Virtual Sensors for Emissions of a Diesel Engine Produced by Evolutionary System Identification

Stephan M. Winkler[1], Markus Hirsch[2], Michael Affenzeller[1], Luigi del Re[3], and Stefan Wagner[1]

[1] Heuristic and Evolutionary Algorithms Laboratory
Upper Austria University of Applied Sciences
School of Informatics, Communications and Media
Softwarepark 11, 4232 Hagenberg, Austria
{stephan.winkler,michael.affenzeller,stefan.wagner}@fh-hagenberg.at

[2] Linz Center of Mechatronics, Altenbergerstraße 69, 4040 Linz, Austria
markus.hirsch@lcm.at

[3] Institute for Design and Control of Mechatronical Systems
Johannes Kepler University Linz, Altenbergerstraße 69, 4040 Linz, Austria
luigi.delre@jku.at

Abstract. In this paper we discuss the generation of models for emissions of a Diesel engine, produced by genetic programming based evolutionary system identification: Models for the formation of NO_x and particulate matter emissions are identified and analyzed. We compare these models to models designed by experts applying variables section and the identification of local polynomial models; analyzing the results summarized in the empirical part of this paper we see that the use of enhanced genetic programming yields models for emissions that are valid not only in certain parts of the parameter space but can be used as global virtual sensors.

1 Introduction and Experimental Data

Virtual sensors are in general simulation models that can be used instead of physical sensors. If there are no appropriate models available to the required precision, virtual sensor design must be based on data; we are in this context speaking of data based system identification [6]. In this paper we concentrate on the discussion of models for emissions of a common rail direct injection 2 liter 4 cylinder production Diesel engine, produced by local polynomial modeling as well as genetic programming based evolutionary system identification; models for the formation of nitrogen oxides (NO_x) and particulate matter (PM), the two most demanding emissions of Diesel engines, are identified and analyzed.

In Section 2 we summarize the identification methods that have been applied for identifying emission models for the investigated engine, namely local in parameter linear regression using a restricted set of input parameters (as described in Section 2.1) and enhanced evolutionary system identification [9] based on genetic programming (GP, see [5], e.g.) as discussed in Section 2.2. The results of

these empirical test studies are summarized in Section 3; a discussion of these results (given in Section 4) concludes this paper.

We have used data recorded at a dynamic engine test bench at the Institute for Design and Control of Mechatronical Systems at Johannes Kepler University (JKU) Linz, Austria. The data was composed of selected engine states relevant for defining the combustion process, given as measured or calculated quantities of the engine control unit (ECU), as well as target values for NO_x and PM emissions which have been measured with fast emission sensors. Seven different measurements with a sampling time of 50 ms have been recorded:

- Six measurements $M_{1...6}$ (each recorded over approximately 10–15 minutes) have been recorded for different engine speeds and amounts of injected fuel. These measurements were done as a result of optimal design of experiment strategies for achieving short but well exciting signals for nonlinear identification (for details on this see [4]). The average engine speed (N) was set to 1000 and 2000 revolutions per minute, and the average amount of total injected fuel (q) to 5, 10 and 20 mg per cycle; thus, combining these parameter settings we get 6 conditions under which the engine has been tested. Figure 1 shows a visualization of the parameters N and q of the samples that are included in these six sets of measurements.
- The engine was also tested following the New European Driving Cycle (NEDC), a standardized driving cycle which is used for evaluating the fulfillment of emission standards for passenger cars. The NEDC data ($NEDC$) are within the operating range covered by the measurements $M_{1...6}$.

Based on these data sets our goal is to identify models for NO_x and PM emissions using the data sets $M_{1...6}$ and testing these models on $NEDC$ data.

2 Identification Methods Used for Designing Virtual Sensors

2.1 Variables Selection and Local Polynomial Regression

Given a data collection including m input features storing the information about n samples, a in parameter linear model is defined by the vector of coefficients $\theta_{1...m}$. For calculating the vector of modeled values $Y \in \mathbb{R}^n$ using the given input values matrix $U \in \mathbb{R}^{m \times n}$, these input values are multiplied with the corresponding coefficients and added: $Y = U \cdot \theta$; the coefficients vector θ can be computed by standard least square error minimization according to $\theta = \left(U^T U\right)^{-1} U^T Y$. Theoretical background of this approach can be found in [6].

For identifying models for NO_x and PM emissions in the context of our research project we have restricted the set of input features to the following four variables: The total amount of injected fuel per cycle (q), the engine's speed (N), the manifold air pressure (MAP), and the concentration of oxygen in the exhaust (O_2exh). This selection of input variables, all available in the ECU, has been made on the basis of knowledge about physical knowledge in combustion

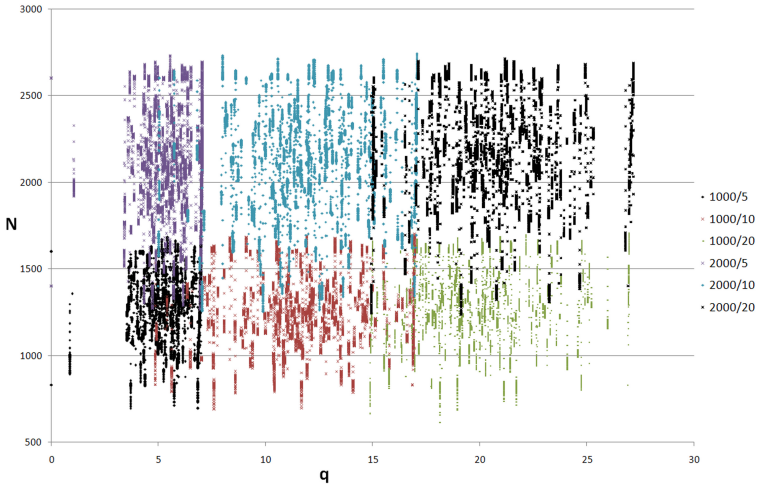

Fig. 1. Visualization of engine parameter values of the training measurements: Each spot represents the engine speed (N) and the amount of injected fuel (q) of one data sample. Each sample belongs to one of the six available sets of measurements; spots representing samples belonging to the same set are displayed using the same visualization style (as indicated in the legend of the chart).

engines; further information can for example be found in [3] and [8]. As the process of emission formation is a nonlinear one, even in the six restricted data sets, second order multiplications of these input features are also used. Though the formulation now includes nonlinearities, the structure is linear in parameters and we are looking for is a set of (in total 30) parameters $\theta_{i,j}$ so that NO_x and PM can be described as $NO_x(t) = \theta_{1,1} + \theta_{1,2} \cdot q(t) + \theta_{1,3} \cdot N(t) + \theta_{1,4} \cdot MAP(t) + \theta_{1,5} \cdot O_2exh(t) + \theta_{1,6} \cdot q(t)^2 + \theta_{1,7} \cdot q(t) \cdot N(t) + \ \ldots \ + \theta_{1,15} \cdot O_2exh(t)^2$ and $PM(t) = \theta_{2,1} + \theta_{2,2} \cdot q(t) + \theta_{2,3} \cdot N(t) + \theta_{2,4} \cdot MAP(t) + \theta_{2,5} \cdot O_2exh(t) + \theta_{2,6} \cdot q(t)^2 + \theta_{2,7} \cdot q(t) \cdot N(t) + \ \ldots \ + \theta_{2,15} \cdot O_2exh(t)^2$, respectively.

The fact that static models (present output is defined only by present input values without time offsets) are used for representing a dynamic system can be reasoned by the fact that not only directly settable input quantities such as $q(t)$ but also internal engine states such as $MAP(t)$ or $O_2exh(t)$, which already contain dynamic effects, have been used here.

2.2 Evolutionary System Identification

Basically, genetic programming (GP) is based on the theory of genetic algorithms (GAs) and utilizes a population of solution candidates which evolve through many generations towards a solution using certain evolutionary operators and a selection scheme increasing better solutions' probability of passing on genetic information; the goal of a GP process is to produce a computer program solving

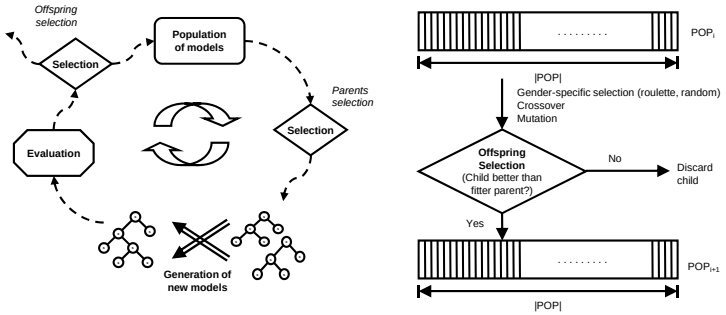

Fig. 2. Left: The extended genetic programming cycle including offspring selection. **Right:** Strict offspring selection as used here within the GP process.

the optimization problem at hand. In the case of structure identification, solution candidates represent mathematical models; these models are applied to the given training data and the so generated output values are compared to the original target data. The left part of Figure 2 visualizes how the GP cycle works: As in every evolutionary process, new individuals (in GP's case, new programs) are created and tested, and the fitter ones in the population succeed in creating children of their own; unfit ones die and are removed from the population [5].

Within the last years we have set up an enhanced and problem domain independent GP based structure identification framework that has been successfully used in the context of various different kinds of identification problems for example in mechatronics, medical data analysis, and the analysis of steel production processes; please see [9] for an extensive overview the authors' research activities in these fields. One of the most important problem independent concepts used in our implementation of GP-based structure identification is offspring selection [1], an enhanced selection model that has enabled genetic algorithms and genetic programming implementations to produce superior results for various kinds of optimization problems. As in the case of conventional GAs or GP, offspring are generated by parent selection, crossover, and mutation. In a second (offspring) selection step (as it is used in our GP implementation), only those children become members of the next generation's population that outperform their own parents; the algorithm repeats the process of creating new children until the number of successful offspring is sufficient to create the next generation's population. In [2] and [9] interested readers can find several examples and analyses of the effects of OS in GAs.

In our research project in the context of the identification of virtual sensors we have used all available (or rather "allowed") variables, i.e., not only those 4 variables that have already been described (q, N, MAP, and O_2exh), but also several other ones (in total 31 variables) including temperatures and parameters of the ECU. These variables are allowed in this context simply because these values are also available in the context of standard commercial automobiles; information about other emissions (as for example CO_2) has not been used.

3 Results Documentation: Analysis of Identified Models

3.1 Using in Parameter Linear Models

The approach applied using in parameter linear regression models for representing local behaviors was the following one: All six available measurement data sets $M_{1...6}$ have been used for identifying 2^{nd} order polynomial models using MATLAB®. When estimating the target values for test data X all models $m_{1...6}$ are applied and the resulting target value for sample i ($t(i)$) is calculated using the following procedure for smooth switching:

$$
\begin{aligned}
x_k &:= \texttt{apply}(m_k, X(i))(\forall k \in [1 \dots 6]) \\
c &:= (N - 1400)/200) \quad (c < 0 \Rightarrow c := 0; c > 1 \Rightarrow c := 1) \\
y_1 &:= x_1 \cdot c + x_2 \cdot (1 - c) \\
y_2 &:= x_3 \cdot c + x_4 \cdot (1 - c) \\
y_3 &:= x_5 \cdot c + x_6 \cdot (1 - c) \\
c &:= (q - 4)/2) \quad (c < 0 \Rightarrow c := 0; c > 1 \Rightarrow c := 1) \\
y_4 &:= y_2 \cdot c + y_3 \cdot (1 - c) \\
c &:= (q - 14)/2) \quad (c < 0 \Rightarrow c := 0; c > 1 \Rightarrow c := 1) \\
t(i) &:= y_1 \cdot c + y_4 \cdot (1 - c)
\end{aligned}
$$

This procedure is of course applied for estimating NO_x as well as PM emissions (separately); for estimating NO_x values we use the models identified using NO_x training data (${m^{NO_x}}_{1...6}$) as prediction models $m_{1...6}$, and for estimating PM values we use the prediction models (${m^{PM}}_{1...6}$).

We here give the qualities of the so resulting combined estimation models as the estimated values' mean squared error (mse) on NEDC data: The mse of the estimated NO_x values for the NEDC data set is $\mathbf{1.4 \cdot 10^4}$, for the particulate matter emissions (which are given in terms of opacity) the mse is **2.635**.

3.2 Using Evolutionary System Identification

Using GP we have trained models on the basis of all six data sets $M_{1...6}$, i.e., we have collected all samples in one big training data collection and applied enhanced GP using strict OS for learning models; the GP implementation in HeuristicLab (as described for example in [7]) has been used as described for example in [9]. We have not defined any model structures that are fixed for the NO_x and PM identification tasks, all available arithmetic and logical functions (as given in Table 1 and discussed in detail in [9]) have been used; the maximum hierarchy depth of the produced formulas has been set to 12. As the reader can see in Table 1, mathematical functions and terminal nodes are used as well as Boolean operators for building complex arithmetic expressions. There are in fact no structural restrictions for the use of Boolean blocks in formulae; of course, [Then/Else] and Boolean expressions have to be connected to [IF] nodes, but there are no other restrictions regarding the use of Boolean blocks within mathematical expressions.

Table 1. Set of function and terminal definitions for GP based system identification.

Functions		
Name	**Arity**	**Description**
+	2	Addition
-	2	Subtraction
*	2	Multiplication
/	2	Division
e^x	1	Exponential Function
IF	3	If [Arg0] then return [Then] branch ([Arg1]), otherwise return [Else] branch ([Arg2])
$\leq$, $\geq$	2	Less or equal, greater or equal
&&, ||	2	Logical AND, logical OR
Terminals		
Name	**Parameters**	**Description**
var	x, c	Value of attribute x multiplied with coefficient c
const	d	A constant double value d

The population size was set to 1000, we applied standard one-point crossover and mutation operators (the mutation rate was set to 15%), and the maximum selection pressure (defining the algorithms' termination criterion) was set to 500. We have applied static modeling (i.e., no time offsets were allowed for model inputs – as also done in the case of constrained polynomial modeling). The GP approach has been tested 5 times independently; the quality of these models is estimated by testing them on the NEDC data set. Using this static GP approach we retrieved models for NO_x that show an average test mse of **9351.1742** ($\sigma = 1624.25$); for PM we retrieved models with average test mse of **1.8455** ($\sigma = 0.1418$).

3.3 Discussion

Figure 3 shows the evaluation of the models for NO_x (produced by polynomial regression as well as GP) that performed best on training data, evaluated on a part of the given test (NEDC) data; Figure 4 shows the residuals of these models when evaluated on this part of the NEDC data. The mean squared error of the model produced by GP (when evaluated on test data) is **7847.43**. As we can see in these figures, there are significant errors that occur when the amount of injected fuel becomes very low (or even zero). This is because the data for the identification of this area was not available in sufficient detail, and therefore we here extrapolate by applying the models to parameter regions not available in the training data. Very high local peaks of the regression model can be explained: Small delays, not important for slow changes, given by the geometrical location of the lambda sensor in the exhaust, which is used for measuring O_2exh, cause a mismatch of the inputs which causes the high peaks. Slightly filtering q could reduce this effect.

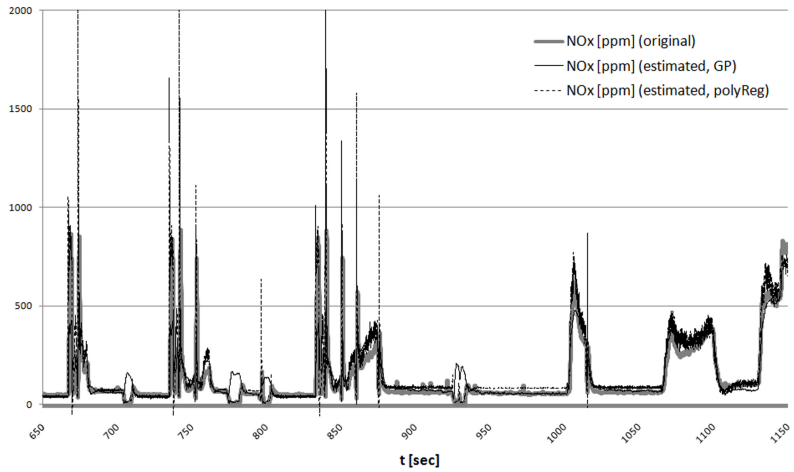

Fig. 3. Evaluation (on a part of the NEDC data) of models produced by GP and polynomial regression for the engine's NO_x emissions.

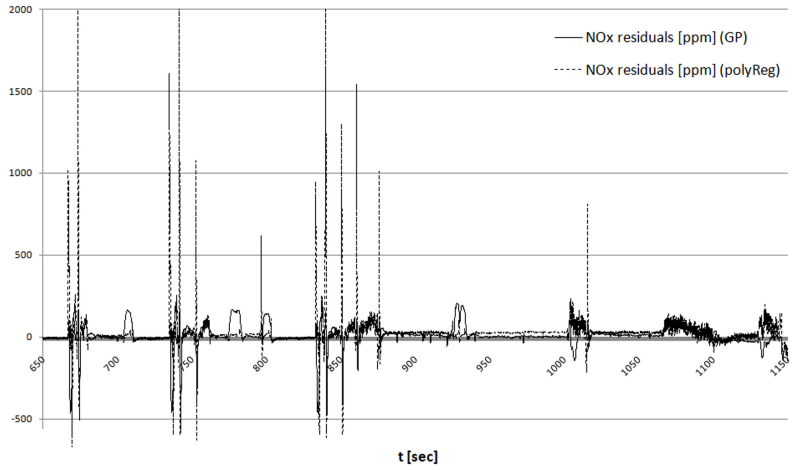

Fig. 4. Residuals (on a part of the NEDC data) of models produced by GP and polynomial regression for the engine's NO_x emissions.

4 Conclusion

Differently from classical optimization methods, genetic algorithms address the optimization issue in a biologically inspired way trying to recognize dominant "parts of the solution" and build the final result around these parts. Conceptually, this holds also for genetic programming: GP is able to provide analytical

expressions, a frequent wish of designers, especially if these can be interpreted physically. In the course of the project which is the background of the condensed information presented in this paper, GP based structure identification has been compared to a much better established method, the NARX polynomial modelling approach, and has shown to be a viable alternative, with validation results absolutely comparable, if not superior. Differently from classes like the polynomial ARX models, GP is able to build new function kernels which can allow a much better insight into the system. Of course, as for every heuristic method, there is no guarantee for it and the computational effort can become rather large, but it can provide a very interesting and probably unique approach to the system model, in this case to the emissions.
Another important discovery of this work was that, at the end, the choice of the data has a paramount importance, much more than in the linear case.

5 Acknowledgements

The work described in this paper was done within the Translational Research Program project L284-N04 "GP-Based Techniques for the Design of Virtual Sensors" sponsored by the Austrian Science Fund (FWF). The involved research organizations are the Heuristic and Evolutionary Algorithms Laboratory at the Upper Austria University of Applied Sciences, Faculty of Informatics, Communications and Media, and the Linz Center of Mechatronics.

References

1. Michael Affenzeller, Stefan Wagner, and Stephan Winkler. Goal-oriented preservation of essential genetic information by offspring selection. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2005*, volume 2, pages 1595–1596. Association for Computing Machinery (ACM), 2005.
2. Michael Affenzeller, Stephan Winkler, Stefan Wagner, and Andreas Beham. *Genetic Algorithms and Genetic Programming – Modern Concepts and Practical Applications*. Chapman & Hall/CRC, 2008.
3. Markus Hirsch, Daniel Alberer, and Luigi del Re. Grey-box control oriented emissions models. In *Proceedings of IFAC World Congress 2008*, pages 8514–8519, 2008.
4. Markus Hirsch and Luigi del Re. Adaped D-optimal experimental design for transient emission models of diesel engines. In *Proceedings of SAE Congress 2009*, 2009.
5. William B. Langdon and Riccardo Poli. *Foundations of Genetic Programming*. Springer Verlag, Berlin Heidelberg New York, 2002.
6. Lennart Ljung. *System Identification – Theory For the User, 2nd edition*. PTR Prentice Hall, Upper Saddle River, N.J., 1999.
7. Stefan Wagner. *Heuristic Optimization Software Systems – Modeling of Heuristic Optimization Algorithms in the HeuristicLab Software Environment*. PhD thesis, Johannes Kepler University Linz, 2009.
8. Jürgen Warnatz, Ulrich Maas, and Robert W. Dibble. *Combustion - Physical and Chemical Fundamentals, Modeling and Simulation, Experiments, Pollutant Formation*. Springer-Verlag, Heidelberg, 1996.
9. Stephan Winkler. *Evolutionary System Identification - Modern Concepts and Practical Applications*. PhD thesis, Johannes Kepler University Linz, 2008.