetic operators. When a crossover is performed, the crossover node in the first parent is selected randomly and uniformly. The level from which this node is sampled dictates the selection of a crossover node in the second parent - the levels of the nodes should be the same or similar. According to our experience, and to [45], balanced crossovers allow reduction in the risk of bloat and also overcome the crossover bias in expressional complexity, introduced by the standard uniform crossover (for the latter, see [36]).

D. Results and Discussion

The solutions of each independent GP run are stored in an archive that contains 50 expressions. These 50 individuals lie at the Pareto front in ‘optimization complexity’ versus ‘model fitness’ objective space containing all individuals evaluated during the current GP run. For our purposes, all of these individuals are equally valuable GP solutions. The ‘customer’, or the domain expert, will have to choose one of these solutions or, better, an ensemble of solutions that satisfies customer needs; see [46]. We therefore combined all archive solutions of independent runs in one ensemble at a post-analysis stage and analyzed the properties of the resulting set of 50×50 = 2500 solutions across different cases and different test problems.

In addition to analyzing the archive solutions, we studied the features of the best-of-the-run solutions, to conform with the standard GP practice, where the elite-preservation strategy based on the multi-objective model selection is not commonly used.

The detailed results appear in Table III. The second- and third columns of this table contain the fraction of equations that showed pathological behaviour on the test data. Column two is a percentage of equations producing infinite or undefined mean-squared error. The latter is computed from the vectors of predicted values of the model, $\text{py}$, and the target values on the test data, $\text{ty}$, as:

$$\text{RMSE}(\text{ty}, \text{py}) = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (\text{ty}_i - \text{py}_i)^2},$$

where $m$ is the number of test points.

Column three is a percentage of solutions for which the root mean-squared error is infinite, undefined or excessively large. The threshold for pathologically high error is chosen to be 100. It corresponds to the mean-squared error equal to $10^4$. Equations producing these large errors on test data are dangerously erroneous, and a high fraction of them in the set of solutions indicates the tendency to over-fitting.

Column four of the table represents the percentage of equations that have the highest allowed order of non-linearity, equal to 10,000. This value indicates highly non-linear behaviour of a model and the potential to have a pathology on unseen data. We expect best-of-the-run solutions to have these high

<table>
<thead>
<tr>
<th>Problem Name</th>
<th>Training Data</th>
<th>Test Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kotanchek Eq (11)</td>
<td>100 points $x_1, x_2 = \text{Rand}(0.3,4)$</td>
<td>2026 points $x_1, x_2 = (-0.2:0.1:4.2)$</td>
</tr>
<tr>
<td>Salutowicz Eq (12)</td>
<td>100 points $x = 0.05:0.1:1.10$</td>
<td>221 points $x = -0.5:0.05:10.5$</td>
</tr>
<tr>
<td>Salutowicz2D Eq (13)</td>
<td>601 points $x_1 = 0.05:0.1:10$, $x_2 = 0.5:2:10.05$</td>
<td>2554 points $x_1 = -0.5:0.05:10.5$, $x_2 = 0.5:0.5:10.5$</td>
</tr>
<tr>
<td>UBall5D Eq (14)</td>
<td>1024 points $x_i = \text{Rand}(0.05,6)$</td>
<td>5000 points $x_1 = \text{Rand}(0.25,6.35)$</td>
</tr>
<tr>
<td>RatPol3D Eq (15)</td>
<td>300 points $x_1, x_2 = \text{Rand}(0.05,2)$</td>
<td>2701 points $x_1, x_2 = (-0.05:0.15:2.1)$, $x_2 = (-0.5:0.5:12.05)$</td>
</tr>
<tr>
<td>SineCosine Eq (16)</td>
<td>30 points $x_1, x_2 = \text{Rand}(0.1,5.9)$</td>
<td>961 points $x_1, x_2 = (-0.05:0.02:6.05)$</td>
</tr>
<tr>
<td>Ripple Eq (17)</td>
<td>300 points $x_1, x_2 = \text{Rand}(0.05,6.05)$</td>
<td>1000 points $x_1, x_2 = \text{Rand}(0.25,6.35)$</td>
</tr>
<tr>
<td>RatPol2D Eq (18)</td>
<td>50 points $x_1, x_2 = \text{Rand}(0.05,6.05)$</td>
<td>1157 points $x_1, x_2 = (-0.25:0.2:6.35)$</td>
</tr>
<tr>
<td>Tower</td>
<td>3136 points all input values in [0.02, 0.098]</td>
<td>1864 points one of the inputs in [0, 0.02] U(0.98, 1]</td>
</tr>
</tbody>
</table>
non-linearity values, due to their inclination to over-fitting. Since the rest of the solutions are expected to have lower non-linearity, small percentages in column four are preferred.

Column five contains the mean order of non-linearity of those solutions that have the non-linearity below the threshold of 10,000. Low values in this column for solutions of CASE I for Salustowicz, Salustowicz2D, UBall5D, and SineCosine problems demonstrate the ‘all or nothing’ phenomenon for the orders of non-linearity of models generated by expression complexity minimization. For example, for CASE I of the Salustowicz problem, we see that 86% of final solutions have the order of non-linearity 10,000, and the remaining 14% have average non-linearity equal to only 12.2. We illustrate such a situation for one GP run in Figure 9.

Column six contains the average expressional complexity of 2500 solutions among independent runs. From columns two to six we observe that CASE I produces, on average, more compact expressions than CASE II (see column six), although, a greater fraction of these have pathologies on test data (see columns two and three). We performed pair-wise statistical significance tests for solutions of CASES I-III, and concluded that CASE II outperforms CASE I with respect to the error of expressions on test data. For significance tests, the solutions with infinite non-linearity were assigned an error value of 100,000. ANOVA tests and Wilcoxon-Mann-Whitney rank sum tests were performed to compare the means and the medians of different sets of error values of the equal number of samples. Table IV represents the p-values for the 95% significance level. The mean and the median error on test data for the solutions of different sets of error values of the equal number of samples.
CASE II were significantly smaller than those of the solutions of CASE I (the maximum p-value is 0.0004 for the Wilcoxon test on the RatPol3D problem).

The smoothness of solutions of CASE II comes at the expense of higher expressional complexity. Solutions, generated in CASE II consist, for the most part, of ‘simple’ operators (such as addition, subtraction, multiplication), but are bulky and sometimes difficult to interpret. This excessive growth in structure disappears when the CASE III experiment is used. Comparing the results of CASE II and CASE III in Table III, we observe that the average expressional complexity of solutions can be reduced in CASE III (column six), however, with a side effect of an increased pathology rate (columns two and three). The significance tests show that CASE II significantly outperforms CASE III in the error on the test set on eight out of nine test problems: Kotanchek, UBall5D (only for the mean error), RatPol3D, RatPol2D, Tower, and SineCosine and Ripple (only for the median error).

In a comparison of CASE III with CASE I, the tests show that the errors on the test data produced by solutions of CASE III are significantly smaller than those of CASE I in all test problems (see columns two and three of Table IV). This brings us to the first important conclusion: solutions obtained in CASE III with alternating expressional complexity and the order of non-linearity, as well as solutions of CASE II with non-linearity minimization, are significantly smoother than the ones of CASE I with minimization of expressional complexity. The fact that CASE III produces solutions competitive with CASE I (with respect to the error) and CASE II (with respect to the expressional complexity) is counter-intuitive and rather surprising. The alternation of optimization complexities at each generation is a very crude heuristic aimed at producing both compact and smooth equations. The fact that it works motivates us to explore the scalability of this approach to cases in which a multitude of objectives needs to be satisfied.

The second part of Table III shows the results for the 50 best-of-the-run solutions for each case. The median and the interquartile range of the root mean-squared error over 50 best-of-the-run solutions per experiment are given in columns seven and eight of Table III. The resulting values look similar for CASES I-III, and no clear trends can be observed with respect to the superiority of any one approach on the training data.

The difference among solutions of CASES I-III becomes obvious when the best-of-the-run equations are evaluated on the test sets. Column nine of Table III represents the fraction of best-of-the-run equations that have a pathology on the test data, defined as an infinite root mean-squared error. The trend is similar to the one revealed in columns two and three: CASE I has the highest rate of pathologies at extrapolation. The only exception in this rule is the Kotanchek problem, with 24%, 28%, and 34% pathological equations from the 50 best-of-the-run equations.

Columns 10 and 11 of Table III contain the median error and the interquartile range of a set of errors that are smaller than 100. For example, for the solutions of CASE I of the Kotanchek problem we observe that 52% of best-of-the-run solutions (26 equations out of 50) have a pathology on the test data. If those and also other equations producing errors higher than 100 are removed from the sample, then the median of the remaining 24 equations will be 0.075, and the interquartile range will be 0.06. This is an argument for using archives of equations and for being very cautious in using best-of-run solutions. If only best-of-the-run solutions are sought for, then all runs where the best equation produces a pathology are lost. This corresponds to an incredible waste of 52% of the spent effort in the example of CASE I solutions of the Kotanchek.
TABLE IV
SIGNIFICANCE OF CONCLUSIONS ABOUT THE RESULTS OF THE THREE EXPERIMENTS. WE PERFORMED ONE-WAY ANOVA TESTS AND
WILCOXON-MANN-WHITNEY TESTS FOR ANALYZING DIFFERENCES IN THE MEANS AND THE MEDIAN VALUES OF SOLUTIONS OF
CASES I-III. THE P-VALUES FOR ANOVA TESTS ARE OBTAINED FROM THE F-STATISTICS AT THE 95% CONFIDENCE LEVEL (ANOVA). P-VALUES FOR
WILCOXON-MANN-WHITNEY TESTS ARE OBTAINED FROM THE Z-STATISTICS AT THE 95% CONFIDENCE LEVEL AND ARE DUBBED FOR TWO-SIDED
TESTS (WILCOXON). TO OBTAIN EQUAL SAMPLE SIZES, WE TRUNCATED ALL INFINITE AND EXCESSIVELY LARGE VALUES OF RMSE (THOSE WHERE
RMSE$>= 100$) ON THE TEST DATA AT 100.

<table>
<thead>
<tr>
<th>Problem and Hypothesis</th>
<th>Comparing the Error (RMSE) on Test data of All Solutions (2500)</th>
<th>Comparing the Average Area Percentage</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ANOVA, Wilcoxon, p-value</td>
<td>ANOVA, Wilcoxon, p-value</td>
<td>ANOVA, Wilcoxon, p-value</td>
</tr>
<tr>
<td><strong>Kotanchev</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0003</td>
</tr>
<tr>
<td>CASE II outperforms CASE III</td>
<td>9.03E-07</td>
<td>0.0000</td>
<td>0.008</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>6E-33</td>
<td>0.31</td>
</tr>
<tr>
<td><strong>Salustowicz</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.3312</td>
</tr>
<tr>
<td>CASE III outperforms CASE II</td>
<td>0</td>
<td>0</td>
<td>0.5006</td>
</tr>
<tr>
<td>CASE III outperforms CASE III</td>
<td>0</td>
<td>0</td>
<td>0.6971</td>
</tr>
<tr>
<td><strong>Salustowicz2D</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0184</td>
</tr>
<tr>
<td>CASE II outperforms CASE III</td>
<td>0.0462</td>
<td>0.0055</td>
<td>0.0148</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.9295</td>
</tr>
<tr>
<td><strong>UBall5D</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0328</td>
</tr>
<tr>
<td>CASE II outperforms CASE III</td>
<td>0</td>
<td>0.8655</td>
<td>0.5058</td>
</tr>
<tr>
<td>CASE III outperforms CASE II</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0047</td>
</tr>
<tr>
<td><strong>RatPol3D</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0.0004</td>
<td>0.0287</td>
</tr>
<tr>
<td>CASE II outperforms CASE II</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CASE II outperforms CASE III</td>
<td>0</td>
<td>0</td>
<td>0.0503</td>
</tr>
<tr>
<td>CASE III outperforms CASE II</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.8117</td>
</tr>
<tr>
<td><strong>SineCosine</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0006</td>
</tr>
<tr>
<td>CASE II outperforms CASE II</td>
<td>0.1288</td>
<td>0</td>
<td>0.9963</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.9847</td>
</tr>
<tr>
<td><strong>Ripple</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0006</td>
</tr>
<tr>
<td>CASE II outperforms CASE III</td>
<td>0.3902</td>
<td>0</td>
<td>0.1318</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0501</td>
</tr>
<tr>
<td><strong>RatPol2D</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.0004</td>
</tr>
<tr>
<td>CASE II outperforms CASE III</td>
<td>0</td>
<td>0</td>
<td>0.0305</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.1707</td>
</tr>
<tr>
<td><strong>Tower</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CASE II outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CASE II outperforms CASE II</td>
<td>0</td>
<td>0</td>
<td>0.2921</td>
</tr>
<tr>
<td>CASE III outperforms CASE II</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CASE III outperforms CASE I</td>
<td>0</td>
<td>0</td>
<td>0.5648</td>
</tr>
</tbody>
</table>

The significance tests for the errors of best-of-the-run equations are performed in a style similar to that used for all solutions. We first assign an error value of 100 to 100 equations producing undefined and infinite values, as well as those exceeding 100. We then perform the pair-wise ANOVA and Wilcoxon tests to determine the significance in the difference of the mean and median errors among 50 solutions of CASES I-III. The p-values of the tests are given in columns three and four of Table IV. We can observe that CASE II significantly outperforms CASE I with respect to best-of-the-run errors on test data on seven out of nine problems (for **UBall5D** only for the mean error).

CASE III significantly outperforms CASE I on best-of-the-run errors only for the mean error on the **UBall5D** problem. There is no significant difference in the error samples of CASE III and CASE I for the rest of the problems. The second important conclusion that we can make for CASE III is that it is nowhere significantly worse than CASE I – even on best-of-the-run solutions.

The average values of the order of non-linearity and expressional complexity of the best-of-the-run equations are given in columns 12 and 13 of Table III. The conclusions on the complexity of best-of-the-run solutions are the same: 1) CASE I produces more compact expressions at the expense of high orders of non-linearity; 2) CASE II produces solutions with lower non-linearity, but higher expressional complexity; 3) CASE III produces lower orders of non-linearity than CASE
I does, and lower expressional complexity than CASE II.

E. Analysis of the Evolved Programs

Note that the differences in the expressional complexity of best-of-the-run equations for CASES I-III are not big. As an example, we give five best-of-the-run solutions for each CASE on the Salustowicz problem in Table V. All equations in Table V are simplified in Mathematica, so the solutions of CASE II appear to be short (since the linear operations on constants and input variables are already executed). The purpose of Table V is to provide the reader with a visual impression of the differences primarily between CASE I and CASE II solutions and to support our claim that shorter equations may be less convincing for an engineer than will be longer but less non-linear equations. Formulae of solutions of CASE I in Table V illustrate that opting for shorter equations generates many nested functions that may make no physical sense.

F. Further Discussion: Areas under Pareto Fronts

To conclude the analysis of performance of CASES I-III, we compared the average areas under the convex hulls of the archive at the last generation. For two optimization objectives, a good measure to assess the quality of the approach is the archive at the last generation. For two optimization objectives, we compared the average areas under the convex hulls of the archive, computed in the expressional complexity versus error and in the order of non-linearity versus error objective spaces.

We computed the average area percentages of two objective spaces for independent runs of each experiment, and performed the multiple comparison tests for significance differences in the mean values for CASES I-III (ANOVA tests at the 95% confidence level). The results of these comparisons are plotted in Figure 11.

Columns five and six of Table IV report the p-values for the pair-wise comparisons of the average area percentage with the ANOVA tests and the Wilcoxon tests. The conclusion from these statistical tests: CASE III is statistically better than CASE I on three out of nine problems (Salustowicz, Salustowicz2D, UBall5D) and is not statistically different from CASE I on the rest of the problems. CASE III is better than CASE II on three problems (UBall5D, RatPol3D, Tower), and is statistically the same for the rest of the problems. These results make CASE III the winner in the comparison for producing the least average percentage area under the convex hull of the archive in two objective spaces.

V. Conclusions

This paper has introduced a novel complexity measure for creating smoother individuals in symbolic regression via genetic programming. We suggest computing the order of non-linearity iteratively for genotypes of symbolic models according to a set of rules (A)-(G). The notion of the new measure is based on a degree of Chebyshev polynomial approximation of a certain accuracy.

We demonstrate the positive effects of controlling the order of non-linearity on nine non-linear test problems, and indicate that a similar gain in extrapolative capabilities of symbolic models is obtained on other problems from industrial applications (see [1], [9], [46]).

One of the weaknesses of the order of non-linearity is an over-estimation of the true minimal degree of Chebyshev approximation of accuracy $\epsilon$ for unary functions and the approximate nature of the definition for functions of multiple arguments. Even for functions of two arguments, constructing a Chebyshev approximation is performed in terms of tensor products, and represents a non-trivial computational procedure. For functions of more variables it is difficult to construct the Chebyshev polynomial approximation of a given accuracy, thus making it difficult to compare the order of non-linearity with the degree of such an approximation.

The presented order of non-linearity applied as a second optimization objective in combination with numerical accuracy to symbolic regression via genetic programming favors models with smoother response surfaces. On all nine test problems,
these models show significantly better extrapolative capabilities over models generated with controlled expression complexity.

We observed that models generated with minimization of the order of non-linearity (CASE II experiments) are less compact than those generated via optimization of expression complexity (CASE I experiments). To combine the benefits of creating compact expressions with smoother response surfaces, we have proposed a new hybrid approach to symbolic regression: Pareto-optimization of the goodness of fit and expression complexity, alternated with the Pareto-optimization of the goodness of fit and the order of non-linearity at every generation (CASE III experiments).

The vast majority of models obtained with the order of non-linearity control (CASE II and CASE III) exhibit ‘graceful degradation’ by extrapolation. This corresponds to an intuitive expectation that smoother approximations mimic the original output longer when extrapolated outside the training range.

Models generated with non-linearity control do not get singularities when extrapolated over reasonable distances. The fundamental question regarding the way in which to obtain a reliable prediction of the output on an extrapolated domain remains a subject for further research.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for extremely valuable comments and enlightening questions that have not only helped us to improve the presentation of the
results and the convincingness of our experiments, but in general, made our research on this subject more mature.

The first author would also like to thank Dr. Stef van Eijndhoven from Eindhoven University of Technology for connecting her to Dow Benelux, Dr. Arthur Kordon from the Dow Chemical Company for the Tower data set, Dr. Mark Kotanchek from Evolved Analytics LLC, Dr. Randy Pell from the Dow Chemical Company, and Dr. Arthur Kordon for helpful comments on the earlier versions of this paper.

This research was supported by Tilburg University and Dow Benelux B.V.

REFERENCES


