Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Many-Objective Optimization of a Hybrid Car Controller

Tobias Rodemann, Kaname Narukawa, Michael Fischer, Mohammed Awada

2015

Preprint:

This is an accepted article published in EvoStar 2015 Conference. The final authenticated version is available online at: https://doi.org/[DOI not available]

Many-Objective Optimization of a Hybrid Car Controller

Tobias Rodemann, Kaname Narukawa, Michael Fischer, and Mohammed Awada

[1] Tobias Rodemann and Kaname Narukawa**, Honda Research Institute Europe
Carl-Legien-Strasse 30
63073 Offenbach/ Main
Germany
www.honda-ri.de
tobias.rodemann@honda-ri.de,kaname_narukawa@n.f.rd.honda.co.jp
[2] Michael Fischer and Mohammed Awada, Honda R&D Europe Germany
Carl-Legien-Strasse 30
63073 Offenbach/ Main
Germany
[michael_fischer,mohammed_awada]@honda.co.uk

Abstract. Hybrid cars are a promising approach for providing individual mobility with lower CO_2-emissions and without compromising on affordability and driving range. In order to reach these targets a highly efficient control (energy management) is required. In mass production vehicles control is often organized using simple, quick, and easy to understand rule-based systems. Such a rule-base typically contains a moderate number of parameters which can be tuned using methods like evolutionary algorithms to improve performance. However, prior work basically targets a minimization of fuel consumption. In this work we present a many-objective evolutionary optimization that considers up to 7 objectives in parallel. We outline the additional optimization challenges that arise due to the large number of objectives and demonstrate that a substantial performance increase, over all objectives, can be achieved.

Keywords: Many-Objective Optimization, Hybrid Cars, Data Mining

1 Introduction

In order to optimally drive a certain, pre-defined driving cycle using a hybrid car a sophisticated controller is required. A hybrid powertrain allows to provide power partially or even completely using only the electric motor, employing the combustion engine only for some parts of the route or driving cycle. The controller has to decide if the current power demand is provided by combustion engine or electric motor, or, from a different perspective, using fuel or battery energy. This problem can not be solved statically since it depends on the current

** K. Narukawa's new affiliation is Honda R&D Ltd., Japan.

and future driving situations. Depending on the definition of optimality (e.g. minimum fuel consumption or CO_2-emission), the optimal power split might depend on the current speed of the car (and the resulting efficiency of the combustion engine) or the remaining state of charge (SOC) of the battery. A typical problem is that a locally (at the current point in time) good control might later turn out to be bad, like for example when a car with an empty battery has to go uphill, leading to an unpleasant driving experience (low acceleration) and operation of the combustion engine in low efficiency regimes. A smarter (foresighted) strategy would have kept enough energy in the battery to master the climb, allowing a battery recharge in the subsequent down-hill phase.

A simple approach for a controller in a hybrid car is using a set of basic rules that decide on the current power split based on a few system inputs, like current speed and SOC. These controllers are easy to implement and validate. They are also simple enough to be computed at high frequencies even on very small on-board ECUs. Unfortunately, these controllers are often far away from an optimal control in terms of fuel consumption and drivability.

To reduce the gap to an optimal controller various optimization methods can be used to tune parameters in the rule-base, in order to provide a better performance on one or more relevant driving cycles.

A well-tested set of methods is provided through evolutionary algorithms. The computation of the fitness is typically done using a simulation model of the hybrid car with a defined driving cycle like the NEDC (New European Driving Cycle). Evolutionary algorithms (EAs) have a relatively high (offline) computation demand due to the semi-random search process, but are very flexible, i.e. they can quickly be adapted to handle new constraints and objectives, and require less domain knowledge on the optimization side. Their usefulness has been shown in several articles [1, 5].

Prior work has mostly focused on a single objective, often fuel consumption, but the resulting operation strategies often lack in many other criteria. Some of the additional criteria are directly noticeable by the customer, like engine noise, others, like battery lifetime, are more important to the manufacturer.

In recent years several state-of-art MAny-Objective optimization (MAO) evolutionary algorithms have been proposed, which allow optimization for a larger number of objectives.

In this work we demonstrate the usage of a many-objective optimization algorithm, the SMS-EMOA [2], for tuning a rule-based controller for a hybrid car towards 7 different objectives. In a previous work [9] we have compared different optimization algorithms without giving too much detail about the specific application. In this paper our focus is more on the relation between many-objective optimization and the specific application of a hybrid car controller.

The following sections first give a brief outline of hybrid car technology as far as it concerns the optimization problem and the employed simulation system, then we describe in more detail the applied optimization algorithm. In the results section we show some of the found solutions and compare this to a manually tuned system. The summary and outlook section wraps up the work presented

and sketches some of the remaining issues, especially the post-processing and visualization of the optimization results.

2 Hybrid cars

A Hybrid Electric Vehicle (HEV) combines a standard internal combustion engine linked to a conventional fuel tank with an electric motor that can turn battery energy into wheel torque or act as a generator that recuperates energy. The higher cost and weight of hybrid vehicles through the additional motors, battery, and other parts is in theory compensated by a number of advantages:

- Higher mean operation efficiency of combustion engine resulting in lower fuel consumption
- Energy recuperation during braking and down-hill phases
- Better acceleration performance when using both engine and motor
- Reduced noise and emissions when engine is off

These advantages depend on the specific type of hybrid architecture and an efficient controller. Emission and noise benefits matter most in urban driving situations, and a wrong current SOC level (too high or too low) will reduce the benefits for engine operation, acceleration or recuperation. It is therefore essential to have a good energy management to make optimum use of the hybrid power supply.

In this work we will study a model car that is a plug-in serial hybrid with a battery capacity of 24.4 kWh (corresponding to approximately 150 km of pure electric driving range).

Typical driving cycles that are used to assess fuel consumption are rather short and could be managed fully without combustion engine. We are therefore using our own driving cycle, which is a combination of urban and extra-urban driving conditions. It is shown in Fig. 1. Its total length is 230 km, beyond the purely electric range. Note that in the following, remaining battery energy at the end of the driving cycle (above the 20% minimum charge level) is translated into (reduced) fuel consumption for the final assessment of the driving strategy. For this we assumed that 1 kWh of remaining battery energy corresponds to 0.5 l of fuel.

2.1 Objectives

As outlined in the introduction there are several objectives besides fuel consumption that influence the perception of a car's performance by the customer. Many of these criteria are difficult to define and there is rarely a hard constraint that can be set. In many cases, the different objectives are conflicting, so that a trade-off has to be found. It also turned out to be difficult to put relative weights to these objectives, since they are from rather different domains.

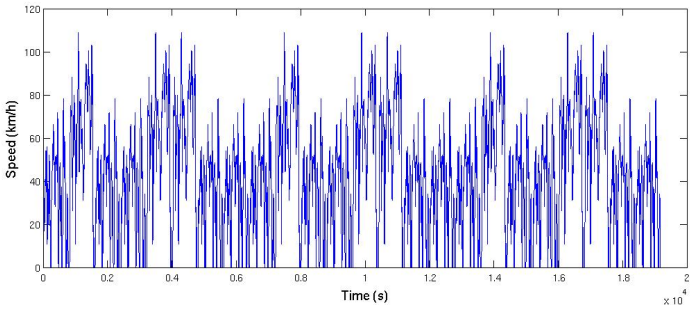


Fig. 1. Drive Cycle used for evaluating the performance of the hybrid energy management. Total cycle length is 230 km. We can see both urban and extra-urban (e.g. highway) parts.

We have therefore decided not to integrate those objectives as fixed constraints nor to perform single-objective optimization using a weighted aggregation of objectives. Rather we include all separate objectives in a many objective optimization.

Since the focus in this article is not a detailed model of the car and the different objectives, we have decided to use simplified methods for computing objective values (also in order to keep computation times low). Specifically we are optimizing for the following objectives:

1. Fuel consumption (**FC**) (as l / 100 km) over the full driving cycle. This measure directly correlates with the CO_2-emissions. This information is taken directly from the simulator using motor efficiency maps. The fuel consumed during the trip is reduced by the amount of fuel saved through the remaining battery energy at the end of the trip (see above).
2. Battery stress (**BS**): The battery as one of the most sensitive and expensive parts of the car has to be kept at a high State of Health (SOH) over the full lifetime of the car. The operation strategy therefore has to consider how the battery is affected by different control actions. Unfortunately, accurate battery aging models are very complex and difficult to calibrate. We therefore used a simplified model, by monitoring only (dis-)charging current to and from the battery and battery temperature. If currents or temperature leave a specified safety zone, battery stress is generated. The target of the optimization is to minimize these stress events.
3. Operation changes (**OPC**): A start and stop of the engine can often be heard and felt by the driver. It is therefore advisable to keep the frequency of on/off switches below a certain level. We therefore compute the number of on/off switches that happen within less than a given interval time (we chose 60 s) after the last on/off switch.
4. Emissions (**Emis**): While CO_2-emissions are directly proportional to the consumed fuel, other emissions like NOx are also linked to the operation

of the combustion engine but only when the catalytic converter has not yet reached its operation temperature. We therefore compute emissions (excluding CO_2) considering the current temperature of the catalytic converter.

5. Urban operation (**U-Noise**): plugin-hybrid vehicles are a potential answer to pollution and noise problems in metropolitan areas, if the operation strategy manages to keep the combustion engine off while driving in the city. For this objective we count the number of simulation time steps when the engine is on while the car is driving slower than 50 km/h (urban speed limit).
6. Noise (**Noise**): A car generates noise from various sources, the two most important ones being tire and wind noise (resulting from car motion) on the one hand and engine noise on the other hand. While the former can't be avoided given a targeted speed of the car, the latter is influenced by the operation strategy. We model wind and tire noise after [11] and engine noise using measured data. The noise objective will only consider instances when the engine noise exceeds tire and wind noise, because otherwise it will not be perceived by the driver. The objective is to minimize those instances.
7. Battery State-of-Charge (**SOC**): Since we translate the remaining state of charge into reduced fuel consumption, the operation strategy might decide to end the trip with a higher final state-of-charge. But there are several reasons to prefer a lower SOC: Firstly, for some battery types, lifetime is higher when the battery is kept at a lower state of charge. Secondly, when the car arrives with full batteries at a charging station, it can't use grid electricity to recharge. We therefore compute the accumulated SOC level (the area under the SOC curve) as our last objective.

We also add a penalty term to all objectives when the simulator reports an infeasible solution or does not finish due to some model inconsistency.

2.2 Rule-based Controller

The core of the controller is a small set of rules, that are configured by a set of parameters (in *italics*):

- **Rule 1:** If speed is below v_{off}, turn off engine.
- **Rule 2:** If SOC is above SOC_{max2}, turn off engine. Note that SOC_{max2} is not the maximum SOC of the battery but the SOC level below which the ICE will begin to recharge the battery.
- **Rule 3:** If SOC falls below SOC_{min} turn on engine (overrides first rule)
- **Rule 4:** If engine is ON and speed is below v_1 use engine operation point 1 (revolutions per minute (rpm) rev_1 and torque $torque_1$)
- **Rule 5:** If engine is ON and speed is between v_1 and v_2 use operation point 2 (rev_2, $torque_2$)
- **Rule 6:** If engine is ON and speed is above v_2 use operation point 3 (rev_3, $torque_3$)

This simple rule base is associated with a set of 11 parameters. Initially the parameters were manually tuned by an engineer (a tedious task considering

the large number of objectives). We will tune those parameters using our MAO method. The rules can be computed very quickly and are easy to understand, but their impact on different objectives is hard to predict.

Minimum and maximum values for parameters are given in Table 1. Parameter ranges are soft constrained by adding a penalty to all objectives proportional to the deviation of the parameters from the permitted range.

Parameter	Range	Parameter	Range
SOC_{max2} (%)	[25,50]	SOC_{min} (%)	[20,30]
v_1 (km/h)	[20,60]	v_2 (km/h)	[50,100]
rev_1 (/m)	[2500,3000]	$torque_1$ (N · m)	[5.23,12.56]
rev_2 (/m)	[3000,4000]	$torque_2$ (N · m)	[5.65,17.59]
rev_3 (/m)	[4000,5000]	$torque_3$ (N · m)	[7.77,23.03]
v_{off} (km/h)	[20,50]		

Table 1. Parameter ranges

2.3 Simulation Environment

The behavior of the car is simulated using a model of a hybrid car implemented in Matlab/Simulink [8]. The runtime for a single simulated driving cycle is a couple of seconds (8 s on a standard desktop computer in Simulink Normal mode). This is sufficiently short for an extensive evolutionary optimization (using 10,000 iterations) that finishes in less than a day.

3 Evolutionary Many Objective Optimization

In the literature the optimization for two or more objectives is often referred to as multi-objective optimization (MOO). For more than one objective there is for most applications not one single optimum solution. Rather, the target is to approximate the Pareto-front of non-dominated solutions. Therefore the target of MOO is to find a diverse set of non-dominated solutions that come as close to the Pareto-front as possible

For example, the *NSGA-II* algorithm proposed by Deb [4] is a standard multi-objective optimization algorithm that uses the distribution of solutions and their ranking regarding the Pareto-front as selection criteria. It has been successfully used in a number of applications [12] for two or three objectives.

In a recent paper [9] we have shown that for an optimization task with 6 or 7 objectives standard methods like NSGA-II [4] fail to improve the quality of the solution set over the course of generations.

Since methods which are commonly referred to as *multi*-objective optimization methods perform weakly for more than 3 objectives, the term *many*-objective optimization (MAO) has been introduced [6] to label a new class of algorithms that show better performance for a larger number of objectives.

3.1 SMS-EMOA

The SMS-EMOA (*S*-*M*atrix *S*election - *E*volutionary *M*ultiobjective *O*ptimization *A*lgorithm) [2] directly uses the hypervolume [14] of a population for the selection operator. The hypervolume is the volume in fitness space covered by a solution relative to a pre-defined reference point (e.g. the worst solution found). A population's hypervolume is the total hypervolume covered by all solutions of the population. It is maximized when the solutions approach the Pareto-front.

The SMS-EMOA is a steady-state optimization algorithm where the current candidate solution's hypervolume contribution is compared to those of solutions already in the population and the individual with the weakest contribution is removed. This method is conceptually very simple and as we have shown in [9] has a very good performance. The algorithm's drawback is the substantial computation effort for the hypervolume. For larger population sizes and number of objectives, hypervolume computation time can easily exceed fitness computation even for demanding applications. We encountered this effect when during the process of optimization we switched from 6 to 7 objectives and total optimization times increased substantially. Based on a comparison study of different methods for hypervolume computation [10] we decided that the method of While et al. [13] provides the best performance of all tested methods and drastically reduced computing times. With the new algorithm up to 10 objectives should be feasible on a typical computing budget. Table 2 shows the settings we used for the SMS-EMOA.

Parameter	Value
Population Size	100
Crossover Probability	1.0
Mutation Probability	0.2
Mutation distribution parameter η_m	20
Crossover distribution parameter η_c	15
Stopping Condition	10000 fitness evaluations

Table 2. Settings used in the SMS-EMOA

4 Optimization results

The result of the many-objective optimization is a population of (100) solutions (configurations) representing the set of trade-offs for all involved objectives. In the following we show a number of performance measures. To better compare with the baseline solution (parameters manually tuned by an engineer) all objective values are normalized by the values from the baseline solution, i.e. the baseline solution has a value of 1.0 for all objectives.

4.1 Summary of optimization results

First we show in Fig. 2 the development of the hypervolume of all solutions relative to the worst solution found (in all previous runs). We see that the hypervolume increases smoothly up to approximately 6000 iterations. The actual shown hypervolume in the plot might decrease due to either a wrongly specified reference point to calculate the hypervolume and the nature of SMS-EMOA [7].

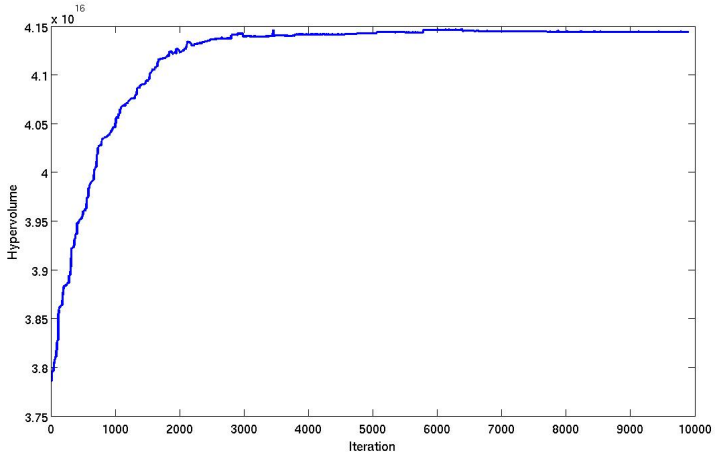


Fig. 2. Development of hypervolume over iterations for the SMS-EMOA optimization algorithm. Values were computed with unnormalized objective values.

We then show a plot of the pairwise correlations of objective values of all final solutions in Fig. 3. Diagonal elements show histograms of objective values, plots above the diagonal give pairwise correlations between two objectives each, while below diagonal elements show the correlation factor between the elements. Positive values are indicated as upward pointing arrows, zero correlation as right pointing and negative correlations as downward pointing arrows. We see that some objectives (like Emission and operation point changes (OPC)) are highly correlated, others, like fuel consumption and battery stress, are anti-correlated. Especially the objective SOC seems to be rather uncorrelated to other objectives. Information like this can help the engineer to understand the basic compromises that need to be made, like between fuel consumption and battery lifetime in our example. It also shows that either OPC or emissions might be removed as an objective with limited impact due to their strong correlation.

Fig. 4 shows a parallel coordinate plot in objective space of the final 100 solutions from the SMS-EMOA. This plot again visualizes the relation between different objectives, the clustering of solutions in terms of objective values, and the resultant minimum and maximum values. Please note especially the very low (good) objective values for many solutions. The majority of objectives is below

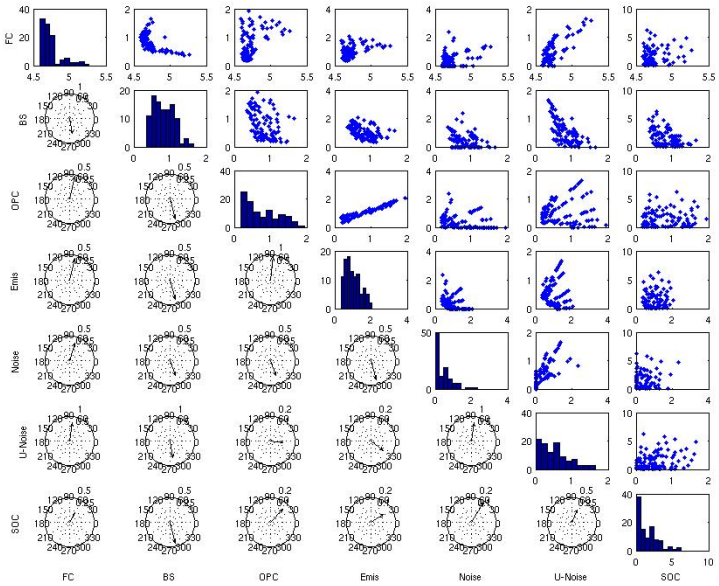


Fig. 3. Correlation plot of SMS-EMOA final solutions after optimization

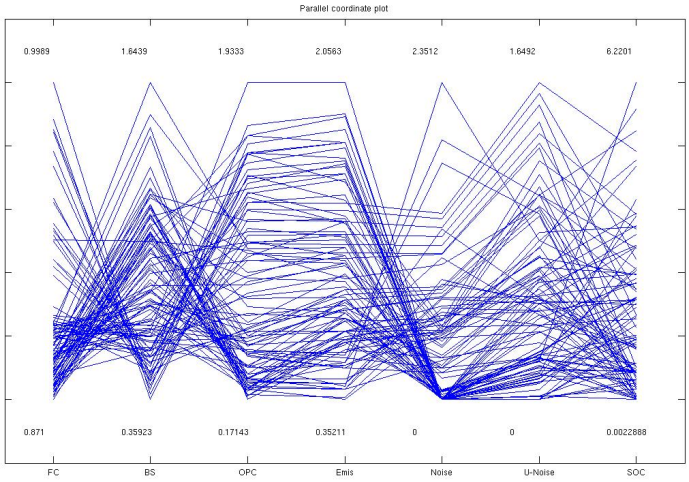


Fig. 4. Parallel coordinate plots in objective space of all 100 solutions in the final population of SMS-EMOA.

the baseline value (1.0) for a large share of the solutions. While fuel consumption was reduced only little, the other objectives can be improved by a large margin. The average reduction over all objectives was up to almost 50% for the best solution. Among the solution set, there are also configurations that improve on all objectives (i.e. no deterioration compared to the baseline).

5 Discussion on post-processing

Finding an optimal setting for a hybrid car controller is a challenging optimization task. However, due to recent advances in many-objective optimization algorithms and faster simulators, it is now possible to derive a larger set of solutions that provide a variety of parameter sets each being optimal in its own regard. We even found solutions that are better than a manually tuned one for all objectives. From a purely optimization point of view this is a very satisfying result, however, the engineer is stilled faced with the issue of deciding which strategy (solution) to implement. By design, many or even all solutions are non-dominated, that means that the majority of solutions is not clearly worse than any other solution. Some of the solutions might be sorted out by considering constraints that were not included in the optimization (e.g. because they are difficult to formalize). Also a clustering might reduce the number of candidates, if many solutions are rather similar. Another option might be to search for knee points [3] in objective space, these are solutions that strongly stand out (in terms of their objective values) relative to similar solutions. If the number of candidates has been reduced sufficiently, the remaining solutions could be implemented in a real car and validated directly (probably reducing the number of candidates further). Alternatively, using the insights gained during the optimization (distribution and correlation of objective values), the engineer might define weights for the different objectives and determine the best overall solution through weighted fitness aggregation.

Finally, the engineer might decide to implement several solutions (parameter sets) and shift the decision to a later stage, in an extreme case leaving the customer with the decision which solution to run currently in the car.

6 Summary and Outlook

In this work we have shown a many-objective optimization for the rule base of a hybrid car controller. Using 11 design parameters and 7 objectives, the well-known SMS-EMOA method finds solutions that provide a substantial improvement over the baseline solution, which was manually tuned by an engineer. While the main objective (fuel consumption) could be reduced only by a few percent, other objectives, which are often ignored, could be strongly improved. Total optimization time is less than a day without any technical speed-up, but can be brought down to far less than an hour using Simulink code export and parallelization. We also demonstrated how a visualization of optimization results can help the engineer to better understand the underlying optimization problem

in terms of parameter and objective correlations and trade-offs. A challenge that remains is how to decide for one out of the set of final solutions from the optimizer. This decision will have to be made by the responsible engineer but could probably be supported by a clever visualization, analysis, and post processing of optimization results.

An important next step is to compare our optimization results to other methods like Dynamic Programming which promise a solution close the global optimum but only consider a single objective. It could be interesting to compare our solutions in terms of fuel consumption, but also the other objectives.

Another research direction we investigate is preference-based optimization. If the preference of a decision maker is available in optimization, the preference information can be used to focus the search on the preferred regions in the many-objective space by evolutionary algorithms.

References

1. Bacher, C., Krenek, T., Raidl, G.: Reducing the number of simulations in operation strategy optimization for hybrid electric vehicles. In: Proceedings of EvoStar Conference (EvoApplications). pp. 553–564. LNCS, Springer (2014)
2. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: Multiobjective selection based on dominated hypervolume. European Journal of Operational Research 181(3), 1653–1669 (2007)
3. Branke, J., Deb, K.: Integrating user preferences into evolutionary multi-objective optimization. In: Jin, Y. (ed.) Knowledge Incorporation in Evolutionary Computation, Studies in Fuzziness and Soft Computing, vol. 167, pp. 461–478. Springer (2004)
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182 – 197 (2002)
5. Desai, C., Williamson, S.S.: Optimal design of a parallel hybrid electric vehicle using multi-objective genetic algorithms. In: Vehicle Power and Propulsion Conference, 2009. VPPC'09. IEEE. pp. 871–876. IEEE (2009)
6. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: A short review. In: Proc. of 2008 IEEE Congress on Evolutionary Computation (World Congress on Computational Intelligence). pp. 2419–2426. IEEE (2008)
7. Judt, L., Mersmann, O., Naujoks, B.: Do hypervolume regressions hinder EMOA performance? Surprise and relief. In: Purshouse, R., Fleming, P., Fonseca, C., Greco, S., Shaw, J. (eds.) Evolutionary Multi-Criterion Optimization: EMO 2013. Lecture Notes in Computer Science, vol. 7811, pp. 96–110. Springer (2013)
8. MATLAB: version 8.2 (R2013b). The MathWorks Inc., Natick, Massachusetts (2013)
9. Narukawa, K., Rodemann, T.: Examining the performance of evolutionary many-objective optimization algorithms on a real-world application. In: Proc. of 2012 IEEE International Conference on Genetic and Evolutionary Computing. pp. 316–319. IEEE (2012)
10. Priester, C., Narukawa, K., Rodemann, T.: A comparison of different algorithms for the calculation of dominated hypervolumes. In: Proc. of 2013 Genetic and Evolutionary Computation Conference. pp. 655–662. ACM (2013)

11. Sandberg, U.: Tyre/road noise: myths and realities. Tech. rep., Swedish National Road and Transport Research Institute (2001)
12. Wang, L., Ng, A.H.C., Deb, K. (eds.): Multi-objective Evolutionary Optimisation for Product Design and Manufacturing. Springer (2011)
13. While, L., Bradstreet, L., Barone, L.: A fast way of calculating exact hypervolumes. IEEE Transactions on Evolutionary Computation 16, 86 (2012)
14. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review. Evolutionary Computation, IEEE Transactions on 7(2), 117–132 (2003)