


# Cost Optimization of Product Families Using Solution Spaces: Application to Early-Stage Electric Vehicle Design

S. Rötzer<sup>1</sup>, V. Berger<sup>2</sup> and M. Zimmermann<sup>1</sup>

<sup>1</sup>Technical University of Munich, Germany, <sup>2</sup>DeepDrive GmbH, Germany

 sebastian.roetzer@tum.de

## Abstract

Companies offer products in different variants to reach more customers. This increases internal variety and cost. However, reducing those cost is difficult due to complexity. Complexity arises from: combinatorics; many design variables interacting with each other; coupling of technical and economical perspectives. This paper presents an approach based on (1) building a complex system model of modular models; (2) identifying the potential for standardization from a technical perspective; (3) cost-optimizing the degree of standardization. A product family of electric vehicles was optimized.

*Keywords: product families, optimisation, complexity, solution space engineering, data-driven design*

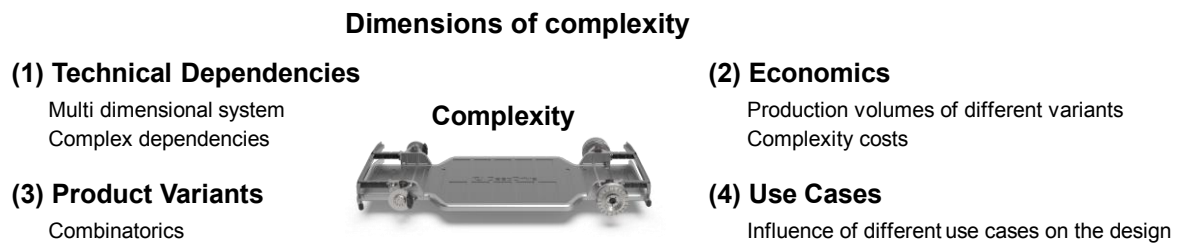
## 1. Introduction

An automotive start-up provides rolling chassis with in-wheel motors for different customer segments. The rolling chassis consist of steered front wheels, in-wheel driven rear wheels and a frame with integrated battery (Rosen et al., 2021). To serve different customer segments the rolling chassis should be available in several product variants. These product variants constitute a product family, which needs to be designed. Designing such a product family comes with complexity in several dimensions. (Figure 1) depicts the four dimensions of complexity related to the example: (1) complexity regarding the technical dependencies; (2) complexity regarding economics; (3) complexity regarding combinatorics; and (4) complexity regarding different use cases of different product variants.

The technical complexity (1) arises from the high number and nature of dependencies in the technical system. The technical system, covered in this paper, consists of a model with 19 dimensions. Support is needed, to visualize the technical dependencies during design. An unstructured interview with experts at the start-up reveals, that one cannot image more than three dimensions in mind.

The economical complexity (2) regards the standardisation of components. Whenever a component variant is shared in different products of a product family, the production volumes of the component variants are accumulated, which leads to economies of scale. But sharing components by standardization also leads to over dimensioning, which increases production costs. Both effects influence how the components are shared and designed within a product family.

Designing a product family with several products (3) comes with a high number of possible standardization opportunities. This example includes four products and three considered components. Component variants are different instantiations of a component which can be shared among a product family. The commonality pattern describes how the component variants are shared (Fujita and Yoshida, 2004). Here generalized commonality applies, i.e., component variants can be shared within



**Figure 1. Dimensions of complexity with respect to the application example (rolling chassis)**

the product family without restrictions: a component variant can be shared in arbitrary subsets of the product family. Bell's number (Baylis et al., 2018) can be used to calculate all possible combinations for generalized commonality. Here, there are  $(B_4)^3 = 3,375$  possibilities to distribute component variants in the product family. This high number requires an algorithm-based approach.

The different product variants are used in different use cases (4): While a small car is rather used in urban use cases, bigger cars are also used for longer motorway journeys. Thus, the design must consider different driving scenarios. An approach needs to consider all those dimensions of complexity.

In early-stage product development, uncertainty is a central topic. In this example, uncertainty occurs in financial planning and the availability of project and product information. First, calculating precise complexity costs is a difficult task and different for each company. Second, a start-up has no information available about its predecessors.

## 2. Related Research

We consider qualitative approaches, such as Johannesson et al. (2017) or Long et al. (2009), as unsuitable for this mathematical problem statement. Their strengths come into play in a fuzzier development environment. The approach should minimize the overall cost of the product family considering sales volumes and effects of economies of scale. Therefore, approaches with different objectives are out of scope, such as maximized commonality (Eichstetter et al., 2015; Fellini et al., 2006). Some approaches restrict commonality: either all product variants share the component or none (Chen and Wang, 2008). But we also want to allow a partial standardization of components to a subset of product variants (generalized commonality). We do not want to prescribe a certain commonality pattern (Wei et al., 2019), but receive an optimized commonality pattern as an output. With the rim allowing only integer values, the problem statement includes discrete design variable values. Therefore, the approach needs to solve a problem with both discrete and continuous values. Approaches with gradient-based algorithms, such as Fujita and Yoshida (2004), can easily get stuck in local minima of discrete problem statements. The uncertainty the start-up is facing requires an optimization of different cost scenarios and, thus, a fast-calculating algorithm. One level approaches quickly become intractable due to the dimensionality of the problem (number of design variables, constraints, combinatorics) (Simpson et al., 2001). But also two-level approaches with classical monolithic optimization algorithms, such as Rötzer et al. (2021), require long calculation times when it comes to an increasing number of product variants. The methodology presented in chapter 3 is capable of overcoming these barriers.

## 3. Methodology

The methodology in this paper combines two existing approaches: (1) sequencing of information in modular model-based systems design (Rötzer et al., 2020a) and (2) cost-optimized product family design using solution spaces (Rötzer et al., 2020b). (1) decomposes the complex product family into manageable modular sub-systems. (2) optimizes the product family with respect to its overall cost.

### 3.1. Cost-optimization of Product Families Using Solution Spaces

Rötzer et al. (2020b) present a two level approach to optimize product families. On the first level the approach analysis each product variant independently. It searches for areas in the design space which fulfil all requirements of the product variant. Zimmermann et al. (2017) call this area solution space. Solution spaces are visualized on the level of the design variables. When solution spaces of different

product variants overlap, it means that those design variable values fulfil requirements of more than one product variant. The corresponding components (described by the design variables) can be shared among those product variants. Thus, overlapping solution spaces indicate potential for standardization from a technical perspective. So far, the approach does not consider any costs. Rectangular solution spaces decouple the design variables from each other by defining permissible intervals for each design variable independently. Those intervals are the input for the cost optimization on the second level. All technical possibilities for standardization from the first level are then evaluated in the second level according to a cost function to find the optimized level of standardization (economical perspective). The result is a cost-optimized commonality pattern and cost-optimized design variable values for the components of the product family.

### 3.2. Prerequisites

To apply this methodology, the problem at hand needs to fulfil the following prerequisites. Solution spaces require both quantitative requirements and quantitative system models (Zimmermann *et al.*, 2017). A benchmark analysis provides the required quantitative requirements. It reveals four variant drivers: acceleration, top speed, range, hill-climbing capacity; and provides quantitative requirements for each product variant of the product family (see Figure 2). Section 3.3 gives an overview over the quantitative models used. Furthermore, the problem at hand needs to be a scale-based product family. Scale-based product families derive product variants by scaling their components. Scaling variables size the components of the product family to instantiate individual performances of different product variants (Simpson *et al.*, 2001). Here the following components are scaled: motor, inverter, and battery.

		Range	Acceleration 0-100 km/h	Ascent	Max velocity	Distance motor / rim
		[km]	[s]	[%]	[km/h]	[m]
Products	S	≥500	≤7	≥40	≥150	≥0.03
	M	≥600	≤7	≥40	≥150	≥0.03
	L	≥600	≤12	≥30	≥150	≥0.03
	XL	≥600	≤12	≥25	≥150	≥0.03

Figure 2. Overview of product variants and their requirements

### 3.3. Model of the Technical System

(Figure 3) shows the representation of the technical system. The nodes indicate the attributes of the technical system. The attributes on the left are design variables. A designer can directly influence design variables, e.g., the rim diameter. The attributes on the right represent the quantities of interest (including the four variant drivers: range, acceleration, max. velocity and hill-climbing capacity). Requirements can be applied upon the quantities of interest, e.g., the range. The system has 19 design variables and seven quantities of interest. The design variable values determine the quantities of interest. Edges indicate the dependency between the attributes. Models quantify those dependencies. 22 MatLab™ sub models quantify the dependencies of the electric vehicle product family. The algorithm described in Rötzer *et al.* (2020a) combines them to the overall system model. The sub models are aggregated into four groups: power train; battery; performance functions; range and cycles. The colour code highlights their position in the system model. The models reach from formulae, such as equations for top speed; over data points for different driving cycles; to neural networks to describe the characteristics of the electric motor. They can be provided from different sources, e.g., the engine designer.

### 3.4. Cost Model

The cost model serves as an objective function for the optimization. Equation 1 formulates the optimization problem.

$$\min_{\xi_{k,j}, x_{j,i}} C_{PF} = C_x + C_{n_{CV}} \quad \text{Subject to } \xi_{k,j} \in \{1, \dots, n_{cv,j}\}; x_{j,i} \in \mathbb{R}^n \quad (1)$$

$j = 1, \dots, n_c$ : component (type) index  
 $i = 1, \dots, n_{cv,j}$ : component variant index  
 $k = 1, \dots, n_p$ : product variant index  
 $\xi_{k,j}$ : commonality pattern that assigns each product variant  $k$  and component type  $j$  one component variant  $j$

$n_c$ : number of component types  
 $n_{cv,j}$ : number of component variants of component type  $j$   
 $n_p$ : number of product variants  
 $x_{j,i}(\xi_{k,j})$ : design variable values for component type  $j$ , component variant  $i$  which contain only good designs (shared solution space according to  $\xi_{k,j}$ )

$C_{PF}$  is the cost function of the product family, which is to be minimized, consisting of costs depend on the design  $C_x$  and costs dependent on the number of component variants  $C_{n_{CV}}$  (see equation (2))

$$C_x = \sum_j^{n_c} \sum_i^{n_{cv}} N_{i,j} c_j d_j \quad C_{n_{CV}} = \sum_j^{n_c} n_{cv,j} c_{c,j} \quad (2)$$

$N_{i,j}$  production volume of component type  $j$ , component variant  $i$ ;  
 $c_j(x_{j,i})$  design-dependent cost function of component type  $j$ , e.g. manufacturing or procurement;  
 $d_j(N_{i,j})$  function for effects of economies of scale for component type  $j$ ;  
 $c_{c,j}$  complexity costs per component variant of component  $j$ .

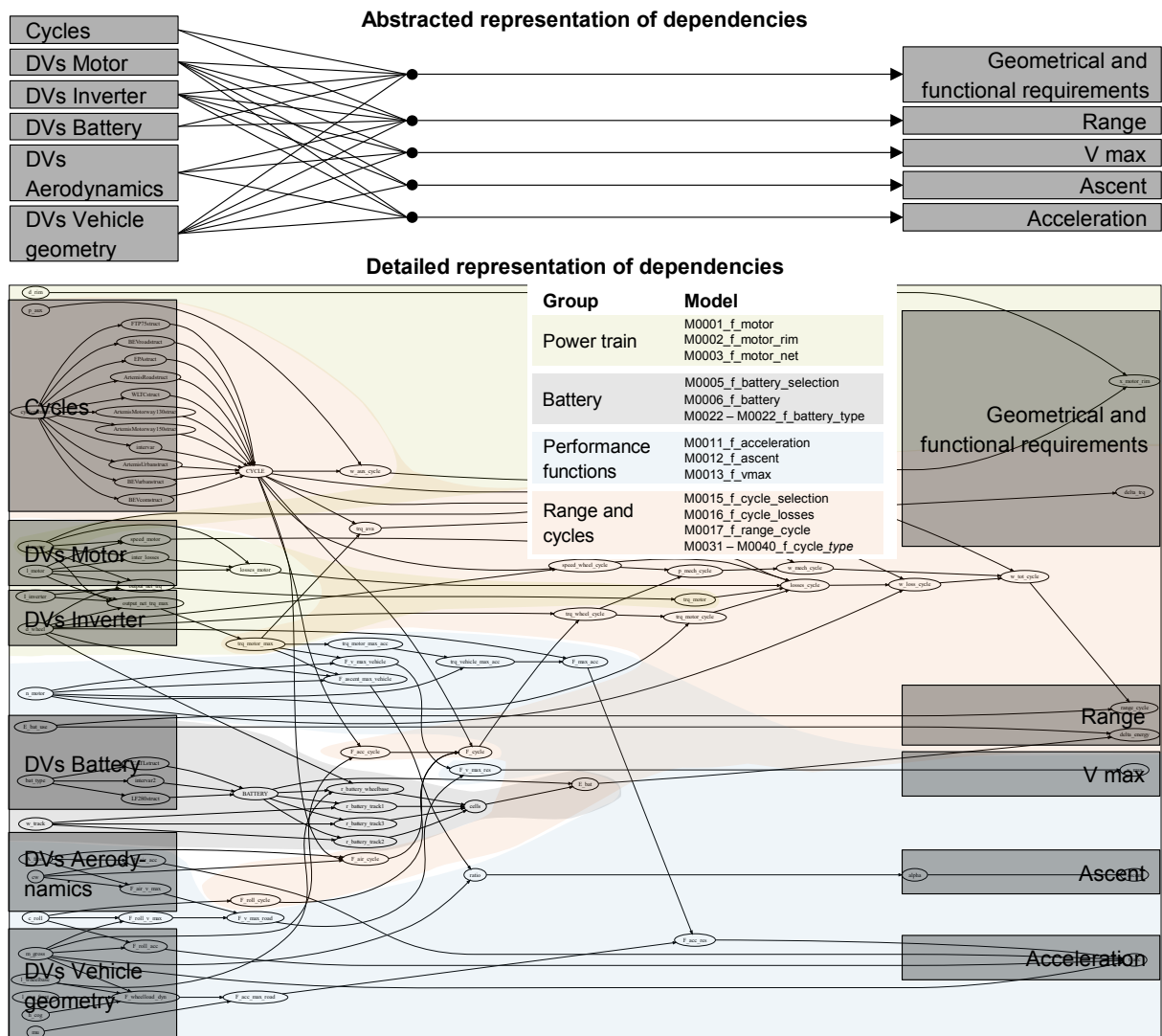


Figure 3. Abstracted (top) and detailed (bottom) representation of dependencies of the technical model; dependencies between design variables (left) and quantities of interest (right)

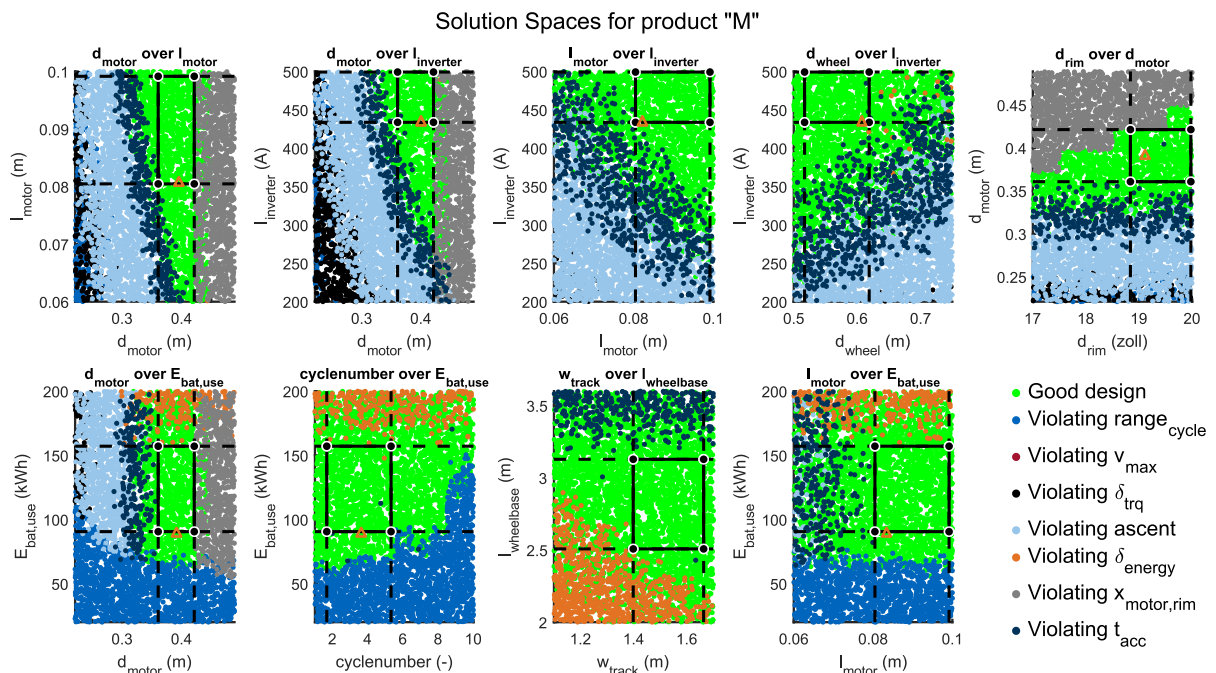
## 4. Results

### 4.1. Cost-optimized Commonality Pattern

The first step encompasses the identification of good designs and the definition of the rectangular solution spaces. (Figure 4) illustrates the solution space for product M. These diagrams are available for all four product variants. The 19-dimensional technical system is projected in 2D diagrams.

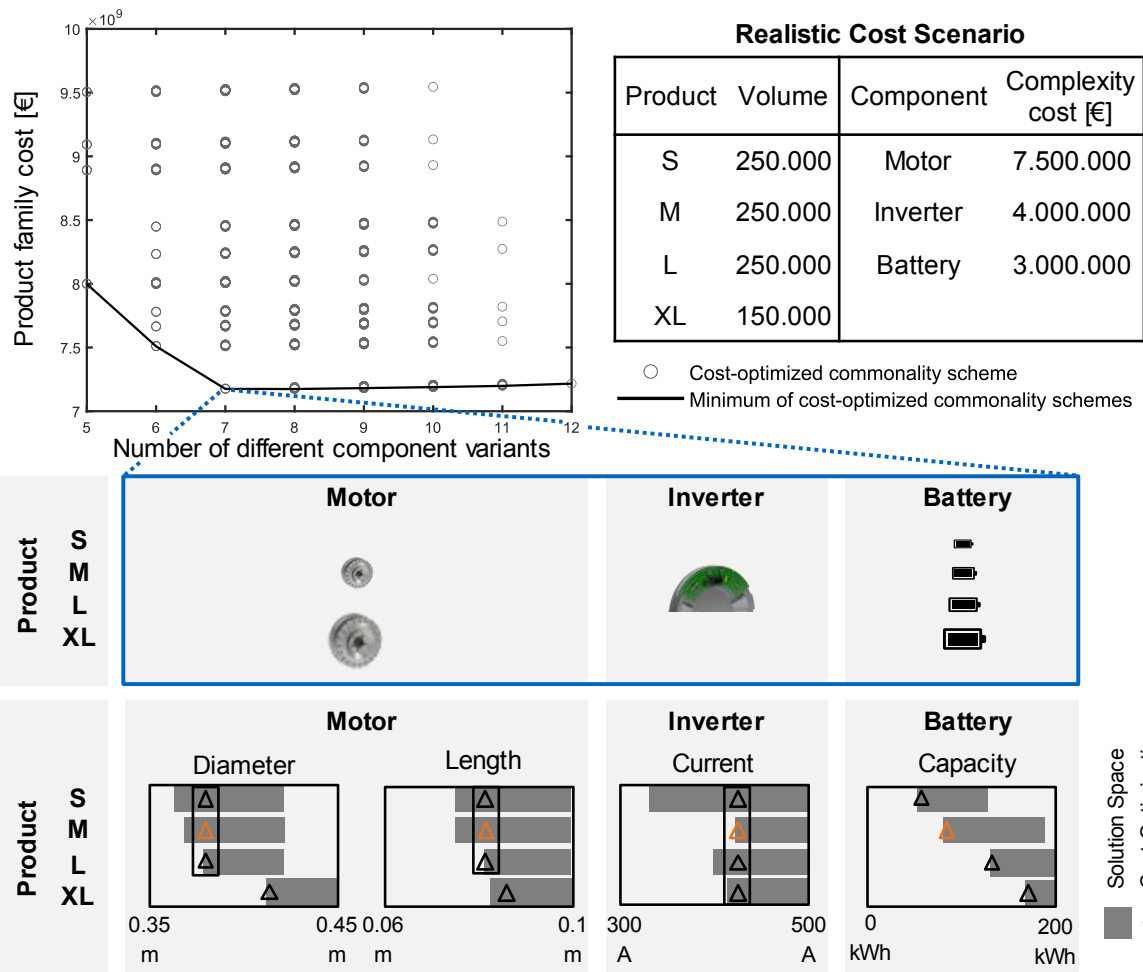
The designer can decide to manually optimize the solution spaces or choose an algorithm. This result is generated by using an algorithm maximizing the solution spaces. Nevertheless, the designer must check the plausibility of the optimization results. Furthermore, the designer may manually adjust the solution spaces with respect to his/her experience. Analysing the solution spaces helps to increase the understanding of the system. This is explained by analysing the motor: The top left diagram of (Figure 4) shows the length and diameter of the motor. In areas, where the motor is small, the black requirement is not fulfilled. That is, that the design does not provide enough torque to serve every operating point of the driving cycle. In areas of a larger motor diameter and length, the vehicle design is not able to climb the required ascent (light blue). The requirement for the minimum acceleration time from 0 to 100 km/h (dark blue) is most critical for product M. In the area, where the motor diameter is  $\geq 0.4$  m, the motor does not fit into the rim (grey). Besides designing components, the designer may also analyse the vehicle design with respect to several driving situations. They are represented by ten driving cycles. The diagram cycle number over  $E_{bat,use}$  shows the energy consumption for all considered cycles. This depiction allows a design of the battery size according to the ten driving cycles considering the overall design simultaneously. As shown in these two examples, every other design variable may be projected and analysed in this manner.

The design variables, which are not considered in the cost-optimization, are constrained by intervals. Those intervals can be directly derived from the solution spaces. For instance, the wheelbase  $l_{wheelbase}$  and the track width  $w_{track}$  are constrained to  $[2.5; 3.1]m$  and  $[1.4; 1.7]m$  respectively (see (Figure 4): column three, row two). This means that we give requirements on the design of subsystems or components of the rolling chassis. If those requirements are fulfilled, the cost-optimized standardization pattern can be realized. The next step is to find the most cost-efficient commonality pattern and design variable values of the components within the product family. We do that by minimizing the overall cost of the product family, consisting of three components: motor, inverter, and battery.



**Figure 4.** 2-dimensional design space projections containing a box-shaped solution space (represented by black rectangle) for product M and cost-optimized design (orange triangle)





**Figure 5. Top: result of the cost optimization for the realistic scenario; middle: standardized component variants of the product family; bottom: cost optimized design variable values**

The components are scaled along the four design variables: diameter and length of the motor, the inverter current and the battery size. The cost optimization considers all good designs in the rectangular solution spaces. (Figure 5) (top) illustrates the cost optimization of the product family for a realistic cost scenario. The y-axis indicates the overall cost of the product family. The x-axis shows the number of component variants used to build the product family. Five component variants represent the highest commonality possible from a technical perspective. 12 is the maximum number, if each of the four products has its own component variants (three components):  $4 \cdot 3 = 12$ . The circles represent the cost-optimized design of a specific commonality pattern. The line connects all minimum costs of different commonality patterns. The product family cost decreases between five and seven component variants. Here, the battery size is adapted to the dedicated use case and range requirements. The manufacturing cost of the battery is the highest for all considered components. Therefore, standardization by over dimensioning is most expensive. Thus, using individual battery sizes for each product variant lead to a decreased product family cost. The most cost-efficient design consists of seven component variants. Choosing a product family with more than seven component variants again increases the product family costs. Here, the complexity costs, which occur when introducing a new component variant, exceed the savings from standardization including cost degression.

(Figure 5) (middle) illustrates the seven cost-optimized component variants. With these component variants each of the four product variants can be created. The commonality pattern consists of two motor variants, one inverter variant and four battery variants. One motor variant is shared in the classes S, M, and L. Another motor variant is used for the product XL. The inverter is shared for all products. The battery is always optimized for the vehicle size and range requirement.

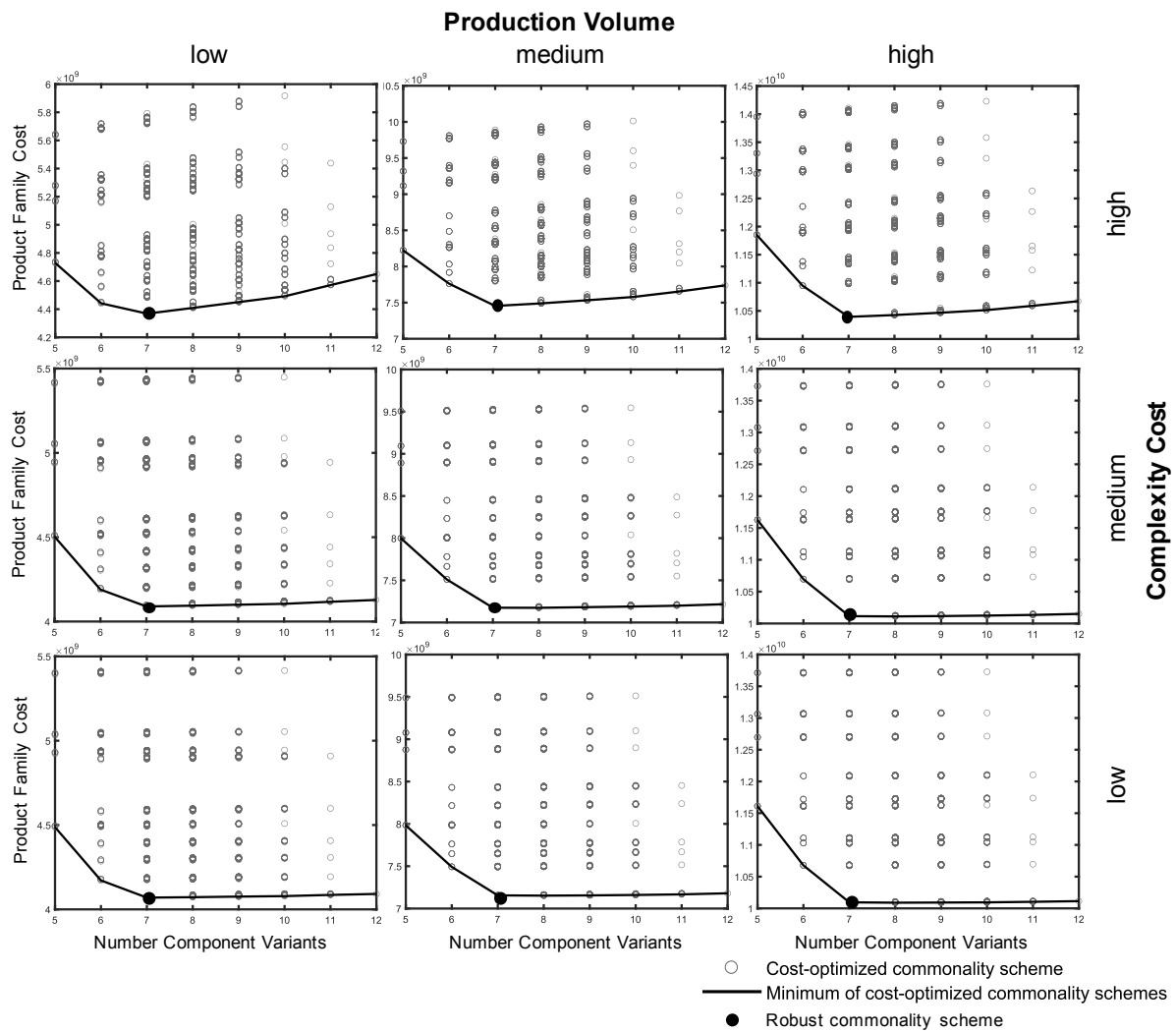
In (Figure 5) (bottom) the triangles show the concrete design variable values of the optimized components.

All shared component variants are marked with a black rectangle. The motor consists of two design variables. Both need to incorporate the commonality pattern, so that the component can be shared. The orange triangles indicate the optimized design for product "M", as can be also seen in (Figure 4).

## 4.2. Analysis of Different Cost Scenarios

(Figure 6) presents nine cost optimization results. The results are varied along production volume and complexity cost. Thus, we consider fluctuations in production volumes as well as higher cost with respect to developing variants and maintaining them over the lifecycle. For all considered scenarios the cost-optimized result is seven component variants, shown by the commonality pattern in (Figure 5). There are two sections of the minimum cost curve: (1) from five to seven component variants, (2) from seven to twelve component variants. Considering five to seven component variants (1), the product family cost is decreasing. When choosing seven component variants, a battery for every product is introduced.

This leads to the usage of smaller battery sizes if it is technically feasible. Due to the high production cost compared to the complexity cost of a battery, the introduction of a smaller battery variant leads to a decreased product family cost even if one considers low production volumes. Considering seven to twelve component variants (2), the product family cost is increasing for all calculated scenarios: each additional component variant adds complexity costs to the product family, which are higher than savings due to individualization of component variants. The higher the complexity costs and the lower the



**Figure 6. Analysis of the influence of changes in production volume and complexity costs on the cost-optimized commonality pattern; robust commonality pattern**

production volumes, the steeper is the increase of product family cost with increasing number of component variants. This means there are fewer produced units that can share the complexity costs. The computation time for one cost scenario takes about ten minutes on a four core, 1.80 GHz Intel Core i7-8550U CPU.

### 4.3. Use Case Related Technical Analysis

The technical dependencies determine the system behaviour. But also the way a product is used needs to be considered. In terms of a vehicle, this can be considered by driving cycles. By projecting a 19D design space to a 2D projection, we can visualize the influence of the diameter of the motor and the capacity of the battery for different cycles without neglecting the influence of the other 19 design variables. These 2D diagrams simplify the handling of complexity without simplifying the system itself. (Figure 7) shows these projections for different driving cycles. The Artemis Urban cycle represents the urban use case. The consumed energy decreases with an increasing motor diameter. This correlation can be explained with the efficiency characteristics of the drive train. The motor is most efficient in areas of medium motor speed and low torque. If we take a bigger motor for the same load, it is more efficient, as the bigger motor can run this load point in a more efficient operating point. Furthermore, in the Urban cycle the efficiency of the drivetrain is a special interest due to the high number of acceleration and deceleration processes. Considering road use cases, the energy consumption for different motor diameters depends on the selected use case. Left road cycle is BEV Road, right is EPA cycle. The cycles differ in their velocity profile. The BEV Road cycle contains many accelerates and deceleration processes. There is hardly any constant driving. EPA contains mostly constant driving speeds around 90 km/h. The more acceleration and deceleration processes, the more important is the motor diameter for the design of an overall energy efficient car. The last use case is motorway. The qualitative curve of the used energy is similar to the EPA cycle. But the overall used energy is higher. This is to be explained by the higher air resistance at these speeds. The considered use cases have a major impact on the motor design.

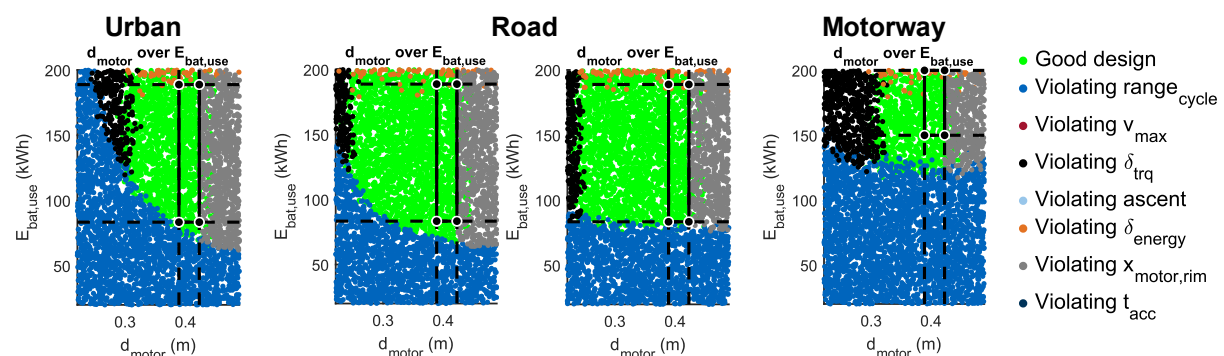


Figure 7. Analysis of consumed energy and motor diameter for four driving cycles

## 5. Summary and Discussion

This paper presents a cost-optimized product family of rolling chassis. The product family consists of three components: motor, inverter, and battery. 19 design variables constitute the system behaviour of the rolling chassis, described by seven quantities of interest. Requirements from a benchmark analysis define four different product variants by constraining the quantities of interest. An algorithm combines 22 modular models to describe quantitatively how the system behaviour emerges from the design variable values. Solution space engineering uses those models to compute a set of good designs (solution space) for each product variant. Good designs are set of design variable values, which fulfil all requirements of a product variant. Rectangular solution spaces decouple design variables from each other, as long as each design variable stays within its intervals. Overlapping design variable values fulfil requirements of two or more product variants. Thus, they can be used to standardize components in a product family. An optimization algorithm then identifies the cost-optimized degree of standardization among all technically possible. A cost function, which includes all decision relevant



parameters, serves as an objective function. Due to fast-calculating (ten minutes) optimization algorithm, the approach can simulate different cost scenarios. We identify a robust, cost-optimized standardization scenario with seven component variants: two motor variants; one inverter variant; and four battery variants. Intervals from the solution spaces constrain the other design variables. They fulfil all requirements of the product family. The model includes complex technical effects, such as the influence of the motor design on the efficiency; the influence of motor efficiency and different use cases (cycles) on the energy consumption.

Especially for a start-up, many calculations are subject to high uncertainty. Even for experts, the estimation of production volumes and complexity costs is a challenging task. Complexity costs differ from one company to another. Complexity costs, which arise along the life cycle of a component, such as warranty or homologation, are hardly measurable. Thus, the variation of complexity cost and production volume, illustrated in (Figure 6), provides useful information in a real-world application. The complexity costs differ by the factor of 30 from the lowest to the highest scenario. Nevertheless, we can reveal a robust standardization pattern: it consists of seven component variants as shown in (Figure 5). This is one big advantage of the approach. We do not need an exact prediction of the complexity costs. We are rather interested in the effect of the costs on the standardization than the costs itself. This helps to identify commonality patterns, which are robust to the quality of predictions. This information helps to define component variants, especially when it comes to deciding, which component variants to develop first. In this example, one motor variant can cover the products S, M, and L. Therefore, we suggest developing this motor variant first.

By constraining design variables, which are not relevant for the optimization of the product family, we can ensure that the overall system meets all requirements, even if we do not at this moment know how the components will look like. Those constrained intervals can guide future design processes, such as frame design, by requirements on those components. This enables a design for commonality at a very early stage in the development process. At the same time the intervals leave freedom for the designers and decouple the design of the components.

By decomposing a complex system in several models, the designer can master complex technical dependencies. The modular model approach also supports the modelling processes involving the whole team. Different experts can integrate their specific models to create a holistic system model, e.g., the electric motor characteristics. Due to the modularity of the models the interfaces are well defined. Thus, the designers can implement different models at the same time. The modular model approach is also useful when we want to analyse the influence of a change to the overall system. For example, the approach can also easily change the battery model if new technologies arise. Nevertheless, setting up the modular models is a time-consuming task especially when every model must be set up from scratch. The effort rises with the number and complexity of the models. Another limitation, as with every other optimization algorithm, is the need for a fast-calculating technical model due to the high number of system evaluations. The motor model used in this paper is based on a very detailed motor model with high calculation time. A neural network is used as surrogate model, which decreases calculation time.

## 6. Conclusion and Outlook

Companies must deal with uncertainty and complexity along different dimensions. This paper presents a methodology consisting of two existing approaches: automated model generation using modular models (Rötzer et al., 2020a); and cost-optimized product family design using solution spaces (Rötzer et al., 2020b). The paper shows that the modular models can increase applicability in modelling complex systems. Modular models divide the overall problem into manageable sub problems which can be solved more easily and are then combined to the overall system model automatically. The paper also shows that product family design using solution spaces can generate robust optimized solutions in a short computational time. Thus, this paper combines two existing approaches and applies this methodology to a novel real-world application example.

Furthermore, the approach can add value along the product development process and life cycle management at three points: (1) early-stage concept development and concept decisions, (2) deriving the cost-optimized commonality pattern, and (3) change management during lifecycle management.

In early-stage concept development, the method is capable of supporting design decisions with solution spaces. Any design variable may be analysed as shown here for energy consumption and the motor diameter. The main purpose of this method is to derive a cost optimized product family. Thereby, the method supports the development of a product portfolio. Another application is in change management.

When new customers order a certain product, the tool can visualize, whether the product can be realized with existing component variants.

The approach only considers good designs, which lie in the rectangular solution spaces. Good designs outside the rectangular solution spaces, are not considered. This results in a loss of good design. Cost optimization using the complete design space can bypass this loss of good designs. In this example, this approach would lead to the high calculation times and a lack of robustness due to point-based optimization. Nevertheless, a comparison of the results considering complete design spaces and rectangular solution spaces can be interesting. Furthermore, the choice of solution spaces influences the optimization result. We have chosen the maximum solution space for each product variant. Other strategies can lead to different optimization results. As an example, minimizing the design variable values may be a touchpoint for further research.

## References

- Baylis, K., Zhang, G. and McAdams, D.A. (2018), “Product family platform selection using a Pareto front of maximum commonality and strategic modularity”, *Research in Engineering Design*, Vol. 29 No. 4, pp. 547–563. DOI: 10.1007/s00163-018-0288-5.
- Chen, C. and Wang, L. (2008), “A modified genetic algorithm for product family optimization with platform specified by information theoretical approach”, *Journal of Shanghai Jiaotong University (Science)*, Vol. 13 No. 3, pp. 304–311. DOI: 10.1007/s12204-008-0304-4.
- Eichstetter, M., Müller, S. and Zimmermann, M. (2015), “Product family design with solution spaces”, *Journal of Mechanical Design*, Vol. 137 No. 12, p. 121401. DOI: 10.1115/1.4031637
- Fellini, R., Kokkolaras, M. and Papalambros, P.Y. (2006), “Quantitative platform selection in optimal design of product families, with application to automotive engine design”, *Journal of Engineering Design*, Vol. 17 No. 5, pp. 429–446. DOI: 10.1080/09544820500287797.
- Fujita, K. and Yoshida, H. (2004), “Product Variety Optimization Simultaneously Designing Module Combination and Module Attributes”, *Concurrent Engineering*, Vol. 12 No. 2, pp. 105–118. DOI: 10.1177/1063293X04044758.
- Johannesson, H., Landahl, J., Levandowski, C. and Raudberget, D. (2017), “Development of product platforms: Theory and methodology”, *Concurrent Engineering*, Vol. 25 No. 3, pp. 195–211. DOI: 10.1177/1063293X17709866.
- Long, L., David, C., Seering, W. and Rebentisch, E. (2009), *Finding Opportunities for Commonality in Complex Systems*.
- Rosen, A., Habersbrunner, M. and Römmelmayer, c. (2021), *Why efficiency matters*, München.
- Rötzer, S., Le Bourgeois, M., Thoma, D. and Zimmermann, M. (2021), “TWO-LEVEL OPTIMIZATION OF PRODUCT FAMILIES: APPLICATION TO A PRODUCT FAMILY OF WATER HOSE BOXES”, *Proceedings of the Design Society*, Vol. 1, pp. 3259–3268. DOI: 10.1017/dsd.2020.178.
- Rötzer, S., Rostan, N., Steger, H.C., Vogel-Heuser, B. and Zimmermann, M. (2020a), “Sequencing of Information in Modular Model-based Systems Design”, *In DS 103: Proceedings of the 22nd International DSM Conference (DSM 2020), MIT, Cambridge, Massachusetts, October 13th-15th 2020*, pp. 1–10. DOI: 10.1017/pds.2021.587
- Rötzer, S., Thoma, D. and Zimmermann, M. (2020b), “COST OPTIMIZATION OF PRODUCT FAMILIES USING SOLUTION SPACES”, *Proceedings of the Design Society: DESIGN Conference*, Vol. 1, pp. 1087–1094. DOI: 10.35199/dsm2020.7.
- Simpson, T.W., Maier, J.R. and Mistree, F. (2001), “Product platform design: method and application”, *Research in Engineering Design*, Vol. 13 No. 1, pp. 2–22. DOI: 10.1007/s001630100002.
- Wei, W., Tian, Z., Peng, C., Liu, A. and Zhang, Z. (2019), “Product family flexibility design method based on hybrid adaptive ant colony algorithm”, *Soft Computing*, Vol. 23 No. 20, pp. 10509–10520. DOI: 10.1007/s00500-018-3622-y.
- Zimmermann, M., Königs, S., Niemeyer, C., Fender, J., Zeherbauer, C., Vitale, R. and Wahle, M. (2017), “On the design of large systems subject to uncertainty”, *Journal of Engineering Design*, Vol. 28 No. 4, pp. 233–254. DOI: 10.1080/09544828.2017.1303664.