Recent research results in

Optimal Design of Product Families

at the

Optimal Design Laboratory
University of Michigan
Ann Arbor
Michigan, USA

A collection of selected publications

May 2005
This collection contains the following selected articles describing research conducted with support by US Army TACOM through the Automotive Research Center, a US Army Center of Excellence in Modeling and Simulation of Ground Vehicles, a Dual-Use Science and Technology Project, the General Motors Collaborative Research Center at the University of Michigan, two grants from the Ford Motor Company and a fellowship from the Department of Mechanical Engineering at the University of Michigan:


Optimal Design Laboratory
University of Michigan
May 2005
Multicriteria Optimization in Product Platform Design

A product platform is a set of common components, modules or parts from which a stream of derivative products can be created. Product platform design requires selection of the shared parts and assessment of the potential sacrifices in individual product performance that result from parts sharing. A multicriteria optimization problem can be formulated to study such decisions in a quantitative manner at the product performance level. Studying the Pareto sets that correspond to various derivative products leads to a systematic methodology for design decision making. Design of a nail gun platform is used to illustrate the concepts presented.

1 Introduction

Traditional design processes typically address design of a single product. However, manufacturing firms increasingly make several variations of a product, each directed to a different market niche. There is even some evidence to suggest a firm must have an appreciable product line to attain market share, stay competitive, and remain profitable (see, for example, [1–3]).

In a product line, products are likely to be related in some way. Exactly how the similarities among products exhibit themselves can either simplify or complicate a number of business issues. For example, if it is possible to manufacture two different products using some of the same parts, a company often sees benefits through reduction of inventory [4,5], reduction in the proliferation of different parts [6], reduction in the design lead-time for products [7], ease of designing for new market niches [3], and reduction in the number and type of factory machinery and processes [8]. Penalties may be incurred by the sharing of components, however. Components used in multiple products must be designed to the criteria of the most demanding product—which may result in a component that far exceeds the requirements of the other models in which it is used. This sharing may also result in a lack of variation across the product family [9].

This is the idea behind a product platform, namely, a set of common components, modules, or parts from which a stream of derivative products can be efficiently created and launched. Several researchers have addressed interesting aspects of product platforms. For example, Gonzalez-Zugasti et al. [10] consider the market niche as well as the platform cost. Ishii et al. [11] look to minimize life-cycle cost while still providing a broad product line. In the automotive field MacDuffie et al. [12] examined overall factory performance, while Siddique et al. [8] have studied the effects of using common assembly methods and factory lines as well as common parts.

In this article, the focus is on using nonlinear programming methods for product platform design. In particular, optimization is used as a method for designing several individual products for market niches. The argument is made that if two or more products are to share the same product, the performance of each product within its own niche will likely change in comparison to its performance if it were designed to not share any parts. Optimization formulations can quantify this change in performance, which is one of the criteria used to justify or deny the use of the common part. Furthermore, if the common part is to be used in two or more products, the design objectives of the two products will often be in competition—leading to the use of multicriteria optimization.

A design example is introduced early in the body of the paper (rather than as an example at the end). This motivates the development of the methodology and illustrates its application. The product is a nail gun (Fig. 1), a device used in construction or woodworking projects to drive nails into wood. A trigger activates the hammer mechanism which drives the nails with an appropriate force. A model for the performance of a nail gun is given in Section 2. In Section 3 the model is used to formulate a multicriteria optimization problem representing a possible platform design. The general approach for quantifying the change in performance due to the introduction of common parts is presented in Sections 4 and 5. Some conclusions are offered in Section 6.

2 Modeling a Nail Gun

In the present design example we focus on the spring-hammer mechanism using the representation in Fig. 2 to derive the mathematical model. A potential vector of design variables is \( \mathbf{x} = [l_h, l_u, l_p, N] \) where \( l_h \) is the starting height of the hammer, \( l_u \) is the length of the nail, \( l_p \) is the preloaded length of the spring, and \( N \) is the number of active spring coils. The remaining dimensions are considered as parameters, namely, the wire diameter \( d \), spring diameter \( D \), preloaded length \( l_p \), and hammer mass \( m_h \). The spring constant of the spring within the hammer mechanism is given by

\[
k_s = \frac{d^4 G}{8D^2 N},
\]

where \( G \) is the shear modulus of the wire. The maximum shear stress in the spring is given by

\[
\tau = \frac{8FD}{\pi d^3},
\]

where \( F \) is the maximum force in the spring

\[
F = k_s [(l_u - (l_p - l_h))].
\]

The energy used to deliver the nail is the energy released from the spring during its travel:

\[
E = \frac{1}{2} k_s [(l_u - (l_p - l_h))^2 - (l_u - l_p)^2].
\]

Additionally, the solid height of the spring cannot be less than the height of the spring at its maximum compression:

\[
Nd - (l_p - l_h) \leq 0.
\]

The positions of the nail \( x_n \) and nail gun \( x_i \) are modeled by three sets of differential equations. The first set governs the acceleration of the hammer until it reaches the end of its travel

\[
\ddot{x}_n = -F_s / (m_h + m_n)
\]
The recoilled experienced by the user is the maximum height reached by the tool. The depth to which the nail is driven is the minimum value of the height reached by the nail head:

\[ l_r = \max(x_i) \]  
\[ l_d = \min(x_i) \]  

3 Formulating an Optimization Problem

When targeting a particular market niche, the performance criteria modeled in Section 2 are used to formulate a design optimization problem. However, if two products are meant to target two separate niches, the respective optimization problems may have different objective functions and constraints, along with different limits and parameters assigned to the corresponding constraints.

For example, suppose that a nail gun manufacturer wishes to create a product family consisting of two separate nail guns. Nail gun A will be an industrial-quality gun for use by professional carpenters, and nail gun B will be a less expensive entry-level model for use by the typical weekend enthusiast. Since model A is the flagship model there is a premium placed on performance, so the objective for model A is to maximize the size of the nail that can be driven into the wood. Model B, intended for the casual user, prescribes a smaller nail and puts a premium on user comfort. Its objective function is to minimize the recoill the user experiences. Two independent optimal design problems can be stated. The model for Product A is

\[
\text{minimize} \quad x_A \quad -m_{n,A} \]  
\[ \text{subject to} \quad l_{h,A} - l_{n,A} \geq 0 \]  
\[ l_{d,A} - l_{d_{min,A}} \geq 0 \]  
\[ \tau_A - S_u \geq 0 \]  
\[ N_A d_A - (l_{p,A} - l_{h,A}) \leq 0. \]  

The maximum nail size \( (m_{n,A}) \) is multiplied by \(-1\) in the objective to keep a common formalism of minimizing the objective. The model for Product B is

\[
\text{minimize} \quad x_B \quad l_{r,B} \]  
\[ \text{subject to} \quad l_{h,B} - l_{n,B} \geq 0 \]  
\[ l_{d,B} - l_{d_{min,B}} \geq 0 \]  
\[ \tau_B - S_u \geq 0 \]  
\[ N_B d_B - (l_{p,B} - l_{h,B}) \leq 0. \]  

The different parameter values in Eqs (14) and (15) are given in Table 1. Both models (14) and (15) are specific cases of the general optimal design problem

\[
\text{minimize} \quad f(x) \]  
\[ \text{subject to} \quad g(x) \leq 0 \]  
\[ h(x) = 0. \]  

When making comparisons between two platforms, the best designs possible in one platform must be compared to the best designs possible in another platform. For instance, suppose that the internal spring mechanism housing is a candidate common part for both nail guns, requiring a consistent starting height \( l_h \). This prompts the following question:

Does the use of the same \( l_h \) in both designs significantly degrade the performance of the individual products?

In order to answer this question, one must compare the best possible designs when sharing parts to the best possible designs when the products are not sharing parts. In the present discussion a product platform is a specific set of shared parts among the
different products. To model a product platform, the different optimization problems are combined into a single multicroteria optimization problem:

\[
\text{minimize} \quad f_i(x_i) \quad i = 1 \ldots p
\]

subject to

\[
g_i(x_i) \leq 0 \quad i = 1 \ldots p
\]

\[
h_i(x_i) \leq 0 \quad i = 1 \ldots p
\]

\[
x_n, k_i = x_n, k_j \quad (k_1, k_2) \in P_{ij} \quad i, j = 1 \ldots p
\]

\[
i < j.
\]

(17)

The set \( P_{ij} \) is a set of index pairs used to represent the equality constraints associated with parts sharing. If products \( i \) and \( j \) share some of the same parts, then \( P_{ij} \) contains the index pairs of the design variables describing the common parts, effectively enforcing the part to be the same for both products \( i \) and \( j \). Thus a platform configuration is defined by a distinct set of index pairs. To compare two different product platforms, the solutions to model (17) are compared with different sets of index pairs, say \( \{P_{ij}\} \) and \( \{Q_{ij}\} \).

To illustrate this idea consider again two nail guns, \( A \) and \( B \), designed such that they use the same spring-resetting mechanism, allowing a common housing. The variable describing the starting height of the hammer \( l_h \) is forced to be the same through the use of equality constraints. The two separate optimal design problems modeled in (14) and (15) are combined to form the multicriteria optimization problem

\[
\text{minimize} \quad f_A(x) = -m_{n,A}
\]

\[
f_B(x) = l_r, B
\]

subject to

\[
l_h, A = l_h, B \geq 0
\]

\[
l_d, A \geq l_d, \text{min}, A
\]

\[
\tau_s \leq \alpha
\]

\[
N_A d_A - (l_{p,A} - l_{h,A}) \leq 0
\]

\[
l_{h,B} = l_{h,B} \geq 0
\]

\[
l_{d,B} = l_{d,B} \geq 0
\]

\[
\tau_s \leq \alpha
\]

\[
N_B d_B - (l_{p,B} - l_{h,B}) \leq 0
\]

\[
l_{h,A} = l_{h,B}
\]

(18)

The equality constraint \( l_{h,A} = l_{h,B} \) is represented by a set of index pairs containing one element, \( P_{AB} = \{ (1,1) \} \), because \( l_h \) is the first variable in \( x_A \) and \( x_B \). The platform consisting of \( \{P_{AB}\} \) can be compared with the null platform where no parts are shared. The null platform is denoted by \( \{O_{ij}\} \) where \( O_{ij} = \{ \} \) for every \( i \) and \( j \).

For a specific \( P_{ij} \) set the solutions of model (17) and, therefore, model (18) form a Pareto set, defined such that for each point in the Pareto set it is not possible to improve the objective function of one product without making the other objective worse. Pareto optimality has been extensively studied, so no background will be presented here. For extensive reviews of its use in design the reader is referred to the books by Eschenauer et al. [13], Ouyang [14], and Statnikov and Matusov [15], or to articles by Freudenthal and Rao [16], and Koski [17]. Because Pareto optimality gives a set of solutions rather than a unique solution, the next section discusses the limits of the change in performance by defining individual minima and bounds for the Pareto set.

### 4 Bounding the Pareto Optimal Solutions

Before determining the Pareto set for a particular platform, bounds can be placed on the performance of the products within the platform.

As a convention, different superscripts will represent optimal values from different platform configurations. For the nailer example a superscript circle represents optimal quantities for the null platform \( (f_A^* \text{ and } f_B^*) \). A superscript bullet \( (f_A^• \text{ and } f_B^•) \) represents optimal quantities for the platform with the common parts. The individual minima \( f_A^* \) of Eq. (17) are defined as the extreme values of the Pareto set. These are the solutions to Eq. (17) with only one of the scalar functions \( f_A \text{ or } f_B \) used as an objective.

There are three possible designs, \( (f_A^*, f_B)$, \( (f_A^*, f_B^•) \), and \( (f_A^•, f_B^•) \). An additional fictitious design \( (f_A^•, f_B^•) \) called the utopia point, is also considered. Plotted together in Fig. 3, the four designs bound the Pareto set and provide a means of visualizing the cost of commonality. In most instances, the utopia point (an idealized best design under the commonality constraint) is outperformed by the null platform (which has no such constraint). In other words, forcing two products to share parts changes both designs, and no solution to model (18) can be better than the solutions to the separate optimal design problems. This leads to a general statement about product platforms.

If there are two sets of index pairs \( \{P_{ij}\} \) and \( \{S_{ij}\} \) such that \( R_{ij} \subseteq S_{ij} \) for each \( i \) and \( j \), then the feasible space for platform \( S \) cannot be larger than the feasible space for platform \( R \). Therefore the performance for platform \( R \) will be at least that of platform \( S \) as measured by the objective function values.

In fact, the solution to the separate optimal design problems (the null platform) will typically be better than the utopia point of Eq. (18), and the designer of a product platform should expect to give up some acceptable amount of performance. The magnitude of this sacrifice is an indication of the cost of commonality.

This is important for three reasons. First, by defining the individual minima, the limits of the Pareto set and therefore the changes in performance are known. Second, simply quantifying the change in performance is useful to justify further investigation. If there is too much degradation in performance (i.e.,

### Table 1 Parameter values for the optimal design models in Eqs. (14) and (15)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Units</th>
<th>Prod. A</th>
<th>Prod. B</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_a )</td>
<td>maximum shear stress</td>
<td>Pa</td>
<td>6.4E8</td>
<td></td>
</tr>
<tr>
<td>( G )</td>
<td>shear modulus of wire</td>
<td>Pa</td>
<td></td>
<td>1.235E11</td>
</tr>
<tr>
<td>( d )</td>
<td>wire diameter</td>
<td>m</td>
<td>0.002</td>
<td></td>
</tr>
<tr>
<td>( D )</td>
<td>coil diameter</td>
<td>m</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>( l_p )</td>
<td>preloaded length</td>
<td>m</td>
<td></td>
<td>0.12</td>
</tr>
<tr>
<td>( l_u )</td>
<td>unstretched length</td>
<td>m</td>
<td>0.139</td>
<td>0.141</td>
</tr>
<tr>
<td>( N )</td>
<td>number of coils</td>
<td></td>
<td>25</td>
<td>31</td>
</tr>
<tr>
<td>( k_p )</td>
<td>spring constant of user’s arm</td>
<td>N/m</td>
<td>25</td>
<td>10</td>
</tr>
<tr>
<td>( m_t )</td>
<td>mass of other components of gun</td>
<td>kg</td>
<td>2.5</td>
<td>2.0</td>
</tr>
<tr>
<td>( m_{na} )</td>
<td>mass of the longest nail to be driven</td>
<td>kg</td>
<td>3.95E-3</td>
<td>4.29E-4</td>
</tr>
<tr>
<td>( \beta )</td>
<td>parameter determining nail-wood interaction</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The primary focus here is on the investigation and comparison of a number of possible combinations of parts. Each possible configuration has an associated set of index pairs. For example, many

5 Using the Pareto Set

There are two general approaches used to make single design decisions in multicriteria optimization. In the first approach, single points on the Pareto set are found based on a decision-maker’s a priori preferences and trade-off decisions. In the second approach, the entire set is computed first and the decision-maker uses it to establish a posteriori preferences. In order to explore the trade-offs presented by different potential platform designs, automated methods that capture the complete Pareto set are more appropriate here. The reader is referred to the theses by Athan [18] or Das [19] for synopses of methods.

The Pareto set for the nailer example is given in Fig. 5. Notice that the utopia point \((f'_A, f'_B)\) is not part of the Pareto set. Only when there is no trade-off between designs does the utopia point belong to the Pareto set. Therefore, a final decision of whether or not to use the common part should not be made until the Pareto set (or at least part of the Pareto set) has been explored.

Some of the benefits of using a platform have not been explicitly taken into account in the present formulation. It is inherently difficult to quantify considerations such as the competitive advantage via faster product development or the reduced inventory, tooling, production costs and factory floor-space via smaller variety of parts. The problem is made more complex by the fact that sharing some parts may be more beneficial than sharing others, and the benefit is usually measured on the company level instead of the product performance level. The reader is referred to Meyer and Lehnerd [3] who make arguments for competitive advantages as well as Fujita et al. [20] who use simple models for quantifying these advantages monetarily.

Regardless, the discussion and example presented here is suggestive of a general method consisting of the following six steps:

1. Identify a set of parts that could be shared between two or more products. For each possible pair of products, assign a set of index pairs, i.e., \(\{O_i\}, \{P_i\}, \{Q_i\}, \ldots, \{Y_i\}, \{Z_i\}\).

2. Formulate the multicriteria design problem as modeled in Eq. (17).

3. Determine the individual optima (i.e., extreme points) of the multicriteria design problem for each possible configuration.

4. Use the individual optima for each configuration to decide if investigating the Pareto set of that configuration is worthwhile.

5. For each combination of common parts that has an acceptable degradation in performance calculate the Pareto set (or an appropriate approximation of it).

6. From the candidate Pareto sets choose the design that offers the best value for all appropriate products while still allowing for the benefits of having a flexible product platform.

The primary focus here is on the investigation and comparison of a number of possible combinations of parts. Each possible configuration has an associated set of index pairs. For example, many
product platforms are possible from the nail gun model, such as a common spring-housing, common spring-wire, common power source, or any combination of these. Using the methodology presented above two more Pareto curves can be generated, as shown in Fig. 6. One curve investigates the effect of having a common power source across the product platform. The nature of this curve is quite different from that obtained in the previous example. Note that when the spring-resetting mechanism is held in common, as in the previous example, the performance trade-off between the two modes (or objective functions) is nearly linear—there is no point on the curve that is significantly nearer the null platform than any other. That is not the case when the power source is the common part. Although the individual optima \( f_A \) and \( f_B \) are in nearly the same location as the previous example, the curve dips in towards the null platform/utopia point. The area nearest the null platform/utopia point, the “knee,” represents the portion of the design space where the least amount of performance is sacrificed to commonality (see Das [21] for an in-depth discussion of this region).

A third Pareto curve is created to represent the performance when both the spring-resetting mechanism and the power source are common across the platform. Note that the knee that existed in the power-source curve no longer exists. Indeed, the constraint inducing most of the performance reduction, namely that of the common spring-resetting mechanism, dominates the response. The relationship between this last curve and the first two is similar to that between the utopia point and the null platform: when a new commonality constraint is introduced, the performance of the product cannot improve—at best it remains the same.

The optimal platform design should lie in one of these Pareto sets, but exactly which Pareto set is “the best” is a question of performance as well as of other business issues. Defining the product platform configuration is a combinatorial problem because it requires searching through several possible configurations and making decisions based on the Pareto sets of each configuration.

6 Conclusion

A product platform, being a set of common components, modules or parts from which a stream of derivative products can be efficiently created and launched, is becoming a popular business practice today. The main premise in this paper was that a product platform can be formulated as a multicriteria optimization problem. In so doing, it has been shown that the performance of the products within the platform will degrade, and that amount given up in performance can be quantified. The designer of a product platform should therefore expect to make compromises that can be rigorously studied through a multicriteria optimization process.

Acknowledgments

This research has been partially supported by the Automotive Research Center at the University of Michigan, a US Army Center of Excellence in Modeling and Simulation of Ground Vehicles, under Contract No. DAAE07-98-R-L008, and by a graduate student fellowship from the Department of Mechanical Engineering at the University of Michigan. This support is gratefully acknowledged.

Nomenclature

\[ A = \text{Subscript denoting product A} \]
\[ B = \text{Subscript denoting product B} \]
\[ f = \text{Objective function} \]
\[ f^* = \text{The optimal value of the objective functions for the null platform} \]
\[ f^* = \text{The optimal value of the objective functions for a given platform} \]
\[ g = \text{Vector of inequality constraints} \]
\[ h = \text{Vector of equality constraints} \]
\[ i = \text{Subscript index denoting a particular product} \]
\[ k = \text{Design variable index used to define a platform} \]
\[ n = \text{Number of products in a particular platform} \]
\[ x = \text{Vector of design variables} \]
\[ \{P_{ij}\}, \{Q_{ij}\}, \ldots = \text{Index set denoting a specific product platform} \]

References


Extension of the target cascading formulation to the design of product families

M. Kokkolaras, R. Fellini, H.M. Kim, N.F. Michelen, and P.Y. Papalambros

Abstract The target cascading methodology for optimal product development is extended to product families with predefined platforms. The single-product formulation is modified to accommodate the presence of shared systems, subsystems, and/or components and locally introduced targets. Hierarchical optimization problems associated with each product variant are combined to formulate the product family multicriteria design problem, and common subproblems are identified based on the shared elements (i.e., the platform). The solution of the overall design problem is coordinated so that the shared elements are consistent with the performance and behaviour of the product variants. A simple automotive design example is used to demonstrate the proposed methodology.

Key words product platforms, product families, commonality, target cascading, systems design optimization

1 Introduction

Product platforms enable rapid enrichment of a product portfolio to meet changing market needs while keeping design and manufacturing cycle times and costs low (Meyer and Lehnerd 1997; Ericsson and Erixon 1999). According to Siddique et al. (1998), McGrath’s description of a product platform for high-tech products (McGrath 1999) can be summarized as a collection of the common elements, especially the underlying core technology, implemented across a range of products. Meyer and Lehnerd (1997) elaborated on the term underlying core technology by defining the product platform as the set of parts, interfaces, and manufacturing processes that are shared among a set of products and allow the development of derivative products with cost and time savings. In this manner, it is emphasized that a platform does not only refer to common physical components but also to a common manufacturing and assembly procedure.

A family of products is developed based upon a product platform. Meyer and Lehnerd (1997) define a product family to be a set of individual products that share common technology and address a related set of market applications. It can be argued that the relation between the product platform and the product family is governed by the product architecture. Product architecture is defined by Ulrich and Eppinger (1995) as (a) the arrangement of functional elements, (b) the mapping from functional elements to physical components, and (c) the specification of the interactions among interacting physical components. The same authors classify two main categories of product architectures according to the mapping of functions to components: integral and modular.

Meyer and Lehnerd (1997) discuss strategies for developing robust product platforms by defining the market segmentation grids and then leveraging product design over these segments. They also provide metrics for measuring the platform’s efficiency, development cycle-time efficiency, commercial effectiveness, and cost price ratio. Ericsson and Erixon (1999) focus on modular product platforms, propose a modular function deployment (MFD) method based on the quality function deployment (QFD) technique, and demonstrate the use of the latter in, among others, automotive industry applications. The MFD method provides a means for identifying the optimal number of parts that the modules should consist of. Moreover, metrics and rules are proposed for evaluating the effects of modularity in terms of development lead time, costs, and capacity, product and systems costs, quality, variant flexibility, upgrading, and recyclability.

Commonality and differentiation indices have been also proposed by Martin and Ishii (1997) within the framework of their design for variety methodology. In their approach, the authors take into account manu-
facturing and propose late-point differentiation concepts for efficient product platform design. In their latest work, Martin and Ishii (2000) differentiate between variety within the current product line and variety across generations of products. Kota and Sethuraman (1998) present a product line commonality index and present a method for benchmarking product families.

Simpson et al. (1999) propose a method for the synthesis and exploration of product platform concepts based on the market segmentation grid and lever aging/scaling concepts of Meyer and Lehnerd (1997) and the solution of compromise decision support problems using goal programming. Conner et al. (1999) discuss the fact that many metrics proposed by other researchers are accounting measures for efficiency rather than engineering measures for performance. They propose a method for evaluating platform design alternatives and tradeoffs between commonality and individual product performance by means of non-commonality and performance deviation indices and goal programming. However, they recognize that their method makes possibly misleading assumptions in regard to manufacturing and assembly criteria.

Gonzalez-Zugasti et al. (1998) formulate product platform design as a general optimization problem taking into account both performance requirements and cost, and implement the latter as an interactive negotiation model. They recognize some flaws in their method, namely, the subjective choice of weights when defining the objective function, the lack of tradeoffs evaluation, and the possibly vast size of the combinatorial problem. In an extension of that work, Gonzalez-Zugasti and Otto (2000) present a method for determining design specifications of modules and their best combination, assuming a fixed architecture. They account for family, individual, sharing, and compatibility constraints, and use genetic algorithms for solving the general optimization problem. In a similar approach, Pujita et al. (1998) propose a framework for assessing optimality and sensitivity of product platforms including cost models for development and production. They consider a quite simple application and emphasize the need for further developing the mathematical approach.

Siddique et al. (1998) examine the applicability of product variety concepts to automotive design. In particular, they investigate whether product variety design concepts such as standardization, delayed differentiation, modularity, module interfaces, robustness, and mutability can be utilized. Keeping in mind that they limit their consideration for platform only to the underbody structure of a vehicle, they come to the conclusion that some of these concepts cannot be applied, mainly because of the integral nature of its architecture. However, they do mention the possibility of partitioning the underbody platform into major manufacturable and assembly modules.

The research work presented so far refers to efforts for developing methods and tools for designing and evaluating product platforms and families. Some are philosophical frameworks that address design issues in a heuristic manner, and some are more mathematical in nature. The development of a product family depends on an appropriate design strategy that addresses commonality, i.e., the platform. Moreover, different product variants are typically characterized by conflicting performance criteria. A multicriteria optimization formulation can identify the associated tradeoffs by means of Pareto surfaces, with commonality constraints used to define the common elements of the various product variants (Nelson et al. 1999; Fellini et al. 2000). Design decisions can then be made based on subjective engineering, marketing, and manufacturing criteria.

Although contributing significantly to the development of design methodologies for product platforms and families, none of the aforementioned methods can both evaluate tradeoffs between family and individual design targets and determine design specifications for the components, given a predefined platform architecture. The target cascading methodology, extended and applied to product family design, aims at filling this gap.

The article is organized as follows. The target cascading methodology for single optimal product development is introduced in the next section, extended for product family development in Sect. 3, and implemented for a family of vehicles in Sect. 4. Results are presented and discussed in Sect. 5, and conclusions are drawn in Sect. 6.

2 Target cascading for optimal product development

The development of any complex product is strongly associated with setting and enforcing proper specifications for each of the product’s attributes. Analytical target cascading (Kim 2001) is a methodology for the design of large engineering systems at the early product development stages. First, the design problem is partitioned into a hierarchical set of subproblems associated with systems, subsystems, and components. Design specifications (or targets) are defined at the top level of the multilevel design formulation and “cascaded down” to lower levels. Design subproblems are formulated at each level so that components, subsystems, and systems are designed to match the cascaded targets consistent with the overall system targets. The main benefits of target cascading are reduction in design-cycle time, avoidance of design iterations late in the development process, and increased likelihood that physical prototypes will be closer to production quality. Target cascading also facilitates concurrency in system design: Once targets are identified for systems, subsystems, and components, these elements can be isolated and designed in detail independently, allowing the outsourcing of subsystems and components to suppliers. Target cascading offers a robust framework for multilevel design and has been demonstrated to be convergent,
whereas other similar problem formulations exhibit convergence difficulties.

The analytical target cascading process was presented by Kim et al. (2000) in the context of automotive engineering systems. In this article a more general notation is introduced, from which the design problem for each element (i.e. system, subsystem, or component) can be recovered as a special case. Moreover, the formulation presented herein allows for design specifications to be introduced not only at the top level for the overall product, but also “locally” to account for individual system, subsystem, and/or component requirements. To represent the hierarchy of the partitioned design problem, the set \( \mathcal{E}_i \) is defined at each level \( i \), in which all the elements of the level are included. For each element \( j \) in the set \( \mathcal{E}_i \), the set of children \( \mathcal{C}_{ij} \) is defined, which includes the elements of the set \( \mathcal{E}_{i+1} \) that are children of the element. An illustrative example is presented in Fig. 1. At level \( i = 1 \) of the partitioned problem we have \( \mathcal{E}_2 = \{A, B\} \), and for element “B” on that level we have \( \mathcal{C}_{2B} = \{C, D\} \).

![Fig. 1 Example of single product hierarchically partitioned design problem](image)

There are two types of responses: responses \( \bar{R} \) linked to “local” targets (e.g. at the top level), and responses \( R \) linked to “cascaded” targets, i.e. linking two successive levels in the problem hierarchy. The design problem \( P_{ij} \) corresponding to the \( j \)-th element at the \( i \)-th level is formulated as follows:

\[
\min_{x_{ij}} \left\| \tilde{R}_{ij} - T_{ij} \right\| + \left\| R_{ij} - R^\ell_{ij} \right\| + \left\| y_{ij} - y^\ell_{ij} \right\| +
\]

\[
\epsilon^R_{ij} + \epsilon^y_{ij}
\]

subject to

\[
\sum_{k \in \mathcal{C}_{ij}} \left\| R_{(i+1)k} - R^L_{(i+1)k} \right\| \leq \epsilon^R_{ij}
\]

\[
\sum_{k \in \mathcal{C}_{ij}} \left\| y_{(i+1)k} - y^L_{(i+1)k} \right\| \leq \epsilon^y_{ij}
\]

\[
g_{ij}(\tilde{R}_{ij}, x_{ij}, y_{ij}) \leq 0, \quad h_{ij}(\tilde{R}_{ij}, x_{ij}, y_{ij}) = 0 \tag{1}
\]
presented by Kim (2001) under the assumption that (1) is convex. Convergence properties of analytical target cascading will be presented by Park et al. (2001).

3 Target cascading for product family development

When designing a family of platform-based products, one must identify the tradeoffs that are a consequence of shared systems, subsystems, and/or components. These tradeoffs exist because the products are no longer optimized for their individual performance, but for the family as a whole. The shared elements of the product family influence the performance of the individual products.

A Pareto-based approach that quantifies the aforementioned tradeoffs was proposed by Nelson et al. (1999). The first step in this method is to formulate individual design problems for each product. The problems are then combined into one family problem using a multicriteria optimization formulation that includes the individual product requirements. By introducing an equality commonality constraint one can specify the product elements to be shared. Solving the multicriteria problem for the Pareto set one can then visualize the design tradeoffs, and choose a suitable design.

In order to exploit the product family structure, the design problem is formulated as a hierarchical optimization problem, as investigated by Fellini et al. (2000). The top level problem addresses family attributes, while lower levels address attributes associated with particular elements. To achieve this within the modified target cascading formulation presented in this article, family targets are defined at the top level and locally introduced targets are defined at lower levels (e.g. the product level) to satisfy individual requirements. Given a set of family and individual product targets, analysis models for all design elements, and a predefined platform, targets are cascaded to elements lower in the hierarchy (i.e. system, subsystem, and component targets are determined).

The rest of this section discusses modifications to the single-product target cascading formulation necessary for the design of product families. These modifications enable subproblems to return design response and linking variable values to multiple parents. Figure 2 illustrates a simple example of a product family with two product variants; each variant is partitioned into two systems, and the two variants share one system.

Desired product family design specifications are defined as targets at the top level and cascaded to lower levels. Product variant targets are introduced at the product level. Design subproblems are formulated at each level so that components, subsystems, and systems are designed to match the cascaded targets while the overall system is consistent. To allow for elements to be shared (i.e. to have multiple parents), the set of parents $P_{ij}$ is defined for each element $j$ of the set $E_i$ at every level $i$ this set includes the elements of the set $E_{i-1}$ that are parents of this element. The design problem $P_{ij}$ for the $j$-th element at the $i$-th level is reformulated as follows:

$$\min_{x_{ij}} \| \hat{R}_{ij} - T_{ij} \| + \sum_{q \in P_{ij}} \| R_{ij} - R_{ijq}^U \| + \sum_{q \in P_{ij}} \| y_{ij} - y_{ijq} \| + \epsilon_{ij}^R + \epsilon_{ij}^g$$

subject to

$$\sum_{k \in C_{ij}} \| R_{(i+1)k} - R_{(i+1)k}^L \| \leq \epsilon_{ij}^R$$

$$\sum_{k \in C_{ij}} \| y_{(i+1)k} - y_{(i+1)k}^L \| \leq \epsilon_{ij}^g$$

$$g_{ij}(\hat{R}_{ij}, x_{ij}, y_{ij}) \leq 0, \quad h_{ij}(\hat{R}_{ij}, x_{ij}, y_{ij}) = 0,$$  \hspace{1cm} (2)

with $P_{ij} = \{q_1, \ldots, q_{p_{ij}}\}$, where $p_{ij}$ is the number of parent elements, and where

- $R_{ijq}^U \in \mathbb{R}^{d_{ij}}$ is the vector of response values cascaded to the element from its $q$-th parent,
- $y_{ijq}^U \in \mathbb{R}^{l_{ij}}$ is the vector of linking design variable values cascaded to the element from its $q$-th parent.

It can be readily shown that the target cascading formulation for optimal single product design shown in (1) is recovered if all the sets of parents $P_{ij}$ consist of only one element.

4 Case study

To illustrate the use of target cascading for designing a family of products, a multi-vehicle design problem has

![Fig. 2 Example of product family hierarchically partitioned optimal design problem](image-url)
been formulated. The model hierarchy, depicted in Fig. 3, consists of four levels: the family (top) level, the vehicle level, the system level, and the component level. At the family level an objective is defined that combines the masses \(m_{v,A}\) and \(m_{v,B}\) of two vehicle variants. More emphasis is given to minimizing the mass \(m_{v,A}\) of vehicle A.

Figure 4 illustrates the overall target cascading formulation and coordination. The individual vehicles are modelled as half-cars at the vehicle level. In addition to the mass targets \(m_{v,A}^U\) and \(m_{v,B}^U\) cascaded from the family level, local targets \(T_A\) and \(T_B\) are set for the ride quality \(z_{v,A}\) and \(z_{v,A}\) of each variant, respectively. Ride quality is defined by the following five responses: front and rear ride frequency, front and rear wheel hop frequency, and under-steer gradient. Vehicle A should have a stiffer ride, whereas vehicle B should have a softer ride. The half-car model computes vehicle mass \(m_v\), ride quality metrics \(z_v\), body-in-white mass \(m_b\), and suspension stiffnesses \(k_{sf}\) and \(k_{sr}\). Vehicle responses must meet targets determined at the family and system levels (denoted by superscripts \(U\) and \(L\), respectively) and local targets \(T_A\) and \(T_B\). Once the responses are computed, they are used as targets at the family and system levels (denoted by superscripts \(L\) and \(U\), respectively).

At the component level, each component \(i\) of the body comprising the platform is designed to match the area targets \(A_{i}^{U}\) and \(A_{hi}^{U}\) and moment of inertia target.
 targeted from the system level by determining optimal combinations of cross-sectional dimensions (width $b_i$, height $h_i$, and thickness $t_i$). Once these dimensions are found, analytical expressions are evaluated and optimal values $A_{L,i}, A_{R,i},$ and $I_{L,i}$ are passed to the system level for each platform component $i$. The product platform for the family consists of three body components: namely, the roof, rocker, and hinge pillar. To implement the product platform:

$$\min \begin{align*}
\| (0.8 \cdot m_{v,A} + 0.2 \cdot m_{v,B}) - T_F \| + \epsilon_R + \epsilon_Y \\
\text{w.r.t.} \quad m_{v,A}, m_{v,B}, k_{sf,A}, k_{sf,B}, \epsilon_R, \epsilon_Y \\
\| m_{v,A} - m_{v,A}^L \| + \| m_{v,B} - m_{v,B}^L \| \leq \epsilon_R \\
\text{s.t.} \quad \| k_{sf,A} - k_{sf,A}^L \| + \| k_{sf,B} - k_{sf,B}^L \| \leq \epsilon_Y
\end{align*}$$

### Product Family

**Vehicle A**

$$\min \begin{align*}
\| z_{v,A} - T_A \| + \| m_{b,A} - m_{b,A}^L \| + \| k_{sf,A} - k_{sf,A}^L \| + \epsilon \\
\text{w.r.t.} \quad m_{b,A}, k_{sf,A}, k_{sr,A}, k_{ur,A}, k_{sf,A}, k_{sr,A}, k_{ur,A}, \epsilon \\
\text{s.t.} \quad \| m_{b,A} - m_{b,A}^L \| + \| k_{sf,A} - k_{sf,A}^L \| + \| k_{sr,A} - k_{sr,A}^L \| \leq \epsilon
\end{align*}$$

**Vehicle B**

$$\min \begin{align*}
\| z_{v,B} - T_B \| + \| m_{b,B} - m_{b,B}^L \| + \| k_{sf,B} - k_{sf,B}^L \| + \epsilon \\
\text{w.r.t.} \quad m_{b,B}, k_{sf,B}, k_{sr,B}, k_{ur,B}, k_{sf,B}, k_{sr,B}, k_{ur,B}, \epsilon \\
\text{s.t.} \quad \| m_{b,B} - m_{b,B}^L \| + \| k_{sf,B} - k_{sf,B}^L \| + \| k_{sr,B} - k_{sr,B}^L \| \leq \epsilon
\end{align*}$$

### Front Suspension A

$$\begin{align*}
\min \| k_{ff,A} - k_{ff,A}^U \| \\
\text{w.r.t.} \quad k_{sf1,A}, k_{sf2,A} \\
\text{s.t.} \quad k_{sf1,A} = 2 \cdot k_{sf2,A}
\end{align*}$$

### Rear Suspension A

$$\begin{align*}
\min \| k_{fr,A} - k_{fr,A}^U \| \\
\text{w.r.t.} \quad k_{fr1,A}, k_{fr2,A} \\
\text{s.t.} \quad k_{fr1,A} = 2 \cdot k_{fr2,A}
\end{align*}$$

### Body A

$$\begin{align*}
\min \{ \delta_{k,A} - T_{b,A} \} + \| m_{b,A} - m_{b,A}^L \| + \epsilon \\
\text{w.r.t.} \quad A_{k,A}, \ldots, A_{k,3}, A_{R1,A}, \ldots, A_{R3,A}, l_{1,A}, \ldots, l_{3,A}, \epsilon \\
\text{s.t.} \quad \sum_{i=1}^{3} \| A_{l_i,A} - A_{l_i,A}^U \| + \| A_{b_i,A} - A_{b_i,A}^U \| + \| l_{i,A} - l_{i,A}^U \| \leq \epsilon
\end{align*}$$

### Body B

$$\begin{align*}
\min \{ \delta_{k,B} - T_{b,B} \} + \| m_{b,B} - m_{b,B}^L \| + \epsilon \\
\text{w.r.t.} \quad A_{k,B}, \ldots, A_{k,3}, A_{R1,B}, \ldots, A_{R3,B}, l_{1,B}, \ldots, l_{3,B}, \epsilon \\
\text{s.t.} \quad \sum_{i=1}^{3} \| A_{l_i,B} - A_{l_i,B}^U \| + \| A_{b_i,B} - A_{b_i,B}^U \| + \| l_{i,B} - l_{i,B}^U \| \leq \epsilon
\end{align*}$$

### Shared Body Components (Platform)

$$\begin{align*}
\min \sum_{i=1}^{3} \| A_i - A_i^U \| + \| A_i - A_i^L \| + \| A_{b_i} - A_{b_i}^L \| + \| A_{b_i} - A_{b_i}^U \| + \| l_{i} - l_{i}^L \| + \| l_{i} - l_{i}^U \| \\
\text{w.r.t.} \quad b_1, b_2, b_3, b_4, b_5, t_1, t_2, t_3
\end{align*}$$

**Vehicle B**

$$\begin{align*}
\min \| k_{fr,B} - k_{fr,B}^U \| \\
\text{w.r.t.} \quad k_{fr1,B}, k_{fr2,B} \\
\text{s.t.} \quad k_{fr1,B} = 2 \cdot k_{fr2,B}
\end{align*}$$

**Front Suspension B**

$$\begin{align*}
\min \| k_{fr,B} - k_{fr,B}^U \| \\
\text{w.r.t.} \quad k_{fr1,B}, k_{fr2,B} \\
\text{s.t.} \quad k_{fr1,B} = 2 \cdot k_{fr2,B}
\end{align*}$$

**Rear Suspension B**
uct platform in the target cascading methodology, one common design problem for the three shared components is formulated at the component level. The shared pillars return a common response to the body models of both vehicle variants.

The front suspension is shared between the two vehicles to illustrate the concept of linking variables and increase the complexity of the case study. This sharing is represented by treating the front suspension stiffness \( k_{sf} \) at the vehicle level as a linking variable. This linking variable is coordinated at the family level by computing the suspension stiffness \( k_{sf} \) to match the values \( k_{sf,A}^L \) and \( k_{sf,B}^L \) determined at the vehicle level for each variant. The computed value is then cascaded to both variants at the vehicle level as \( k_{sf}^U \). Note that the front suspension stiffness is also treated as a response at the vehicle level that is cascaded as target to the system level.

To solve the multilevel problem a generic coordination strategy is implemented that starts at the top-most level. Each level is solved in sequence, and then the problems are solved once again by returning to the top level problem. This process is counted as one iteration, and convergence is tested by checking if deviation terms are sufficiently reduced.

### Table 1 Target and optimal values for vehicle level responses

<table>
<thead>
<tr>
<th>Responses ( z_v )</th>
<th>Target value Vehicle A</th>
<th>Optimal value Vehicle A</th>
<th>Target value Vehicle B</th>
<th>Optimal value Vehicle B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front ride frequency [Hz]</td>
<td>1.273</td>
<td>1.160</td>
<td>0.955</td>
<td>1.120</td>
</tr>
<tr>
<td>Rear ride frequency [Hz]</td>
<td>1.592</td>
<td>1.585</td>
<td>1.592</td>
<td>1.592</td>
</tr>
<tr>
<td>Front wheel hop frequency [Hz]</td>
<td>10.345</td>
<td>10.348</td>
<td>10.345</td>
<td>10.343</td>
</tr>
<tr>
<td>Rear wheel hop frequency [Hz]</td>
<td>10.345</td>
<td>10.347</td>
<td>9.549</td>
<td>9.549</td>
</tr>
<tr>
<td>Under-steer gradient [rad/m/s²]</td>
<td>( 7.19 \times 10^{-3} )</td>
<td>( 7.186 \times 10^{-3} )</td>
<td>( 7.19 \times 10^{-3} )</td>
<td>( 7.191 \times 10^{-3} )</td>
</tr>
</tbody>
</table>

### Table 2 Vehicle responses and linking variable values computed at the family and vehicle levels

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Type</th>
<th>Family level value</th>
<th>Vehicle level value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass of vehicle A ( m_{v,A} ) [kg]</td>
<td>Response</td>
<td>2139</td>
<td>2139</td>
</tr>
<tr>
<td>Mass of vehicle B ( m_{v,B} ) [kg]</td>
<td>Response</td>
<td>2162</td>
<td>2163</td>
</tr>
<tr>
<td>Front suspension stiffness of vehicle A ( k_{sf,A} ) [N/mm]</td>
<td>Linking variable</td>
<td>35.400</td>
<td>35.490</td>
</tr>
<tr>
<td>Front suspension stiffness of vehicle B ( k_{sf,B} ) [N/mm]</td>
<td>Linking variable</td>
<td>35.400</td>
<td>35.500</td>
</tr>
</tbody>
</table>

### Table 3 System responses computed at the vehicle and system levels

<table>
<thead>
<tr>
<th>Response</th>
<th>Vehicle level value</th>
<th>System level value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front suspension stiffness of vehicle A ( k_{sf,A} ) [N/mm]</td>
<td>35.400</td>
<td>35.499</td>
</tr>
<tr>
<td>Front suspension stiffness of vehicle B ( k_{sf,B} ) [N/mm]</td>
<td>35.500</td>
<td>35.499</td>
</tr>
<tr>
<td>Rear suspension stiffness of vehicle A ( k_{rf,A} ) [N/mm]</td>
<td>39.860</td>
<td>39.790</td>
</tr>
<tr>
<td>Rear suspension stiffness of vehicle B ( k_{rf,B} ) [N/mm]</td>
<td>36.560</td>
<td>36.617</td>
</tr>
<tr>
<td>Body-in-white mass of vehicle A ( m_{b,A} ) [kg]</td>
<td>240</td>
<td>239</td>
</tr>
<tr>
<td>Body-in-white mass of vehicle B ( m_{b,B} ) [kg]</td>
<td>263</td>
<td>263</td>
</tr>
</tbody>
</table>
Table 4  Platform component responses computed at the system and component levels

<table>
<thead>
<tr>
<th>Response</th>
<th>System level value vehicle A</th>
<th>System level value vehicle B</th>
<th>Component level value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moment of inertia of rocker $I_1$ [in$^4$]</td>
<td>15.387</td>
<td>15.384</td>
<td>15.387</td>
</tr>
<tr>
<td>Footprint cross-sectional area of rocker $A_1$ [in$^2$]</td>
<td>8.024</td>
<td>8.031</td>
<td>8.020</td>
</tr>
<tr>
<td>Real cross-sectional area of rocker $A_{R1}$ [in$^2$]</td>
<td>5.788</td>
<td>5.788</td>
<td>5.792</td>
</tr>
<tr>
<td>Moment of inertia of roof rail $I_2$ [in$^4$]</td>
<td>0.162</td>
<td>0.161</td>
<td>0.157</td>
</tr>
<tr>
<td>Footprint cross-sectional area of roof rail $A_2$ [in$^2$]</td>
<td>1.548</td>
<td>1.542</td>
<td>1.550</td>
</tr>
<tr>
<td>Real cross-sectional area of roof rail $A_{R2}$ [in$^2$]</td>
<td>1.043</td>
<td>1.044</td>
<td>1.043</td>
</tr>
<tr>
<td>Footprint cross-sectional area of hinge pillar $A_3$ [in$^2$]</td>
<td>8.152</td>
<td>8.177</td>
<td>8.158</td>
</tr>
<tr>
<td>Real cross-sectional area of hinge pillar $A_{R3}$ [in$^2$]</td>
<td>5.879</td>
<td>5.888</td>
<td>5.883</td>
</tr>
</tbody>
</table>

Table 5  Optimal values of local design variables at the vehicle, system, and component levels

<table>
<thead>
<tr>
<th>Design variable</th>
<th>Level</th>
<th>Optimal value vehicle A</th>
<th>Optimal value vehicle B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance of CG to front end $c_{gf}$ [m]</td>
<td>Vehicle</td>
<td>1.390</td>
<td>1.250</td>
</tr>
<tr>
<td>Distance of CG to rear end $c_{gr}$ [m]</td>
<td>Vehicle</td>
<td>2.310</td>
<td>2.450</td>
</tr>
<tr>
<td>Front tire stiffness $k_{usf}$ [N/mm]</td>
<td>Vehicle</td>
<td>24.10</td>
<td>24.08</td>
</tr>
<tr>
<td>Rear tire stiffness $k_{usr}$ [N/mm]</td>
<td>Vehicle</td>
<td>24.09</td>
<td>20.52</td>
</tr>
<tr>
<td>Front cornering stiffness $k_{cf}$ [N/rad/10$^{-4}$]</td>
<td>Vehicle</td>
<td>10.47</td>
<td>11.30</td>
</tr>
<tr>
<td>Rear cornering stiffness $k_{cr}$ [N/rad/10$^{-4}$]</td>
<td>Vehicle</td>
<td>12.82</td>
<td>11.84</td>
</tr>
<tr>
<td>Front suspension spring stiffness $k_{sf1}$ [N/mm]</td>
<td>System</td>
<td>23.666</td>
<td>23.673</td>
</tr>
<tr>
<td>Front suspension spring stiffness $k_{sf2}$ [N/mm]</td>
<td>System</td>
<td>11.833</td>
<td>11.836</td>
</tr>
<tr>
<td>Rear suspension spring stiffness $k_{sr1}$ [N/mm]</td>
<td>System</td>
<td>26.526</td>
<td>24.411</td>
</tr>
<tr>
<td>Rear suspension spring stiffness $k_{sr2}$ [N/mm]</td>
<td>System</td>
<td>13.263</td>
<td>12.206</td>
</tr>
<tr>
<td>Width of rocker cross-section $b_1$ [in]</td>
<td>Component</td>
<td>1.510</td>
<td>shared</td>
</tr>
<tr>
<td>Height of rocker cross-section $h_1$ [in]</td>
<td>Component</td>
<td>5.310</td>
<td>shared</td>
</tr>
<tr>
<td>Thickness of rocker cross-section $t_1$ [in]</td>
<td>Component</td>
<td>0.497</td>
<td>shared</td>
</tr>
<tr>
<td>Width of roof rail cross-section $b_2$ [in]</td>
<td>Component</td>
<td>1.335</td>
<td>shared</td>
</tr>
<tr>
<td>Height of roof rail cross-section $h_2$ [in]</td>
<td>Component</td>
<td>1.161</td>
<td>shared</td>
</tr>
<tr>
<td>Thickness of roof rail cross-section $t_2$ [in]</td>
<td>Component</td>
<td>0.265</td>
<td>shared</td>
</tr>
<tr>
<td>Width of hinge pillar cross-section $b_3$ [in]</td>
<td>Component</td>
<td>1.642</td>
<td>shared</td>
</tr>
<tr>
<td>Height of hinge pillar cross-section $h_3$ [in]</td>
<td>Component</td>
<td>4.969</td>
<td>shared</td>
</tr>
<tr>
<td>Thickness of hinge pillar cross-section $t_3$ [in]</td>
<td>Component</td>
<td>0.530</td>
<td>shared</td>
</tr>
</tbody>
</table>

vehicles A and B, respectively; the component of the deflection vector due to horizontal loading is 0.357 and 0.358 inches for vehicles A and B, respectively.

Responses and linking variable values at the family and vehicle levels are compared in Table 2. The agreement is satisfactory. Note that the front suspension stiffness is treated as a linking variable during the coordination process between these two levels.

The matching of responses between vehicle and system levels is illustrated in Table 3. Once again, deviations are negligible. Note that during the coordination of these two levels the front suspension stiffness is treated as a response. Tables 2 and 3 show that the target cascading process forces a consistent design.

Table 4 presents the results obtained for the product platform, i.e., the three components of the body that are common to the two vehicles. The agreement between the values obtained at the system level and the values obtained at the component level confirms the ability of the target cascading formulation to account for shared components.

Finally, Table 5 presents the optimal values for local design variables for the vehicle, system, and component levels.

Although design values are obtained for all optimization problems formulated within the target cascading formulation, it should be emphasized that the main outcome of this process are the design specifications for the elements of the variants at the vehicle, system, and component levels; namely, vehicle masses, body-in-white masses, suspension stiffnesses, and cross-section related properties (areas and moments of inertia) for the platform components of the body. These design specifications correspond to the optimal values of the responses, as presented in the far-right columns of Tables 2–4. For example, the design specification for the mass of vehicle B is 2163 kg, the de-
sign specification for the body-in-white mass of vehicle A is 239 kg, and the design specification for the footprint area of the rocker is 8.02 in².

6 Conclusions

The target cascading methodology for optimal product development has been extended to the design of product families. The single-product formulation has been modified to accommodate the presence of shared elements and locally introduced targets. Given a platform, hierarchical partitions of the individual product design problems, and the necessary analysis models, family and product targets are cascaded down to systems, subsystems, and components. In this manner design specifications are determined for all elements, including the product platform, and a consistent design is obtained. Moreover, trade-offs between commonality and target achievement can be identified. The information flow within the coordination strategy is based on the hierarchical multilevel structure underlying the family design problem. The technique was successfully applied to an automotive example of two vehicles that share the front suspension and a number of body components. Application to more complicated examples is straightforward albeit demanding with respect to appropriate element models.

Acknowledgements This research was partially supported by the Automotive Research Center (ARC), a US Army Center of Excellence in Modeling and Simulation of Ground Vehicles at the University of Michigan, and by a URP grant from Ford Motor Company. The authors would also like to thank Noboru Kikuchi for providing the finite element body model.

References


Kim, H.M.; Michelena, N.; Papalambros, P.Y.; Jiang, T. 2000: Target cascading in optimal system design. 26-th Design Automation Conf. (held in Baltimore, MD), Paper No. 14265


Michelena, N.F.; Park, H.A.; Papalambros, P.Y. 2002: Convergence properties of analytical target cascading. Proc. 9th AIAA/ISSMO Symp. on Multidisciplinary Analysis and Optimization (held in Atlanta, GA)


A sensitivity-based commonality strategy for family products of mild variation, with application to automotive body structures

R. Fellini, M. Kokkolaras, N. Michelena, P. Papalambros, A. Perez-Duarte, K. Saitou, and P. Fenyes

Abstract Identification of the product platform is a key step in designing a family of products. This article presents a methodology for selecting the product platform by using information obtained from the individual optimization of the product variants. Under the assumption that the product variety requires only mild design changes, a performance deviation vector is derived by taking into consideration individual optimal designs and sensitivities of functional requirements. Commonality decisions are based on values of the performance deviation vector, and the product family is designed optimally with respect to the chosen platform. The proposed methodology is applied to the design of a family of automotive body structures. Variants are defined by changing the functional requirements they need to satisfy and/or the geometry of the associated finite element models.

Key words product platform, product family, optimal design, automotive body

1 Introduction

Sharing components within a family of products can be an effective method for corporations to increase cost savings (Meyer and Lehnerd 1997; Ericsson and Erixon 1999). A drawback to commonality is that a performance deviation can be incurred with regard to optimized individual product design. The challenge is to choose which components to share (i.e., define the product platform), and design the product family with minimal deviation from individual optimal designs.

Simpson et al. (1999) proposed a method for product platform synthesis and exploration based on a market segmentation grid and leveraging and scaling concepts (Meyer and Lehnerd 1997). They solved the family design problem by using goal programming. This methodology has been built upon in a number of subsequent publications (Messac et al. 2000; Conner et al. 1999; Nayak et al. 2000), where the last method uses robust design principles to aid in selecting the product platform.

Gonzalez-Zugasti et al. (1998) presented a method that uses cost gain models as the driving force for designing the product platform while satisfying performance and budget constraints: a priori specified platforms are optimized first; family variants are designed second. Subsequently Gonzalez-Zugasti and Otto (2000) formulated a design optimization problem for modular product architecture that can be solved to determine simultaneously module designs and their combination for the variant instantiations. Fujita et al. (2001) proposed a method for simultaneous optimization of module attributes and combinations. The modular architecture of the product family is fixed in both of the latter papers.

Siddique et al. (1998) examined the applicability of product variety concepts to automotive design. In particular, they investigated whether product variety design concepts such as standardization, delayed differentiation, modularity, module interfaces, robustness, and mutability can be utilized. They limited their consideration for platform to the underbody structure of a vehicle and came to the conclusion that some of these concepts cannot be applied, mainly because of the integral nature of the product architecture. However, they did mention the possibility of partitioning the underbody platform into major manufacturable modules that can be assembled.
Nelson et al. (1999) formulated platform design as a multicriteria optimal design problem. Given a fixed platform, a set of optimal Pareto points is generated based on the importance of the conflicting variant objectives. The designer can identify trade-offs, evaluate multiple platforms, and then make related decisions. Fellini et al. (2000) applied this concept to the design of an automotive product family based on a powertrain platform and examined the hierarchical structure of the platform design problem. Kokkolaras et al. (2002) extended the target cascading formulation to the design of product families with pre-specified platforms. Both common and individual components, subsystems, and/or systems of the family products were designed optimally with respect to family and variant targets.

In the present work a methodology is proposed for making commonality decisions based on individual optima and sensitivity analysis of functional requirements. Emphasis is put on families of vehicle body structures using modeling approaches proposed by Fenyes (2000). The method assumes only “mild variants”, so that design changes can be guided by sensitivity information reasonably well.

The article is organized as follows: Platform-based design of body structures is introduced in the next section. The mathematical derivation underlying the proposed approach is presented. The methodology based on this derivation is formulated and demonstrated by means of an automobile body structure case study. Results are discussed and conclusions are drawn.

2 Platform-based design of body structures

A component is defined as a manufactured object that is the smallest (indivisible) element of an assembly, and is described by a set of design variables. A product is an artifact made up of components. A product platform is the set of all components, manufacturing processes, and/or assembly steps that are common to a set of products. A product family is the set of products that are built upon a product platform. A family product is also referred to as a product variant. Two types of sharing are possible when selecting a product platform that is not based on manufacturing processes or assembly steps. In component sharing, one or more components are common across a family of products as shown in Fig. 1. In addition, it is possible to share “scaled” versions of components. Mathematically this can be described as design variable sharing, where the components themselves are derived from a platform. The example in Fig. 2 shows the cross-section of two structural beam elements. While the height and width of both parts are the same, the thickness is different. The product containing the “thicker” component variant has higher rigidity requirements. By not sharing the thickness, the other product with lesser rigidity requirements does not have to take on an unnecessarily large deviation from the optimal weight. Possible manufacturing advantages are illustrated by this example. By keeping width and height invariant, the same stamping equipment may be used with different gauge steel. In general, manufacturing considerations should be taken into account in the design of platforms. We do not address this aspect explicitly but we attempt to recognize the associated design impact.

2.1 Problem formulation

The following definitions are necessary to formulate the variant and family design problems:

- \( \mathcal{P} = \{p_1, p_2, \ldots \} \): set of \( m \) products
- \( \mathbf{x}^p \): column vector of design variables for the product \( p \in \mathcal{P} \)
- \( \mathcal{S} = \{s_1, s_2, \ldots \} \): set of all permissible platforms, where each \( s \in \mathcal{S} \) is a set of indices describing a platform
- \( s^* \): set of indices describing the “optimal” platform
- \( \mathbf{x}^{p,*} \): null-platform optimal design of product \( p \), corresponding to the individual optimal design solutions of Problem (1) below
- \( \mathbf{x}^{p,**} \): family-optimal design of product \( p \), solution of Problem (2) below; because of sharing variables whose indices are in \( s \), for \( p, q \in \mathcal{P} \) and \( i \in s \), we have \( x_i^{p,**} = x_i^{q,**} = x_i^* \)
- \( \mathcal{G}^p \): set of indices of the active constraints at the null-platform optimum of product \( p \)

For convenience, equality constraints are assumed to have been eliminated implicitly or explicitly. The individual optimal design problem for product variant \( p \) can be formulated as

\[
\begin{align*}
\min_{\mathbf{x}^p} & \quad f^p(\mathbf{x}^p) \\
\text{subject to} & \quad g^p(\mathbf{x}^p) \leq 0.
\end{align*}
\]
The family design problem is then formulated as
\[
\min_{x = [x^p_1, x^p_2, \ldots]} \{f^p(x^p)\} \quad \forall p, q \in \mathcal{P}, \ i \in \mathcal{S}, \ p < q
\]  
subject to \(g^p(x^p) \leq 0,\)
\[x^q_i = x^p_i.\]

The platform selection methodology can be summarized as follows: Quantify performance deviations by considering individual optimal designs and sensitivities of functional requirements; decide which components can be shared (i.e., determine the platform) with minimal performance deviation; optimally design the product family around the chosen platform.

3 Commonality decisions

The proposed approach is based on the use of optimality and sensitivity information obtained from individual product optimization to assess the potential deviation from the optimal design incurred by sharing variables. When the products in the family contain a large number of components that are candidates for sharing, platform selection entails the solution of a large combinatorial problem. In the approach proposed, this problem is reduced to a simpler one under the assumptions listed below. The derivation presented in the following section is based on a first order Taylor series approximation. Therefore, in order for the approximation to remain reasonably accurate, the general condition is that the individual optimal designs lie not “too far away” from each other so that the linear approximation is valid in the region between them. The derivation will be presented for a family of two products \(A\) and \(B\) for simplicity; it can be generalized readily for more products.

We make the following assumptions:

1. Self-sharing (i.e., component sharing within the same variant) is not possible.
2. Components are either shared by all family members or not at all.
3. Null-platform optimal designs lie “close enough” to each other.
4. The platform design (denoted here by superscript *) lies in the convex hull of the individual solutions (denoted by superscripts \(A,^\circ\) and \(B,^\circ\). That is, \(\exists \lambda_i \in [0, 1]\) such that \(\forall i \in s, \ x^*_i = \lambda_i x^{A,^\circ}_i + (1 - \lambda_i) x^{B,^\circ}_i.\)
5. Constraint inactivity remains unchanged between individual and family design problems.

We refer to the design solutions that satisfy these assumptions as “mild variants”.

Sharing may cause deviations from individually optimized products designs, which is measured by the responses representing the functional requirements. In the context of the approach introduced in this article, the commonality decision consists of deciding which variables to share. The design variables are arranged in order of increased performance deviation value, and the number \(n\) of variables to share is determined by a limit on acceptable design deviations. The optimal platform is determined by minimizing the relative deviation, \(\Delta^p\), of the designs based on any platform with \(n\) shared variables with respect to the null-platform optimal designs – while remaining in the feasible space for the variants. Formally, this is stated as
\[
\min_{s \in \mathcal{S}} \Delta
\]
subject to \(|s| = n,\)
where \(\Delta = \Delta^A + \Delta^B\) and
\[\Delta^p = |f^p(x^{P,^*}) - f^p(x^{P,^\circ})| + \sum_{j \in \mathcal{G}^p} \max \left( g^p_j(x^{P,^*}), 0 \right)\]
for \(p \in \mathcal{P} = \{A, B\}\). By the definition of \(\mathcal{G}^p, \ g^p_j\) is active at the null-platform optimum \(x^{P,^\circ}\); therefore \(g^p_j(x^{P,^*}) = 0\). Normalization is used to enable the meaningful summation of responses of different nature.

A first order Taylor series approximation of the variation in each response \(f^p, g^p_j\), for \(j \in \mathcal{G}^p\) is introduced in agreement with the assumptions described in Sect. 3:
\[
f^p(x^{P,^*}) - f^p(x^{P,^\circ}) \approx \sum_{i \in \mathcal{S}} \left( \nabla f^p_i \right)^T (x^{P,^*} - x^{P,^\circ})
\]
\[g^p_j(x^{P,^*}) \approx \left( \nabla g^p_j \right)^T (x^{P,^*} - x^{P,^\circ}),\]
where \(\nabla f^p_i, \nabla g^p_j\) is the gradient of \(f^p, g^p\) evaluated for the null-platform optimal design of product \(p\).

Furthermore, under Assumption 4, the relation between the shared variables, \(i \in s,\) and the null platform can be re-written as
\[
\left( x^{i,^\circ}_i - x^{i,^\circ}_i \right) = (1 - \lambda_i) \left( x^{B,^\circ}_i - x^{A,^\circ}_i \right).
\]
Consequently, the deviation of the objective \(f\) in one variant \(A\) due to sharing of the variables \(x_i, i \in s,\) is approximated by
\[
f^A(x^*) - f^A(x^{A,^\circ}) \approx \sum_{i \in s} \nabla f^A_i \left( x^{i,^\circ}_i - x^{B,^\circ}_i \right)
\]
\[\approx \sum_{i \in s} (1 - \lambda_i) \nabla_i f^A_i \left( x^{B,^\circ}_i - x^{A,^\circ}_i \right).\]
Letting \(\delta_i = |x^{B,^\circ}_i - x^{A,^\circ}_i|,\) an upper bound and an approximation on the total variation in \(\Delta^A\) is
\[
\Delta^A \leq \sum_{i \in s} (1 - \lambda_i) \left( \left| \nabla_i f^A \right| \delta_i + \max_{j \in \mathcal{G}^A} \left( \nabla_i g^A_j \delta_i, 0 \right) \right).
\]

A similar upper bound can be obtained for \(\Delta^B\). We define the performance deviation vector \(\Pi\), whose entries correspond to performance deviations due to sharing:
\[ H_i = (1 - \lambda_i) \left( \left| \nabla_i f^{A_i} \right| \delta_i + \sum_{j \in G^A} \max \left( \nabla_i g_{j}^{A_i} \delta_i, 0 \right) \right) + \lambda_i \left( \left| \nabla_i f^{B_i} \right| \delta_i + \sum_{j \in G^B} \max \left( \nabla_i g_{j}^{B_i} \delta_i, 0 \right) \right). \] (5)

The performance deviation vector \( \Pi \) provides an upper bound on the actual performance deviation \( \Delta \):

\[ \Delta \leq \| \Pi \|_1. \] (6)

The approach adopted in this article for approximating a solution to the original problem described in Problem (3) is to minimize the upper bound on \( \Delta \) as given in (6). In this regard, the choice of the parameters \( \lambda \) has to be discussed. These parameters are determined theoretically by the position of the family solution for a given platform \( s \) relative to the position of the null-platform solutions for the two variants (Assumption 4). In the framework described here, the exact values of \( \lambda \) are not known a priori since the solution to the family problem is not available. In this regard, we simply assume that \( \lambda_i = 1/2, \forall i \). Hence, there is no bias towards one variant or the other with regard to the family design variable values. The choice of this value does not affect the commonality decisions. However, the validity of the “convex-hull” assumption (Assumption 4) needs to be checked after solving the family design problem to ensure that commonality considerations are reasonable for the related component or design variable.

The design variables are arranged in order of increasing \( H_i \). The variables to be shared are the first \( n \) variables below some threshold. This minimizes the upper bound on \( \Delta \) according to (6) as an acceptable alternative to solving Problem (3).

4 Proposed methodology

The proposed general methodology for selecting the product platform and designing the product family is as follows:

1. Generate product variants based on:
   a. design requirements,
   b. geometry (no topological changes) of the model(s),
   c. or both.
2. Develop appropriate analysis models and identify inputs and outputs.
3. Formulate and solve the optimal design problem (1) for each variant, i.e., find the null-platform optimal designs.
4. Use optimal design variables and sensitivity information to compute the performance deviation vector \( \Pi \) by means of (5).
5. Sort the variables in order of increasing \( H_i \).
6. Using the performance deviation vector \( \Pi \), decide which components to share based on how much performance deviation is acceptable.
7. Formulate and solve the family design problem (2) for the chosen platform.
8. Compare family-optimal designs to individual variant optimal designs and evaluate the actual performance deviation, \( \Delta \); iterate if necessary.

5 Application study

A family of automotive body structures is considered. A variant is defined as a structure associated with specific dimensional properties (lengths) and functional requirements.

5.1 Model description

The structures are modeled using finite elements in MSC Nastran according to modeling approaches described in Fenyes (2000). Modal and static load cases (torsion on the front and rear shock towers, and bending) are considered, as shown in Fig. 3. It is assumed that these load cases give access to the properties that the designer wishes to tailor, and therefore are valid as a basis of the design.

The finite element analysis outputs mass \( m \) and natural frequencies \( \omega \), in addition to displacements and stress responses for static load cases of front torsion, rear torsion, and bending (denoted \( d_{ft}, d_{rt}, d_b \), respectively) along with corresponding sensitivity information for all the design variables. These are the cross-sectional dimensions of the beams (width \( b \), height \( h \), and thickness \( t \)) and thicknesses \( t \) of the shells. There are 66 design variables.

We used the SCPIP algorithm (Zillober 2001) for solving the optimization problems, which is an implementation of the method of moving asymptotes (MMA), tailored to solve large-scale structural optimization problems efficiently.

As mentioned, variants are generated either by implementing dimensional changes or by imposing different design requirements. We examine these two cases next.

Fig. 3 Automotive body structure model
5.2 Dimensional variants

The individual optimal design problem is formulated as

\[
\begin{align*}
\min_{b, h, t} & \quad m \\
\text{subject to} & \quad \omega_1 \geq 21 \text{ Hz}, \\
& \quad \omega_2 \geq 24 \text{ Hz}, \\
& \quad d_{ft} \leq 2.9 \text{ mm}, \\
& \quad d_{rt} \leq 2.9 \text{ mm}, \\
& \quad d_b \leq 0.2 \text{ mm}, \\
& \quad \sigma_{max} \leq 25 \text{ MPa}.
\end{align*}
\]

Here we consider a family of two variants based on dimensional changes (cf. Figure 4) having the same objective functions and constraints.

As shown in Fig. 4, a second variant is generated by stretching the wheelbase and trunk of the baseline vehicle. The engine compartment is shortened, and therefore a smaller engine (and lumped mass representing the engine) is assumed. The models will be correspondingly referred to as the short and long wheelbase body models. The null-platform optima are summarized in Table 1.

The performance deviation vector \( \mathbf{\Pi} \) is computed according to (5), and the platform is determined as described in Sect. 4. Using a threshold value of 0.01, 59 variables are selected for sharing and the family problem (2) is solved. The family optima are summarized in Table 2.

Fig. 4 Automotive body structure dimensional variants

### Table 1 Null-platform optima (dimensional variants)

<table>
<thead>
<tr>
<th></th>
<th>short</th>
<th>long</th>
</tr>
</thead>
<tbody>
<tr>
<td>mass (kg)</td>
<td>715.13</td>
<td>703.36</td>
</tr>
<tr>
<td>( \omega_1 ) (Hz)</td>
<td>21.00</td>
<td>22.06</td>
</tr>
<tr>
<td>( \omega_2 ) (Hz)</td>
<td>24.82</td>
<td>27.00</td>
</tr>
<tr>
<td>( d_{ft} ) (mm)</td>
<td>2.158</td>
<td>2.170</td>
</tr>
<tr>
<td>( d_{rt} ) (mm)</td>
<td>1.905</td>
<td>1.909</td>
</tr>
<tr>
<td>( d_b ) (mm)</td>
<td>0.200</td>
<td>0.200</td>
</tr>
</tbody>
</table>

Fig. 5 Dimensional variants: Non-shared components

Overall, the family based on the 59-variable platform is close to the null platform: The optimal masses of both the short and long wheelbase variants are almost identical to the corresponding null-platform designs. The long wheelbase variant using the total platform is still close to the corresponding null-platform variant, compared with a 10.5-kg difference in mass in the short wheelbase variants. The components that are not completely shared among the variants are shown in Fig. 5.

For each of these components the material thickness is the variable that varies. Overall a large number of variables may be shared with negligible performance deviation, considering that the performance deviation of each variant is less than 1.5% compared to the corresponding null-platform variants. This can be traced to the fact that the variants do not have competing design objective functions, and that their geometric configurations are very similar (cf. Fig. 4). The combination of these two factors results in relatively close individual optima and family optima. The next study was subsequently devised to test the proposed methodology on a problem that does not present these features, and it is discussed more thoroughly.

5.3 Performance variants

We now look at variants based on the same geometric model (the short wheelbase model) having different design objectives and constraints. Two variants with competing objectives are designed, denoted “stiff” and “light weight”, respectively. In the former the designer aims at maximizing the stiffness of the structure to improve ride...
quality, while in the latter the goal is to minimize weight to improve fuel economy.

The flexibility $\varphi$ is defined as a weighted sum of the displacements $d_{ft}$, $d_{rt}$, $d_b$ in the three load cases considered, namely, front torsion, rear torsion, and bending, respectively. The weights approximate the ratios of the expected displacements (cf. null-platform optima in Table 1) in each load case; hence flexibility is computed as follows:

$$\varphi = d_{ft} + d_{rt} + 10d_b.$$  \hfill (8)

The optimal design problem statement for the lightweight variant is

$$\min_{b, h, t} m$$

subject to

$$\omega_1 \geq 15 \text{ Hz},$$

$$\omega_2 \geq 17 \text{ Hz},$$

$$d_{ft} \leq 2.9 \text{ mm},$$

$$d_{rt} \leq 2.9 \text{ mm},$$

$$d_b \leq 0.2 \text{ mm},$$

$$\sigma_{\text{max}} \leq 25 \text{ MPa},$$

while for the stiff variant the statement is as follows:

$$\min_{b, h, t} \varphi$$

subject to

$$\omega_1 \geq 21 \text{ Hz},$$

$$\omega_2 \geq 24 \text{ Hz},$$

$$m \leq 822 \text{ kg},$$

$$\sigma_{\text{max}} \leq 25 \text{ MPa}.$$  \hfill (10)

Each variant is optimized individually to obtain a null-platform design. The optimal objective function values for the light weight and stiff variants are $691.87\text{ kg}$ and $4.4049\text{ mm}$, respectively. The null-platform optimal designs and sensitivities are used to compute the performance deviation vector $\Pi$.

The design variables are arranged in order of increasing performance deviation. Figure 6 depicts a plot of the sorted performance deviation vector.

The graph shows that the performance deviation remains low for the first 50 variables approximately, and then begins to increase sharply. We chose a 54-variable platform based on the fact that the curve exhibits a sharp increase after 54 variables (cf. Fig. 6). The components that are not shared among the variants are shown in Fig. 7. As in the previous example the material thickness is most often the dimension that varies. One exception is that the rocker panels differ in width, height, and thickness. We solved the family problem for the 54-variable platform and the total platform by minimizing the distance to the null-platform optimum. The results are shown in Table 3, where the variant objectives have been underlined.

Figure 8 shows the Pareto sets for the 54-variable and total platforms. The 54-variable platform shares all but

<table>
<thead>
<tr>
<th>variant</th>
<th>null platform</th>
<th>54-variable platform</th>
<th>total platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>mass (kg)</td>
<td>stiff</td>
<td>l. weight</td>
<td>stiff</td>
</tr>
<tr>
<td>$d_{ft}$ (mm)</td>
<td>1.581</td>
<td>2.429</td>
<td>1.595</td>
</tr>
<tr>
<td>$d_{rt}$ (mm)</td>
<td>1.396</td>
<td>2.148</td>
<td>1.409</td>
</tr>
<tr>
<td>$d_b$ (mm)</td>
<td>0.1427</td>
<td>0.2922</td>
<td>0.1429</td>
</tr>
<tr>
<td>flexibility (mm)</td>
<td>4.405</td>
<td>7.499</td>
<td>4.433</td>
</tr>
</tbody>
</table>

Fig. 6 Sorted performance deviation vector $\Pi$ (performance variants)

Fig. 7 Performance variants: Non-shared components

Table 3 Optima for null, 54-variable, and total platforms (performance variants)
1.005
1.01
1.015
1.02
1.025
1.03
weight
flexibility
pareto set
family solution
(A)
(B)
Fig. 8 Pareto sets for the 54-variable and total platforms, with normalized objectives (performance variants). The plot on the top contains the null platform point (A) and the Pareto set for the 54-variable platform (B). The plot on the bottom includes also the Pareto set for the total platform (C).

1.05
1.1
1.15
1.2
weight
flexibility
54 var platform
total platform
(A)
(B)
Fig. 9 Normalized actual performance deviation vs. upper deviation bound as computed by means of the performance deviation vector for different platforms, i.e., number of shared variables (performance variants)

18% of the variables, with a deviation of 0.6% for the stiff variant and 1.16% for the light weight variant. In contrast, the total platform has a 1.4% deviation for the stiff variant and an 18.8% deviation for the light weight variant.

5.4 Discussion

The validity of some of the assumptions described in Sect. 3 can only be checked a posteriori, i.e., after solving the individual optimization problems and the family design problem. We checked the assumptions for both case studies.

Assumptions 1 and 2 are automatically satisfied by the implementation of the methodology for this case study.

The distance between the null-platform designs was relatively small for both cases (Assumption 3).

Assumption 4 is satisfied; by inspecting the results obtained from solving the family design problems, \( \exists \lambda_i \) such that \( x^*_i = \lambda_i \cdot x^{stiff}\circ + (1 - \lambda_i) \cdot x^{weight}\circ \) \( \forall i \) in all cases. This assumption holds for this problem but may not hold for other problems. It is rather strong and further research is needed in order to relax it.

Assumption 5 is designed to avoid the case in which a constraint that is inactive in the individual solution becomes active in the family solution, a case which is not taken into account in the current derivation of Sect. 3.2. Here this assumption is validated with no additional constraints becoming active. In fact, one of the active constraints became inactive in the performance variants case. This is expected, since adding equality constraints in the family design problem (the commonality constraints), it is likely that some constraints may become inactive.

If the assumptions required by the approximation are satisfied, the performance deviation vector \( \Pi \) provides an upper bound on the potential performance deviation. To illustrate the fact that the performance decision vector is a conservative metric, the family problem was solved for several platforms, with a number of variables ranging from 30 to 65. In Fig. 9, the actual objective deviations for each variant are compared with the upper bound given by the \( l_1 \) norm of \( \Pi \) for different platforms (number of shared variables).

6 Conclusions

The proposed methodology uses first-order information, obtained from individual design optimizations to compute a metric for performance deviations attributed to component sharing. Under the assumption of mild variations among family products, this analysis can be used
to identify which components should or should not be part of the product platform, and becomes essential when the number of sharing combinations becomes too large to search exhaustively. The methodology has been applied to the design of a family of automotive body structures. Results obtained for the two case studies demonstrate the usefulness of the proposed approach. Future work focuses on integrating this methodology with the commonality decision formulation presented by Fellini et al. (2002).

Acknowledgements This research was partially supported by the Automotive Research Center, a US Army Center of Excellence in Modeling and Simulation of Ground Vehicles, by a US Army Dual-Use Science and Technology Project, and by the General Motors Collaborative Research Lab at the University of Michigan. This support is gratefully acknowledged. We would like to thank Christian Zillober for providing the SCPIP algorithm, and Joe Donndelinger, Devadatta Kullarni, and Robin Stevenson for their many helpful ideas and suggestions. The opinions expressed in this paper are solely those of the authors and do not reflect acknowledgement or acceptance by the sponsors.

References


Platform Selection Under Performance Bounds in Optimal Design of Product Families

Designing a family of product variants that share some components usually requires a compromise in performance relative to the individually optimized variants due to the commonality constraints. Choosing components for sharing may depend on what performance losses can be tolerated. In this article an optimal design problem is formulated to choose product components to be shared without exceeding user-specified bounds on performance. This enables the designer to control tradeoffs and obtain optimal product family designs for maximizing commonality at different levels of acceptable performance.

A family of automotive body side frames is used to demonstrate the approach.

[DOI: 10.1115/1.1899176]

1 Introduction

A product platform is defined as the set of components and manufacturing and assembly processes that are common in a family of products. Commonality reduces costs but may require some sacrifice in individually optimized product performance [1]. The design challenge is how to optimize the family of products with maximum commonality while satisfying individual constraints and controlling performance losses.

The literature includes a significant amount of work focusing on developing methodologies for the design of product families. These methods are applied when the product platform has been determined by the designer a priori. Gonzalez-Zugasti et al. [2] used cost gain models for designing the product platform while satisfying performance and budget constraints. Several a priori specified platforms are optimized first, while the family variants are designed at the second stage. Simpson et al. [3] proposed a product platform concept exploration method for configuring and exploring product platforms that are scaled to derive product families. A compromise decision support problem was formulated to optimize the design, while in Messac et al. [4] a physical programming approach was adopted. Conner et al. [5] utilized Simpson’s product variety tradeoff evaluation method and the combination of the above two concepts to evaluate platform portfolios based on commonality and performance losses indices. Nelson et al. [1] formulated and solved a multiobjective optimal design problem by means of Pareto set theory. Each platform is shown to correspond to a different Pareto surface. The designer can identify trade-offs, evaluate multiple platforms, and make related decisions. Fellini et al. [6] applied this concept to the design of automotive powertrains and examined the hierarchical structure of the platform design problem. Kokkolaras et al. [7] extended the analytical target cascading formulation to hierarchical models of product families with pre-specified platforms.

At a level above the product family design problem one must also decide what can be shared among a set of products. This is a key ingredient in the complete optimal design of product families. The following methodologies look at not only designing the family of products but aiding the engineer in selecting which components to make common. There have been three groups of approaches taken to address this problem.

The first approach has been simple exhaustive enumeration of the sharing possibilities. Conner Seepearsad et al. [8] enumerate a subset of all combinatorial possibilities for assigning fixed platforms to a family of products. These platform alternatives are designed and evaluated by comparing performance losses and cost benefits. This type of method could be applied for relatively small problems; however, these type of methods are severely limited as the problem size moves beyond what can be searched exhaustively.

The second approach to the commonality decision problem has been to develop optimal design formulations which are used to select both the platform and design the product family. This group of methods have been applied both in an all-at-once approach or in a series of intermediate steps. Gonzalez-Zugasti and Otto [9] formulated a modular product platform optimization problem to determine simultaneously module designs and their combination in the variant instantiations. Platform and individual goals form a multiobjective function, and additional compatibility and sharing constraints are included. Fujita and Yoshida [10] also proposed a method for simultaneous optimization of module attributes and combinations, driven by cost-related functions. In both of these papers a modular architecture of the product family is fixed, and a genetic algorithm (GA) is used to solve small-sized combinatorial problems. Other researchers have also adopted GAs for solving the commonality selection and family design problems [11,12].

The main drawbacks of these methods are: (1) the use of heuristic algorithms such as GAs, which require extensive fine tuning, often are extremely expensive to run, and have no formal proof of convergence; (2) the solution of any pure combinatorial problem is quickly bound by the problem size where combinations are increasing exponentially. For example, in the case studies presented in the aforementioned work, a relatively small number of possible sharing combinations are explored, often no more than on the order of $10^3$. One benefit of using a combinatorial algorithm, as shown by Gonzalez-Zugasti and Otto, is that distinct module types can be chosen through a catalog selection. This type of problem is however once again limited due to the combinatorial nature. Additional characteristics of these methodologies include: the first two papers allow for “subplatforms” to be chosen, component sharing that is not necessarily across the entire family of products. The latter papers explore component sharing only across the family of products.

The third group of methodologies attempt to approach large problem sizes by developing metrics that sort the components by shareability. Nayak et al. [13] employed robust design concepts to formulate a variation-based platform design methodology that consists of two steps: identifying the platform by solving a compromise decision support problem and designing the family.
around this platform. Fellini et al. [14] employ sensitivity information at the individually optimized products to produce a sorted vector ranking the shareability of the components, and then optimize the product family on a selected platform. The benefits of these methods are that they can aid in reducing the size of the commonality decision problems, however the actual selection of a platform based on the rankings is ad hoc. The methodologies by D’Souza and Simpson attempt to use design of experiments (DOE) to reduce the problem size, but likely as the problems become large these tools lose much of their accuracy [11,12].

The focus of this paper is to address the problem addressed by the second group of methodologies. In the present work the combinatorial problem is relaxed into a continuous form and reformulated to maximize commonality among the family members (allowing for subplatforms) while satisfying individual design constraints and observing a designer-specified bound on the reduction from the individually optimized performance. The designer can identify a set of components that can be shared, and obtain optimal designs for a number of scenarios based on the willingness to sacrifice a certain amount of individual performance. By relaxing the combinatorial decision into a continuous problem formulation, gradient-based methods such as sequential quadratic programming can be employed. The contribution of the proposed methodology is an approach that not only allows for an enormous increase in the size of the problem that can be solved but also benefits from the use of deterministic methods that have rigorous convergence proofs and do not require tuning parameters.

The paper is organized as follows. After some definitions the product family design and commonality decision problems are posed mathematically. The methodology for solving them is then presented and subsequently demonstrated on a family of automotive body side frames. Results are discussed and conclusions are drawn.

2 Definitions

In this section we review the vocabulary and basic model for a product platform.

A component is defined as a manufactured object that is the smallest (indivisible) element of an assembly and is represented by a set of design variables. A product is an artifact that is made up of components. The product architecture is the configuration (or topology) of components within the product. A module is a component or subassembly that can be interconnected within a product architecture to produce a variety of similar products. A model is a mathematical representation of a product that accepts a vector of design variables and returns a vector of responses.

The mathematical notation begins with the set \( \mathcal{P} = \{ p_1, p_2, \ldots \} \) used to distinguish each product \( p \in \mathcal{P} \). Likewise, the set \( \mathcal{C} = \{ c_1, c_2, \ldots \} \) is defined to represent the components that form a particular product \( p \). Thereby the union of all components, \( \mathcal{C} = \bigcup \mathcal{C}_p \), constructs the product. Figure 1 illustrates the above notation. A product \( p \) is associated with a vector of design variables \( x_p \), a vector of responses \( \mathbf{R}_p \), and design inequality and equality constraints \( \mathbf{g}_p \) and \( \mathbf{h}_p \), respectively. The subset of design variables describing component \( c \) is \( x_c \subseteq x_p \).

A product platform is the set of all components, manufacturing processes, and/or assembly steps that are common in a set of products. The following notation is used to describe a product platform: The set \( \mathcal{S}^p \) consists of the index pairs of elements that are shared between two products \( p \) and \( q \). The set \( \mathcal{S}^p = \{(p,q) \mid p,q \in \mathcal{P}, p \neq q \} \) describes element sharing throughout the family.

Two types of sharing can be used when selecting a product platform that is not based on manufacturing processes or assembly steps. In component sharing, one or more components are common across a family of products as shown in Fig. 2. In addition, it is possible to share “scaled” versions of components. Mathematically this can be described as variable sharing, where components are based on a platform (of variables) themselves. The example in Fig. 3 shows the cross section of two structural beam elements. While the height and width of both parts are the same, the thickness is different. The manufacturing advantage can be illustrated by this example. By keeping width and height invariant, the same stamping equipment can be used with different gauge steel. In general, manufacturing (and therefore cost) considerations should be taken into account in the design of platforms. We do not address this aspect explicitly, but we attempt to recognize the associated design impact. The methodology presented holds for both component and variable sharing. When defining a platform that includes both component and variable sharing the subscripts \( \mathcal{C} \) and \( \mathcal{P} \) are used to denote the individual platform element types, respectively. The information is then held in the set \( \mathcal{S}^c = \mathcal{S}^c \cup \mathcal{S}^p \).

Finally, a product family is the set of product variants that share a product platform. A family product derived from a platform is also referred to as a product variant.

The formulation to be used in the next section is the multicriteria optimization statement proposed by Nelson et al. [1],

\[
\begin{align*}
\max_{x \in \{x^p, x^c, \ldots\}} & \quad \{p(x)\} \quad \forall \ p, q \in \mathcal{P}, (i,j) \in \mathcal{S}^p, p < q \\
\text{subject to} & \quad \mathbf{g}(x) \leq 0 \\
& \quad \mathbf{h}(x) = 0 \\
& \quad x_i = x_j.
\end{align*}
\]

The objective function for the family is a weighted aggregate of the individual product objectives. The constraint set includes all individual design constraints. Commonality constraints are represented as equality constraints \( x_i = x_j \) for each set of shared vari-
ables specified in $S$. An optimization problem is solved for each product separately to determine the null-platform design. The optimal null-platform objective function and design values are denoted by $f^{\text{null}}$ and $x^{\text{null}}$, respectively. Likewise, $f^{p}$ and $x^{p}$ are used to represent the optimal objective function and design values for family designed products, respectively. The null-platform objectives from all family products define the null-platform point, shown in Fig. 4 as the point $\times(f^{\text{null}}, f^{\text{null}})$. Solving the multicriteria optimization problem Eq. (1) yields a Pareto set. Two possible platforms (with different equality constraints) are shown in Fig. 4. The bounds of the Pareto set determine the utopia point, the best trade-off design that might be achieved with the platform. In practice, the designer first computes the design penalty from the null-platform point, the optimal null-platform objective function and design values are determined. The objective in Eq. (1) is selected according to other criteria (cost, customer preferences, etc.) not included in the above model.

Fig. 4 Null-platform point and Pareto sets for different platforms

Fig. 5 The approximation of the function $D$. \[ \max_{q,a=(x^p,x^p...)} \sum_{(i,j)\in\mathcal{P}} \eta^{pq}_{ij} \quad \forall \ p,q \in \mathcal{P}, (i,j) \in \mathcal{S}^{pq}, \ p < q \] \[ \text{subject to} \quad g^p(x^p) \leq 0 \] \[ h^p(x^p) = 0 \] \[ \eta^{pq}_{ij}(x^i - x^j) = 0 \] \[ \eta^{pq}_{ij} \in \{0, 1\}. \]

Simple monotonicity analysis shows that varying bounds systematically will generate the Pareto set so that the two formulations are treated as equivalent: selecting bounds corresponds to a specific set of objective weights.

Using the function \[ D_a(x^p - x^q) = \begin{cases} 0 & \text{if} \ x^p = x^q \\ 1 & \text{otherwise} \end{cases} \]
the term $\sum_{(i,j)\in\mathcal{P}} D_a(x^i - x^j)$ can be computed based on the values of the design variables:

\[ \sum_{(i,j)\in\mathcal{P}} D_a(x^i - x^j) = \sum_{(i,j)\in\mathcal{P}} \left( \sum_{pq \in \mathcal{S}^{pq}} D_a(x^p - x^q) \right) \]

where $|\mathcal{S}^{pq}|$ is the number of elements in the set $\mathcal{S}^{pq}$, which is constant and can be left out. Note that the equality constraints [Eq. (2d)] are included in Eq. (5). Therefore, maximizing $\sum_{(i,j)\in\mathcal{P}} \eta^{pq}_{ij}$ is equivalent to minimizing $\sum_{(i,j)\in\mathcal{P}} D_a(x^i - x^j)$.

To address the combinatorial nature of this problem the function $D_a$ is approximated by a function $D_a$. The function $D_a$ should satisfy two requirements: its range should be $[0, 1]$ and it should be continuously differentiable. The function we have selected for $D_a$ is defined as:

\[ D_a(x^i - x^j) = 1 - \frac{1 - \left(\frac{x^i - x^j}{\alpha}\right)^2}{1 + \alpha}. \]

This function is constructed as a measure of the distance between designs and approaches the function $D_a$ as $\alpha$ goes to zero. Figure 5 shows $D_a$ for $\alpha=0.05$. Since $D_a$ is continuously differentiable, gradient-based algorithms can be used to solve the approximate commonality problem.

3 The Commonality Decision Problem

The important tradeoff when choosing a platform is commonality versus performance. The product family design problem is now restated to include component commonality in the objective and which variables to share in the decision making, leading to a mixed-discrete programming problem due to the presence of the vector of binary (0-1) sharing decision variables $\eta$:

\[ \max_{q,a=(x^p,x^p...)} \left\{ \{f^p(x^p)\}, \sum_{(i,j)\in\mathcal{P}} \eta^{pq}_{ij} \right\} \quad \forall \ p,q \in \mathcal{P}, (i,j) \in \mathcal{S}^{pq}, \ p < q \]

\[ \text{subject to} \quad g^p(x^p) \leq 0 \]

\[ h^p(x^p) = 0 \]

\[ \eta^{pq}_{ij}(x^i - x^j) = 0 \]

\[ \eta^{pq}_{ij} \in \{0, 1\}. \]

The set $\mathcal{S}^{pq}$ in Eq. (2a) consists of index pairs of variables in products $p$ and $q$ that are candidates for sharing. The sharing decision variables $\eta^{pq}_{ij}$ are set to 1 if variables $x^i$ and $x^j$ will be shared and 0 otherwise, hence Eq. (2d). Equations (2b) and (2c) are the individual design constraints. Finally, the objective in Eq. (2a) now includes one more term, the “commonality” term that sums up all shared variables.

This multicriteria problem is reformulated keeping the term $\sum_{(i,j)\in\mathcal{P}} \eta^{pq}_{ij}$ as the scalar objective function and treating the terms $f^p(x^p)$, as lower bound constraints (where the constant $f^p_i$ represents the lower bound):
As discussed previously, solving the multicriteria problem can be reformulated by considering the objective terms $f^p$ as constraints. To do this, we define performance loss factors $L^p$ that represent the loss in performance of the family products compared to the null-platform optimum $f^p$. The constraints are defined as:

\[
f^0(x^p) \geq (1 - L^p)f^p = f^p_0.
\]

The complete formulation of the (approximate) commonality problem is now as follows:

\[
\max_{x^p \in \{x^p_1, x^p_2, \ldots\}} \sum_{pq} |S^{pq}| - \sum_{(i,j) \in S^{pq}} D_i(x^p_i - x^p_j)
\]

\[
\forall p, q \in \mathcal{P}, (i,j) \in S^{pq}, p < q
\]

subject to

\[
g^p(x^p) \leq 0
\]

\[
h^p(x^p) = 0
\]

\[
f^0(x^p) \geq (1 - L^p)f^p = f^p_0.
\]

(8)

Here the performance bounds are placed as lower bounds by defining them as percentage loss from the individually optimized values. For example, if a 15% loss in performance for a particular attribute is acceptable, the loss factor $L^p$ is simply equal to 0.15.

The solution to Eq. (8) may not be unique. Figure 6 shows the reduced feasible set resulting from the introduction of performance bounds. Furthermore, multiple combinations of the same number of shared components can exist; these must be differentiated by their relative performance after solving the family problem, Eq. (1). Recall that this step of the methodology aims at selecting the feasible platform set, not at finding the design values of $x^p$.

The loss factors $L^p$ are considered input parameters specified by the designer. A postoptimal parametric study may be necessary to determine the acceptable tradeoff between performance and commonality, essentially generating the Pareto set of Eq. (2).

Even the commonality decision formulation in Eq. (8) is multiobjective and, as stated, contains no bias towards sharing one variable over another. However, it is possible to include the designer’s preference for sharing by modifying the objective function.

\[
\max_{x^p \in \{x^p_1, x^p_2, \ldots\}} \sum_{pq} |S^{pq}| - \sum_{(i,j) \in S^{pq}} w_{ij}D_i(x^p_i - x^p_j)
\]

\[
\forall p, q \in \mathcal{P}, (i,j) \in S^{pq}, p < q.
\]

(9)

The scalar weight $w_{ij}$ corresponds to preference on sharing variables $x^p_i$ and $x^p_j$, which can be chosen using manufacturing cost, or other criteria not included in the model here. Including cost may be straightforward by replacing the $w_{ij}$ with the actual cost of the components: If component ‘1’ costs 200 times more to manufacture than component ‘2,’ preference should be placed on sharing the more expensive component. Exploring these multiple layers of multiobjective decision is beyond the scope of this article. Therefore, in the remainder we assume no preference in component sharing, so that all $w_{ij}$ will be equal.

4 Solving the Commonality Decision Problem

Since the solution to the approximate commonality problem will not yield 0–1 values, a determination must still be made about which variables will be selected as common. This is done after solving the commonality decision problem, Eq. (8), and in the following manner: the values of the design variables of the candidate components are compared, and assumed to be shared if their relative difference does not exceed a numerical tolerance. The designer must choose an appropriate value that ensures accuracy of the solution for the particular problem being solved. The shared variables are included as commonality constraints when solving the family design problem, Eq. (1). Therefore, one might compare the tolerance on sharing to the tolerance on satisfying constraints. A high tolerance might often lead to the suggestion of more sharing; however, this decreases the chances of satisfying the performance deviation constraints when the family design problem is solved.

The multicriteria family design problem, Eq. (1), is reformulated to minimize the distance between the null-platform design and the Pareto set corresponding to the selected platform.

\[
\min \{ (f^p(x^p) - f^0(x^p)) \}^2 \forall p, q \in \mathcal{P}, (i,j) \in S^{pq}, p < q
\]

subject to

\[
g^p(x^p) \leq 0
\]

\[
h^p(x^p) = 0
\]

\[
f^0(x^p) \geq (1 - L^p)f^p_0
\]

\[
x^p_i = x^p_j.
\]

(10)

The performance bounds are included in case the Pareto point closest to the null-platform point lies outside the area of allowable performance loss.

The proposed methodology can be stated now with the following steps:

(1) Determine the optimal null-platform design $f^p_0$, $x^p_0$ for each individual product $p \in \mathcal{P}$ by solving the individual optimal design problem Eq. (11).

\[
\max_{x^p} f^p(x^p)
\]

subject to

\[
g^p(x^p) \leq 0
\]

\[
h^p(x^p) = 0
\]

(11)

(2) Identify the components that could be shared between products, i.e., define the candidate platform set $S^{pq}$ for any two products $p$ and $q$ in the set $\mathcal{P}$. The candidate platform set for the whole product family is then $S^{*} = \{S^{pq} | p, q \in \mathcal{P}, p \leq q \}$.

(3) Determine the performance loss factors $L^p$ acceptable for each of the products.

(4) Solve the approximate commonality decision problem Eq. (8).

(5) Based on the results of the commonality decision problem, make a selection of components to be shared, i.e., determine the set $S^{*} = \{S^{pq} | p, q \in \mathcal{P}, p \leq q \}$.

(6) Solve the family design problem Eq. (10).

Figure 7 illustrates the above methodology. The conceptual plot shows a null-platform point $(f_0^{p_1}, f_0^{p_2})$. The null-platform objective function values are multiplied by the performance loss tolerances in the form $(1-L^p)$, and so the region where the associated feasible platforms reside is bounded by the points $(f_0^{p_1}, (1-L^p)f_0^{p_2})$, $(1-\sum f_0^{p_1}, (1-L^p)f_0^{p_2})$, and $(f_0^{p_1}, (1-L^p)f_0^{p_2})$. Solving the commonality decision problem Eq. (8) a feasible platform (i.e., common components) is found. The performance loss
bounds in Eq. (8) will be active, unless they are dominated by design constraints, and the obtained designs will correspond to the objective function values \((1-L^A)f^{B_A}\) and \((1-L^B)f^{B_C}\). The family design problem Eq. (10) is solved next to obtain a Pareto-optimal design, searching for the point closest to the null-platform design.

5 Designing Automotive Body Side Frame Variants

We now consider a family of side frames for an automotive body with two variants. A variant can be defined by changing the functional requirements and/or the geometry of the model. In this study variants A and B are designed for minimum mass and maximum stiffness (minimum deflection), respectively.

The side frame of the automotive body is modeled simply as an assembly of ten beam elements and seven flexible joints (Fig. 8). A finite element solver is used to compute deflections, stresses, and body mass for different values of the rectangular cross section parameters of the beams (width \(b\), height \(h\), and thickness \(t\)). Two loading cases (bending and torsion) are considered, as depicted in Fig. 9 where \(F=1500\) lbs. and \(Q=1650\) lbs. The vehicle dimensions used for this example are \(h=30\) in., \(a=20\) in., \(b=40\) in., \(c=40\) in., and \(d=10\) in. To compute \(\alpha\) and \(\beta\) the following equations are used: \(\alpha=(c+d)h/(a+b+c+d)\) and \(\beta=(a+b)h/(a+b+c+d)\), where for this example \(\alpha=10/33\) and \(\beta=6/33\). The engine and rear compartments are included in the model as reaction forces applied at the connection of the “A” and hinge pillars and at the centerpoint of the “C” pillar for the bending load case. Torsion is represented by a horizontal force applied at the joint connecting the “B” pillar and the roof; this force simulates the shear that the structure undergoes under such loading and provides a torsional displacement, \(\delta_B\). An overall bending displacement is calculated as \(\delta_B=\alpha \Delta_1+\beta \Delta_2+\Delta_3\). Each component is represented by three design variables. This means that a component can be shared within the family if all three design variables have equal values. However, it may be possible to consider some other form of sharing if only one or two design variables have equal values, for example, from a manufacturing point of view as discussed in Sec. 2. In this regard, all variables are treated as platform candidates in this study.

At first the optimal design problem is solved individually for each variant to obtain null-platform optima \(f^{B_A}\). Two variants are considered: Variant A is designed with a minimum mass objective subject to a stiffness constraint represented by deflections

\[
\min_{x^A} f^A(x^A) = m
\]

subject to \(g^A(x^A) \leq 0\)

\[
\delta_1 \leq 0.4 \text{ in.}
\]

\[
\delta_2 \leq 0.8 \text{ in.}
\]  \hspace{1cm} (12)

Variant B is designed with a maximum stiffness objective subject to a mass constraint

\[
\min_{x^B} f^B(x^B) = \delta_1 + \delta_2
\]

subject to \(g^B(x^B) \leq 0\)

\[
m \leq 250 \text{ lbs.}
\]  \hspace{1cm} (13)

Maximal stress constraints, \(\sigma\), for each beam element are taken into account for all three variants. A modulus of elasticity for steel of 30 MPsi. is used along with a safety factor equal to 3.0. The computed null-platform optima for the considered variants are 53.4 lbs. for Variant A and 0.587 in. for Variant B.

Having computed the null-platform optima \(f^{B_A}\), the commonality decision problem [Eq. (8)] can be solved for different values of the loss factor \(L^o\). Experience to date shows that 0,025 is a good value for the parameter \(\alpha\) in Eq. (6). Further investigation is necessary to determine a recommended value of \(\alpha\) for general use.

In the side frame problem there are 24 design variables representing the cross-sectional variables \(b\) (width), \(h\) (height), and \(t\) (thickness) of the beams. The candidate platform set \(S\) is defined by allowing each design variable in Variant A to be shared only with the same variable of the corresponding component in Variant B. Therefore there are \(2^{24}\) possible sharing combinations (platforms).

Equation (8) is solved to determine the “maximal” feasible platform under performance bounds set equal for both variants. A tolerance of 0.5% on the commonality constraints in the objective was used to determine which variables will be shared among the two variants. From the null-platform results, twelve variables were found to be “naturally” shared by inspection, i.e., they had the same optimal values in both designs. Allowing a performance loss of 1%, 5%, 10%, 20%, and 50% resulted in a commonality...
decision of sharing 17, 20, 21, 22, and 24 variables, respectively. The trade-off between sharing and performance is shown in Fig. 10. This trade-off is analogous to the Pareto set that could be generated solving the combinatorial optimal design problem Eq. (2).

The optimal family problem Eq. (10) is solved next, with results presented in Table 1. The performance results obtained after solving the family problem for all platforms selected is shown. The performance in all cases are within the specified bounds. The optimal values of the product family design variables based on the null platform, the platform determined by accepting a 5% performance loss, and the total platform are compared in Table 2.

In an attempt to validate the results, a reduced version of Eq. (2) is solved. "Naturally shared" variables, as determined by the individual variant optimizations, are shared. Since twelve out of the sixteen width and height variable values of the two variants are always equal, we considered width and height as shared variables, and reduced the size of the combinatorial problem by defining the platform to include these sixteen variables. The problem size is thus reduced to $2^8$ platform combinations.

A “top-down” algorithm was implemented by starting with sharing all eight component thicknesses (total platform). If the performance bounds were exceeded, we moved a “level” down by decreasing the number of candidates for sharing from eight to seven; eight different platforms of sharing seven thicknesses were then considered, and so on, until at least one platform was found, for which the performance bounds for a given “level” were not exceeded. Note that it is possible that more than one platform may satisfy the performance bounds at a given “level.”

One platform was obtained for each of the loss factors 1%, 5% or 50%, by solving the commonality decision problem Eq. (8). For loss factors of 10% and 20%, two and seven feasible platforms were found, respectively, each once again containing the same number of shared variables as determined by solving Eq. (8). It is encouraging that for both cases the platform obtained by solving the commonality decision problem [Eq. (8)] is the one that corresponds to least performance loss where multiple platforms were found. Intuitively this makes sense: since Eq. (8) is solved over a continuous design space, it is natural that components that have less impact (sensitivity) on the performance will be shared first.

6 Multiple Variants

When product designers study the implementation of a platform the number of products in the family will likely be more than two. The examples provided in this section demonstrate the use of the formulation for families with multiple products.

The size of the problem increases substantially with the number of variants. The number of possible platforms can be calculated as

$$\text{Number of platform combinations} = \left(1 + \sum_{i=2}^{m} \frac{m!}{i!(m-i)!}\right)^n$$

where $m$ is the number of variants and $n$ is the number of shareable elements. The assumption made for this calculation is that each variant has the same components to be shared. The unity in the equation allows inclusion of the null platform as a possible combination. Likewise, the total platform is included in the summation for $i=m$.

The number of terms in the objective function of Eq. (8) is equal to the number of sharing possibilities among the variants. Assuming one variant with one shareable component, there exist three sharing possibilities among them, for four variants there exist six sharing possibilities, for five variants there exist ten sharing possibilities, and so on. The number of sharing possibilities for $n$ shareable components in $m$ products can be calculated by

**Table 1 Optimal product family design results and associated performance losses**

<table>
<thead>
<tr>
<th>Variant</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Null platform</td>
<td>53.4 lbs.</td>
<td>0.587 in.</td>
</tr>
<tr>
<td>Platform of 17 variables</td>
<td>53.9 lbs.</td>
<td>0.593 in.</td>
</tr>
<tr>
<td>Performance loss</td>
<td>0.999%</td>
<td>0.972%</td>
</tr>
<tr>
<td>Platform of 20 variables</td>
<td>55.6 lbs.</td>
<td>0.616 in.</td>
</tr>
<tr>
<td>Performance loss</td>
<td>4.17%</td>
<td>5.01%</td>
</tr>
<tr>
<td>Platform of 21 variables</td>
<td>56.0 lbs.</td>
<td>0.646 in.</td>
</tr>
<tr>
<td>Performance loss</td>
<td>4.87%</td>
<td>9.96%</td>
</tr>
<tr>
<td>Platform of 22 variables</td>
<td>58.3 lbs.</td>
<td>0.687 in.</td>
</tr>
<tr>
<td>Performance loss</td>
<td>9.20%</td>
<td>17.00%</td>
</tr>
<tr>
<td>Total platform of 24 variables</td>
<td>73.2 lbs.</td>
<td>0.881 in.</td>
</tr>
<tr>
<td>Performance loss</td>
<td>37.1%</td>
<td>50.0%</td>
</tr>
</tbody>
</table>
The number of terms will grow significantly with additional products or components, but not as fast as in the original combinatorial problem. The number of possible platforms is reduced by limiting the number of components that the designer considers as shareable.

Consider the design of three automotive body side-frames (Fig. 11). Variant A is an economy vehicle, where the weight is minimized with constraints on the bending and torsional rigidity. The design problem is the same as the one given in Eq. (12). Variant B is a sporty vehicle, where the torsional rigidity is maximized with respect to constraints on the weight and bending stiffness

\[
\min_{\mathbf{x}^B} f^B(\mathbf{x}^B) = \delta_i
\]

subject to \( g^B(\mathbf{x}^B) \leq \mathbf{0} \)

\( m \leq 250 \text{ lbs} \)

\( \delta_i \leq 0.4 \text{ in.} \) \hspace{1cm} (16)

Variant C is a stretched frame, where the lengths of the roof rail and rocker have been extended 20 in. (symmetrically about the “B” pillar); the objective is to minimize the bending displacement subject to weight and torsional rigidity constraints

\[
\min_{\mathbf{x}^C} f^C(\mathbf{x}^C) = \delta_i
\]

subject to \( g^C(\mathbf{x}^C) \leq \mathbf{0} \)

\( m \leq 250 \text{ lbs} \)

\( \delta_i \leq 0.8 \text{ in.} \) \hspace{1cm} (17)

All three vehicle models use the same \( \alpha \) and \( \beta \) values as in the previous example. The optimal objective function values for the three variants are determined to be 53.4 lbs., 0.373 in., and 0.312 in., respectively.

From Eq. (14) the number of platform combinations is calculated as \( 60 \times 10^{15} \). For the three-variant example, given that the same 24 design variables define each variant as before, Eq. (15) yields 72 terms necessary to capture all sharing combinations.

In the commonality decision problem Eq. (8), the objective for each variable \( i \) will have three terms,
The performance bounds are set at 1%, 5%, and 10% across variants. Once again a tolerance of 0.05% is used for the commonality constraints in the objective. The sharing decisions, and resulting family designs for each frame member cross section are shown in Tables 3–5, respectively. With a loss factor of 1% fifteen variables can be shared among all three products, two variables between A and B, one variable between A and C, and five variables between B and C. Using a loss factor of 5% seventeen variables can be shared.

### Table 3 Commonality decisions and family design for \( L^p=1\% \)

<table>
<thead>
<tr>
<th>Component</th>
<th>Variable</th>
<th>Shared among</th>
<th>( x^A )</th>
<th>( x^B )</th>
<th>( x^C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocker</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>BC</td>
<td>0.04</td>
<td>0.48</td>
<td>0.48</td>
</tr>
<tr>
<td>Roof rail</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.65</td>
<td>1.65</td>
<td>1.65</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>BC</td>
<td>0.19</td>
<td>0.33</td>
<td>0.33</td>
</tr>
<tr>
<td>Hinge pillar</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.98</td>
<td>1.98</td>
<td>1.98</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>4.37</td>
<td>4.37</td>
<td>4.37</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>0.04</td>
<td>0.25</td>
<td>0.07</td>
</tr>
<tr>
<td>“A” pillar</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.02</td>
<td>1.02</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td>“B” pillar (lower)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.19</td>
<td>2.19</td>
<td>2.19</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>AB</td>
<td>3.30</td>
<td>3.30</td>
<td>1.66</td>
</tr>
<tr>
<td>“B” pillar (upper)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.06</td>
<td>1.06</td>
<td>1.06</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>2.19</td>
<td>2.19</td>
<td>2.19</td>
</tr>
<tr>
<td>“C” pillar (lower)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>BC</td>
<td>0.09</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>“C” pillar (upper)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>BC</td>
<td>0.11</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>AB</td>
<td>1.01</td>
<td>1.01</td>
<td>1.01</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>3.30</td>
<td>3.30</td>
<td>3.30</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>AB</td>
<td>0.04</td>
<td>0.04</td>
<td>0.26</td>
</tr>
</tbody>
</table>

### Table 4 Commonality decision and family design for \( L^p=5\% \)

<table>
<thead>
<tr>
<th>Component</th>
<th>Variable</th>
<th>Shared among</th>
<th>( x^A )</th>
<th>( x^B )</th>
<th>( x^C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocker</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>BC</td>
<td>0.04</td>
<td>0.46</td>
<td>0.46</td>
</tr>
<tr>
<td>Roof rail</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.65</td>
<td>1.65</td>
<td>1.65</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>BC</td>
<td>0.20</td>
<td>0.33</td>
<td>0.33</td>
</tr>
<tr>
<td>Hinge pillar</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.90</td>
<td>1.90</td>
<td>1.90</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>BC</td>
<td>0.04</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>“A” pillar</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>0.04</td>
<td>0.35</td>
<td>0.35</td>
</tr>
<tr>
<td>“B” pillar (lower)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>AB</td>
<td>3.30</td>
<td>3.30</td>
<td>1.88</td>
</tr>
<tr>
<td>“B” pillar (upper)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>AC</td>
<td>0.12</td>
<td>0.50</td>
<td>0.12</td>
</tr>
<tr>
<td>“C” pillar (lower)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
</tr>
<tr>
<td>“C” pillar (upper)</td>
<td>( h ) (in.)</td>
<td>ABC</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>BC</td>
<td>0.11</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>1.05</td>
<td>1.05</td>
<td>1.05</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>3.30</td>
<td>3.30</td>
<td>3.30</td>
</tr>
<tr>
<td></td>
<td>( i ) (in.)</td>
<td>ABC</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04</td>
</tr>
</tbody>
</table>
shared among all three products, one variable between A and B, one variable between A and C, and five variables between B and C. Finally using a loss factor of 10% twenty variables can be shared among all three products, no variables between A and B, one variable between A and C, and three variables between B and C.

The results from performing the family design optimization can be seen in Table 6. For all three cases performance losses do not exceed the bounds.

7 Aggregating Variables in Shareable Components

So far we examined what design variables can be shared among products. In fact the designer may need to know explicitly whether an entire component can be shared. This section demonstrates how this decision can be reached using the side frame example.

The components for each of the body side frames are included in the component set $O_p$ (because all three products have an identical topology, the set $O_p$ is consistent among variants),

$$O_p = \{\text{Rocker, Roof rail, Hinge pillar, “A” pillar,}$$

“B” pillar(lower)$, “B” pillar(upper)$, “C” pillar(lower)$, “C” pillar(upper)$

Table 5 Commonality decision and family design for $L_p=10\%$

<table>
<thead>
<tr>
<th>Component</th>
<th>Variable</th>
<th>Shared among</th>
<th>Family design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocker</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>$x_{A}^{C}$</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>2.20</td>
</tr>
<tr>
<td>$t$ (in.)</td>
<td></td>
<td>BC</td>
<td>4.40</td>
</tr>
<tr>
<td>$L_t$</td>
<td></td>
<td></td>
<td>0.04</td>
</tr>
<tr>
<td>Roof rail</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>1.04</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>1.10</td>
</tr>
<tr>
<td>$t$ (in.)</td>
<td></td>
<td>ABC</td>
<td>0.18</td>
</tr>
<tr>
<td>Hinge pillar</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>2.10</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>1.00</td>
</tr>
<tr>
<td>$t$ (in.)</td>
<td></td>
<td>BC</td>
<td>0.16</td>
</tr>
<tr>
<td>“A” pillar</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>1.00</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>1.10</td>
</tr>
<tr>
<td>$t$ (in.)</td>
<td></td>
<td>ABC</td>
<td>0.12</td>
</tr>
<tr>
<td>“B” pillar(lower)</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>2.20</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>1.00</td>
</tr>
<tr>
<td>$t$ (in.)</td>
<td></td>
<td>A C</td>
<td>1.10</td>
</tr>
<tr>
<td>“B” pillar(upper)</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>2.20</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>0.05</td>
</tr>
<tr>
<td>“C” pillar(lower)</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>2.20</td>
</tr>
<tr>
<td>$h$ (in.)</td>
<td></td>
<td>ABC</td>
<td>0.05</td>
</tr>
<tr>
<td>“C” pillar(upper)</td>
<td>$b$ (in.)</td>
<td>ABC</td>
<td>2.20</td>
</tr>
</tbody>
</table>

The design variables of the product are then mapped into vectors of design variables which correspond to particular product components as follows:

$$x^i = [x_{A}^{i}, x_{B}^{i}, x_{C}^{i}]$$

where the three vector entries for each component are the width, height, and thickness of the component. With the design vector defined for each component the final step is to modify the objective function of the commonality decision problem to deal with decisions based on vectors rather than scalars. This is accomplished by simply using the $l_2$ norm to compute the difference in design.

$$\min_{x \in \{x_A, x_B, x_C\}} \sum_{i=1}^{n=8} (D_A(\|x_{A}^{i} - x_{A}^{f}\|) + D_B(\|x_{B}^{i} - x_{B}^{f}\|) + D_C(\|x_{C}^{i} - x_{C}^{f}\|)$$

Since the focus is now on component selection, the roof and rocker of the stretched vehicle (C) is not shareable with the other components.
two vehicles. This is because these are the only two components that do not share a common length with the other two variants. Therefore, when \( i = 1, 2 \) the last two terms are removed in Eq. (19). The platform selection process is now performed with a loss factor of 5% and 10%. The tolerance of 0.05% is used again for the commonality constraints in the objective. The platform selection and corresponding family designs for each frame member cross section can be found in Tables 7 and 8.

Figure 12 represents the shared components between pairs of products for each of the two loss factors. The platform among all three products is the intersection of the shared components for each of the sharing pairs. For a loss factor of 5% there are five

<table>
<thead>
<tr>
<th>Component</th>
<th>Variable</th>
<th>Shared among</th>
<th>( x_A^{.5} )</th>
<th>( x_B^{.5} )</th>
<th>( x_C^{.5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocker</td>
<td>( b ) (in.)</td>
<td></td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.04</td>
<td>0.45</td>
<td>0.50</td>
</tr>
<tr>
<td>Roof rail</td>
<td>( b ) (in.)</td>
<td></td>
<td>1.65</td>
<td>1.65</td>
<td>1.65</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.19</td>
<td>0.48</td>
<td>0.43</td>
</tr>
<tr>
<td>Hinge pillar</td>
<td>( b ) (in.)</td>
<td>A C</td>
<td>2.13</td>
<td>2.14</td>
<td>2.13</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.05</td>
<td>0.29</td>
<td>0.05</td>
</tr>
<tr>
<td>“A” pillar</td>
<td>( b ) (in.)</td>
<td>BC</td>
<td>0.95</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.04</td>
<td>0.33</td>
<td>0.33</td>
</tr>
<tr>
<td>“B” pillar</td>
<td>( b ) (in.)</td>
<td>(lower)</td>
<td>2.20</td>
<td>2.20</td>
<td>2.10</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>3.30</td>
<td>3.30</td>
<td>2.41</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.12</td>
<td>0.50</td>
<td>0.16</td>
</tr>
<tr>
<td>“B” pillar</td>
<td>( b ) (in.)</td>
<td>(upper)</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
</tr>
<tr>
<td>“C” pillar</td>
<td>( b ) (in.)</td>
<td>(lower)</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.11</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td>“C” pillar</td>
<td>( b ) (in.)</td>
<td>(upper)</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>3.30</td>
<td>3.30</td>
<td>3.30</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Variable</th>
<th>Shared among</th>
<th>( x_A^{.1} )</th>
<th>( x_B^{.1} )</th>
<th>( x_C^{.1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocker</td>
<td>( b ) (in.)</td>
<td></td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.04</td>
<td>0.49</td>
<td>0.50</td>
</tr>
<tr>
<td>Roof rail</td>
<td>( b ) (in.)</td>
<td>AB</td>
<td>1.65</td>
<td>1.65</td>
<td>1.65</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.18</td>
<td>0.18</td>
<td>0.45</td>
</tr>
<tr>
<td>Hinge pillar</td>
<td>( b ) (in.)</td>
<td>A C</td>
<td>2.08</td>
<td>2.20</td>
<td>2.08</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>4.40</td>
<td>4.40</td>
<td>4.40</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.04</td>
<td>0.39</td>
<td>0.04</td>
</tr>
<tr>
<td>“A” pillar</td>
<td>( b ) (in.)</td>
<td>ABC</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>“B” pillar</td>
<td>( b ) (in.)</td>
<td>(lower)</td>
<td>2.05</td>
<td>2.20</td>
<td>2.05</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>3.29</td>
<td>3.30</td>
<td>3.29</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.13</td>
<td>0.50</td>
<td>0.13</td>
</tr>
<tr>
<td>“B” pillar</td>
<td>( b ) (in.)</td>
<td>(upper)</td>
<td>1.09</td>
<td>1.09</td>
<td>1.09</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
</tr>
<tr>
<td>“C” pillar</td>
<td>( b ) (in.)</td>
<td>(lower)</td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>2.20</td>
<td>2.20</td>
<td>2.20</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.11</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td>“C” pillar</td>
<td>( b ) (in.)</td>
<td>(upper)</td>
<td>1.07</td>
<td>1.07</td>
<td>1.07</td>
</tr>
<tr>
<td></td>
<td>( h ) (in.)</td>
<td></td>
<td>3.30</td>
<td>3.30</td>
<td>3.30</td>
</tr>
<tr>
<td></td>
<td>( t ) (in.)</td>
<td></td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 7 Commonality decision (based on components) and family design for \( L^p=5\% \)

Table 8 Commonality decision (based on components) and family design for \( L^p=10\% \)
components shared, with the upper “B” and “C” pillars being shared among all three variants. For a loss factor of 10% there are seven components shared, now including the “A” pillar in the platform between all three products. Actual losses are computed after solving Eq. (10) and are reported in Table 9. They are within the allowable bounds on deviation from the null-platform designs.

Summarizing, the methodology has been applied to family design problems including multiple products. In addition, the formulation was used for performing platform selection based on component selection (all variables of component shared). These results allow the visualization of modularity as well: components or sets of components which produce variety are defined as modules, and are not part of the platform. A thorough discussion of modularity is beyond the scope of this article.

8 Conclusions

The proposed methodology integrates platform selection under performance bounds with optimal design of product families. The designer can decide (potentially by using business and marketing data) what performance losses are acceptable relative to individual product variant optimality. Component sharing is determined through the solution of the relaxed commonality maximization combinatorial problem subject to these performance bounds. The formulation of the commonality decision problem allows the designer to assign different weights to express sharing preferences. The optimal product family design problem is solved to obtain a point on the Pareto set associated with the determined platform that is closest to the null-platform point. A family of automotive body side frames has been used to demonstrate the proposed methodology. Results have been validated by comparing them to those obtained by solving a reduced version of the original combinatorial commonality problem. It is possible to conclude that the methodology can be helpful in addressing the platform selection problem, which may be intractable in its original combinatorial form. Problems have been examined which have a significantly larger number of sharing combinations than has previously been solved in the design literature.

Gradient-based algorithms are recommended to solve the commonality decision problem Eq. (8). It is desirable to compute gradients analytically. The use of the function $D_p$ as defined in Eq. (6) makes this possible. In the case that simulation-based models are used to evaluate the performance constraints, it is important to use appropriate finite-difference steps for evaluating the constraint functions. For the application considered in this paper, we used the MATLAB implementation of the sequential quadratic programming (SQP) algorithm. Design variables, objective, and constraints were scaled to the order of one. On average, for the three-product examples in this paper, on the order of 100 iterations are necessary to solve the commonality decision problem. In examples so far, different starting points have resulted in the same optimal solution. One important assumption that has been made is that the design variables and functions are continuous. This method is not applicable to a catalog selection problem or where design variables are discrete with no underlying continuous trends.

Solving the commonality decision problem [Eq. (8)] may become inefficient as the number of products and/or the number of shareable components increases dramatically beyond the practical limitations of an algorithm such as SQP. Future work will include looking at implementing the ranking methods to reduce the order of the problem by filtering out components with low impact on sharing [13,14]. Metamodels can also be used to decrease computational expense and reduce function noise, which is detrimental when applying gradient-based methods. Lastly, extending the methodology to hierarchical product family problems can prove useful for solving large-scale problems.

Acknowledgments

This research was partially supported by the Automotive Research Center, a U.S. Army Center of Excellence in Modeling and Simulation of Ground Vehicles, by a U.S. Army Dual-Use Science and Technology Project, and by the General Motors Collaborative Research Laboratory at the University of Michigan. This support is gratefully acknowledged. N. Kikuchi provided the automotive side frame model.

References


A RIGOROUS FRAMEWORK FOR MAKING COMMONALITY AND MODULARITY DECISIONS IN OPTIMAL DESIGN OF PRODUCT FAMILIES

Ryan Fellini, Michael Kokkolaras, and Panos Y. Papalambros

Abstract
This article proposes a set of math-based approaches for making commonality and modularity decisions when designing product families. The performance of products that share some components is usually compromised relative to the individual optimum. This deviation occurs because of the commonality constraints that are included in the optimal design problem, especially when the objectives in the multicriteria formulation are conflicting. Choosing components for sharing may depend on what performance deviations can be tolerated. We present rigorous strategies for choosing components to be shared without exceeding user-specified bounds on performance. In addition, we consider modularity to be an outcome of the commonality decision process.

Keywords: Optimal Design, Product Families, Commonality, Modularity

1 Introduction
We refer to products that have similar architecture but different functional requirements as variants. Product variants can be derived based on a platform, i.e., a set of common parts, in which case they form a product family. Product platforms enable the development of variants for rapid adjustment to changing market needs while keeping development costs and time-cycles low [1,2]. The functional requirements of product variants may be conflicting, in which case family optimal designs are compromised relative to individually optimized designs [3]. The design challenge is to maximize commonality while minimizing individual performance losses.

We focus on math-based methodologies for determining which components should be shared. Chen et al. and Nayak et al. use robust design principles to select the platform for a family of scalable products [4,5]. Gonzalez-Zugasti et al. and Gonzalez-Zugasti and Otto cluster parts into modules to reduce the problem size, and then solve the combinatorial design problem for modular product families [6,7]. Fujita and Yoshida follow a similar approach for simultaneous optimization of module combination and attributes [8]. The latter is based on previous work of Fujita et al. [9,10,11]. A genetic algorithm (GA) is linked with sequential quadratic programming (SQP) in the above methods. The GA is used to choose the modules to be shared and SQP is used to solve the family design problem. D’Souza, B. and Simpson have also adopted GAs for solving the commonality selection and family design problems [12]. The number of possible combinations increases exponentially the number of products and/or variables. Therefore, combinatorial algorithms like GAs may be insufficient for solving even problems of modest size.
In this article we integrate two approaches developed in previous work into a rigorous framework of commonality strategies. The first approach uses first-order information, obtained from solving the individual optimal design problems, to compute a metric for performance deviations attributed to component sharing [13]. The second approach uses a relaxed formulation of the combinatorial problem to maximize commonality, while satisfying designer-specified bounds on individual performance losses [14]. The main idea is to use the first approach as a filter to reduce the platform selection problem size and the second approach to maximize commonality while minimizing individual performance losses. However, there are more than one ways to combine the two approaches. We propose some here.

The article is organized as follows: after presenting some necessary definitions to establish a common glossary, we review the two approaches. We then formulate the general combined methodology, and discuss different options. We conclude with a discussion on how modularity information can be extracted from the commonality decision process.

2 Definitions
A product is an artifact created with the intent to serve user needs, i.e., to satisfy some functional requirements. In the present context the most fundamental term used when discussing product design is the term model. A model is a mathematical representation of the artifact and is a core element of engineering design and analysis. A model accepts a set of variables as inputs and returns a set of responses as outputs. A collection of products will be represented by the set \( P = \{ p_1, p_2, \ldots \} \) used to distinguish each product \( p \in P \). A product \( p \) is associated with a vector of input variables \( x^p \in \mathbb{R}^n \) and a vector of output responses \( R^p \in \mathbb{R}^m \). In the context of a model the vector \( x^p \) is assumed to describe a given design fully, namely, if values are assigned to all the components of \( x^p \) then \( p \) is defined uniquely. Similarly, \( R^p \) is assumed to represent all responses of interest in evaluating the attributes of a given design. Clearly, these are strong assumptions but necessary in the use of any analytical tool. The designer is expected to augment the insights gained from analysis to reach the best design possible by including non-analytical considerations.

Assuming that we can map the functional requirements of a product to model responses, we formulate the optimal design model for a product \( p \) as

\[
\begin{align*}
\max_{x^p} & \quad f^p(x^p) \\
\text{subject to} & \quad g^p(x^p) \leq 0 \\
& \quad h^p(x^p) = 0,
\end{align*}
\]  

(1)

where the objective function \( f^p \), representing performance, is maximized, and additional product requirements are represented by design inequality and equality constraint functions \( g^p \) and \( h^p \), respectively.

A component is defined as a manufactured object that is the smallest (indivisible) element of a product. An assumption is made that the product can readily be decomposed into its base components. A component will have its own set of design variables, \( x^c \subset x^p \), which are a subset of the product design variables. An assembly is an object constructed from a set of connected components.
It should be noted that the point of reference when defining a product can easily change. The engine of a vehicle is an assembly, which is part of an automobile, but could be designed as a product composed of various assemblies and components such as the exhaust system, fuel injection system, etc. In this case the point of reference changes and the engine is at the “top level” of the product hierarchical description.

The product architecture is the configuration of components and assemblies within a product and includes the set of instructions for assembly of the product. The costs associated with manufacturing are not considered in this work. Variable costs having to do with mass production, etc., are also ignored.

A product platform is the set of all elements, interfaces, manufacturing and systems processes that are common in a set of products. This article focuses on commonality in product design only, therefore it is limited to parts sharing only. The following notation is used to describe a product platform. The set $S_{pq}$ consists of the index pairs of elements that are shared between two products $p$ and $q$. The set $S = \{S_{pq} | p, q \in P; p < q\}$ describes element sharing throughout the family. The empty set $S_{\emptyset}$ defines the null platform (no sharing between products).

Various types of sharing can be used based on the definitions in the previous section. In component sharing one or more components are common across a family of products. Likewise, assemblies can be shared between products. In addition, it is possible to share “scaled” versions of components. Mathematically this can be described as variable sharing, where components are based on a platform (of variables) themselves.

A product family is the set of products that share product platform. The set $S$ maps the relationships between products in $P$. The product family can have an objective $f^F$ which is a function of the design variables of the entire family, $x^F = [x^p, x^{p_2}, \ldots]$. Such a family objective could be, e.g., cost or profit. Likewise, we can have family constraints $g^F$ and $h^F$, e.g., for a family of automobiles there may be constraints on production capacity and/or corporate average fuel economy (CAFE) penalties. Assuming that the commonality decisions have been made, the multiobjective optimal family design problem is formulated as

$$\begin{align*}
\max_{x^F} & \quad \{f^F, f^p(x^p)\} \quad \forall p, q \in P, (i, j) \in S_{pq}, p < q \\
\text{subject to} & \quad g^F(x^F) \leq 0 ; \quad g^p(x^p) \leq 0 \\
& \quad h^F(x^F) = 0 ; \quad h^p(x^p) = 0 \\
& \quad x^p_i = x^q_i
\end{align*}$$

(2)

A module is a special case of a component in a product that produces variety when interchanged (cf. Figure 1).

Figure 1. Modular product with one interchangeable module.
Ideally a module should allow for the capability to modify the characteristics defining the product independently. For example, one factor in choosing a computer is sound capability. Therefore as a module the sound card can be interchanged to control this specific metric and does not influence other metrics such as the video quality or the processing speed. The assumption will also be made that a module need not be connected. For example, two components in a product that influence a particular product response, $R_i$, may be not connected, but can be exchanged simultaneously to create a new product. The set $M^p = \{m_1^p, m_2^p, \ldots\}$ is used to represent the modules in a product $p$. A module is formed from a subset of the components which make up a product, and we assume that components are not shared between modules. Again, this is an important assumption. $x^{m^p}$ consists of all the component design variables included in the module.

Two important aspects regarding modules is that they have an interface defined for interchangeability and that the interactions between the module and its surrounding should be minimized to afford better encapsulation of its design and function. For example, if the engine block is considered a module within a set of engine products and the piston is part of the platform, the bore of the engine block modules should fit that of the piston platform. This common interface is the platform shared amongst modules. Modules themselves can be shared by a subset of the products, forming a modular (sub)platform (cf. Figure 2).

Figure 2. Modular products with module sharing among a subset of products.

An example of a modular platform can be a family of automobiles where variety is produced by interchanging engines. While two of the vehicles may use the same engine, a third vehicle may use a different engine (where the engines are interchangeable among the three vehicles). The first two vehicles would be differentiated perhaps by a second module influencing some other vehicle metric.

3 Platform Selection

In this section we review two approaches developed for solving the commonality selection problem.

3.1 Performance Deviation Vector

In this approach we use optimality and sensitivity information obtained from solving the individual optimal design problems to estimate potential deviation from the null-platform optimal design incurred by sharing parts with other product variants. Specifically, assuming that individual optima lie “close” enough to each other and that the family design will be in their convex hull, we use a first-order Taylor series approximation to derive an upper bound
Π, on the performance deviation due to sharing variable \( x_i \) [13]. For two variants A and B this upper bound is given by

\[
\Pi_i = (1 - \lambda_i) \left( \nabla_i f^{A,o} \delta_i + \max_{j \in G^A} \left( \nabla_i g^{A,o} \delta_i, 0 \right) \right) + \lambda_i \left( \nabla_i f^{B,o} \delta_i + \max_{j \in G^B} \left( \nabla_i g^{B,o} \delta_i, 0 \right) \right),
\]

where \( 0 \leq \lambda_i \leq 1 \), \( \delta_i = |x_i^{A,o} - x_i^{B,o}| \), the superscript \( o \) denotes null-platform, and \( G^p \) is the set of active constraints of product \( p \). The designer can sort the values \( \Pi_i \) \( \forall i \) in a so-called performance deviation vector (PDV) and base his decision on sharing according to some threshold. An example on the possible use of the PDV is illustrated in Figure 6 [13]. In this example, there is a total of 63 variables that can be shared. The first 25 variables have zero performance deviation, i.e., the optimal design values corresponding to the null-platform optima were identical. Therefore, these variables correspond to “naturally” shared components. The designer may want to share nineteen additional variables since the performance deviation value is below 0.01 for these variables. In this case, the designer chooses according to an absolute threshold. An alternative is to choose according to a relative increase of performance deviation, i.e., using the ratio \( (\Pi_{i+1} - \Pi_i) / \Pi_i \).

![Figure 6. Graphical representation of the sorted PDV.](image)

The performance deviation vector approach is computationally quite affordable but relies on the above mentioned assumptions and involves the heuristic of selecting a threshold for making commonality decisions. It is thus more appropriate for what we refer to as “mild” variants.

### 3.2 Relaxed Combinatorial Problem

This approach enables the designer to choose what performance losses are acceptable relative to null-platform optima. Component sharing is then determined through the solution of a relaxed combinatorial commonality maximization combinatorial problem subject to these performance bounds. The original mixed-discrete optimization problem has been reformulated using a continuous function to approximate the binary decisions [14]:

\[
\begin{align*}
\min_{x \in \mathbb{R}^{x_1}, x_2, \ldots} & \sum_{(i,j) \in P} D_{ij} \left( x_i^p - x_j^q \right) \quad \forall p, q \in P, (i, j) \in S^{pq}, p < q \\
\text{subject to} & \quad g^p(x^p) \leq 0 \\
& \quad h^p(x^p) = 0 \\
& \quad f^p(x^p) \geq (1 - L^p) f^{p,o},
\end{align*}
\]
where \( D_a(x^p_i - x^q_j) = 1 - 1/((x^p_i - x^q_j)/\alpha)^2 + 1 \). In our experience, a good value for \( \alpha \) is 0.05. The designer can solve Problem (3) for different loss bounds \( L' \) until he/she is satisfied with the tradeoff between maximizing commonality and minimizing individual performance losses. The relaxed combinatorial approach is free of simplifying assumptions but is computationally expensive, especially as the number of product variants and sharing candidates increases.

4 Combined Commonality Strategies

In this section we propose a number of available options for combining the two approaches to take advantage of their strengths. The first approach has the advantage of being very inexpensive and allows for a reasonable approximate ranking of the variables “shareability”. The second approach is quite accurate in that one can specify a bound on performance deviation and then optimize to maximize commonality with respect to this bound. One modification to the first-order method must be made to facilitate the integration of the two approaches. This modification is necessary for commonality decisions involving components specified by vectors of design variables. This is accomplished by aggregating the deviation values of the design variables defining a component \( c \) into a single performance deviation value \( \Pi c \).

4.1 Linking the first-order method with an optimizer

The first option is to link the performance deviation vector approach to an optimization algorithm. This is based on the assumption that the components have been sorted “correctly” with respect to their influence to the product design. The algorithm can be as simple as the bisection or golden section methods or one may implement more sophisticated derivative-free algorithms. Gradient-based algorithms should only be used with caution because of the noisy nature of the vector. The idea is as follows. We can compute the maximum loss on performance by finding the so-called total-platform designs by solving the family design problem - Problem (2) - assuming that all design variables are shared. Since our objective is to maximize component sharing, we want to move to the right on the horizontal axis. At each iteration we optimize the family designs given the selected level of sharing by solving Problem (2), and determine our actual performance deviation. If performance loss is acceptable we select a bigger platform, otherwise a smaller one. Convergence is achieved when the desired bound on performance loss is met given maximum commonality. Figure 7 illustrates this approach for three iterations of the bisection method.

![Figure 7. Optimizing over the performance deviation vector.](image-url)
4.2 Using the first-order method as a filter

Another straightforward approach to combining the approaches is to use the first-order method as a filter to reduce the problem size solved by the relaxed combinatorial problem. We will review two alternative ways to do this.

The first alternative uses the first-order method to assume an initial set of components that can be shared. This is illustrated on the left plot of Figure 8.

![Figure 8](image)

Figure 8. Using the performance deviation vector to reduce the relaxed combinatorial problem size.

The designer decides to share a number of components by means of a threshold according to the discussion in Section 3.1. Solving the relaxed combinatorial problem on the remaining candidates, one can determine if more components can be shared.

Alternatively, one can use the first-order method to determine the candidate platform and solve the relaxed combinatorial problem to determine if, or how many, of the components in the candidate platform can be actually shared. This alternative is illustrated on the right plot of Figure 8.

4.3 Combining the approaches into an iterative method

The previous options can also be combined into an iterative method. The motivation is to formulate a strategy that uses the performance deviation vector information to determine the smallest problem size possible for the relaxed combinatorial problem. In the example of Figure 9, a threshold of 0.01 we observe that the size of the relaxed combinatorial problem is smaller when considering the remaining candidates. If we happen to determine that we cannot share any additional components, we may want to reduce the threshold and solve the relaxed combinatorial problem on a subset of the initial candidate platform, and so on.

![Figure 9](image)

Figure 9. Iterative solution of the relaxed combinatorial problem.
This proposed iterative method is summarized in the following diagram:

5 Modularity Decisions

After making commonality decisions by means of one of the alternatives described in the previous section we are left with a set of components which provide the variety in the family. The variants are derived by changing these components. Therefore, components that are part of the module set are readily identified as the ones that are not being shared by all products. Subplatforms (sharing between a subset of products) represent in effect the sharing of a particular module. Note that we can also consider the platform itself as a module, namely, a module that produces no variety.

The final step is to cluster components that are now members of the module set $M$ into individual modules $m^p_i$. This can be accomplished by using partitioning algorithms [15] to cluster components that impact various critical responses of the product (cf. Figure 10).

![Figure 10. Extracting modularity from partitioned dependency table.](image)
These algorithms can be applied to the functional dependency table (FDT) of the modules \( m^p \). This partitioning also aids the designer in understanding the interactions in order to choose modules that can be designed and interchanged independently.

6 Conclusions

A rigorous framework for making commonality decisions when designing a product family has been developed by combining two previously developed math-based approaches. Several combination alternatives have been presented to provide more than one option to the designer, who can utilize them on a case-by-case basis. It has been also demonstrated that modularity can be extracted as a byproduct of the commonality decision process. Some of the proposed strategies of this article have been applied successfully to automotive body structure and engine family design problems [16,17].

7 Acknowledgement

This research was partially supported by the Automotive Research Center, a US Army Center of Excellence in Modeling and Simulation of Ground Vehicles, by a US Army Dual-Use Science and Technology Project, and by the General Motors Collaborative Research Laboratory at the University of Michigan. This support is gratefully acknowledged.

References


For more information please contact:

Ryan Fellini, University of Michigan, Department of Mechanical Engineering,
2350 Hayward St., Ann Arbor, Michigan, 48109-2125, United States
Tel: +1 734 647-8401, Fax: +1 734 647-8403, E-mail: rfellini@umich.edu
URL: http://ode.engin.umich.edu
Quantitative Platform Selection in Optimal Design of Product Families, with Application to Automotive Engine Design

Ryan Fellini, Michael Kokkolaras, and Panos Y. Papalambros
{rfellini, mk, pyp}@umich.edu
Department of Mechanical Engineering
University of Michigan
Ann Arbor, Michigan, USA

Corresponding author: Ryan Fellini
Department of Mechanical Engineering
University of Michigan
2350 Hayward
Ann Arbor, Michigan 48109-2125
USA
E-mail: rfellini@umich.edu
Phone: +1 734 936-2624
Fax: +1 734 647-8403

Short title: “Platform Selection in Optimal Design of Product Families”

Number of pages: 38
Number of tables: 6
Number of figures: 7
Quantitative Platform Selection in Optimal Design of Product Families, with Application to Automotive Engine Design

Abstract: Product variants with similar architecture but different functional requirements may have common parts. We define a product family to be a set of such products, and refer to the set of common parts as the product platform. Product platforms enable rapid adjustment to changing market needs while keeping development costs and time-cycles low. In many cases, however, the individual product requirements are conflicting when designing a product family. The designer must balance the tradeoff between maximizing commonality and minimizing individual product performance deviations. The design challenge is to select the product platform that will generate family designs with minimum deviation from individual optima. We propose a methodology that combines two previous approaches developed for making commonality decisions. In the first approach optimal values and sensitivity information from the individually optimized variants are used to indicate components that are likely candidates for sharing. In the second approach a relaxed combinatorial problem is formulated to maximize sharing among variants subject to bounds on performance reduction for the individually optimized values. In the combined methodology the first approach is used to identify an initial set of shared components and define the candidate platform to be considered by the second approach. The computational load is reduced significantly and the platform-selection problem is solved in a more robust manner. The proposed methodology is demonstrated on the design of an automotive engine family.

Keywords: Optimal Design, Platform Selection, Product Families, Automotive Engines
1 Introduction

Product variants are defined as artifacts that have similar architecture but different functional requirements. Product variants may share a subset of the parts they consist of. A product family is a set of product variants, and the set of common parts in a family is referred to as a product platform. In many cases functional requirements of product variants are conflicting (Nelson et al., 1999). Platform-based family optimal designs will then be compromised relative to individually optimized designs due to a tradeoff between maximizing commonality and minimizing individual performance deviations. The design challenge is to determine (select) the platform that will generate family designs with minimum deviation from individual (null-platform) optima.

Designing product families based on product platforms enables rapid adjustment to changing market needs while keeping development costs and time-cycles low (Meyer & Lehnerd, 1997; Ericsson & Erixon, 1999). Therefore, the primary objective is to share components which are particularly costly and/or have low impact on design. Although the family design problem has been studied extensively (Siddique et al., 1998; Simpson, 1998; Simpson et al., 1999; Conner et al., 1999; Nelson et al., 1999; Messac et al., 2000; Fellini et al., 2000; Kokkolaras et al., 2002), few math-based methodologies have been proposed for determining which components should be shared ahead of the family design step. One such method uses robust design principles combined with an optimization problem to determine the platform for a family of scalable products (Chen et al., 1997; Nayak et al., 2000). Other methods focus on solving the combinatorial design problem for modular product families, since clustering parts to modules reduces the problem size significantly. Examples are a process for simultaneous module design and combination selection (Gonzalez-Zugasti et al., 1998; Gonzalez-Zugasti & Otto, 2000), and simultaneous optimization of module combination and attributes (Fujita & Yoshida, 2001). The latter is based on previous work of the same research group (Fujita et al., 1998; Fujita et al., 1999; Fujita, 2000). In these methods a genetic algorithm (GA) is linked with sequential quadratic programming (SQP).
The GA is used to choose a configuration of modules to share and SQP performs product
design optimization. Other researchers have also adopted GAs for solving the commonality
selection and family design problems (D’Souza & Simpson, 2002). The number of possible
combinations increases exponentially the number of products and/or variables. Therefore,
combinatorial algorithms like GAs may be insufficient for solving even problems of modest
size.

In the present work we combine two approaches developed in previous work to formulate
an improved commonality strategy. The first method uses first-order information obtained
from individual design optimizations to compute a metric for performance deviations at-
tributed to component sharing (Fellini et al., 2002a). This method has some limitation due
to the heuristic involved when choosing a threshold that will determine what to share. The
benefit, however, is that a large number of design variables can be filtered to identify an
initial set of shared components using information readily available from the individual op-
timizations. In addition, it may be possible in practice to identify an appropriate threshold
by means of knowledge-based design techniques. The second method uses an optimal prob-
lem formulation where the combinatorial problem is relaxed and reformulated to maximize
commonality among family members, while satisfying individual design constraints and
observing a designer-specified bound on individual performance deviations (Fellini et al.,
2002b). The designer can identify the optimal platform and obtain optimal family designs
for a number of scenarios based on the willingness to sacrifice a certain amount of individ-
ual performance. This method has the benefit of being rigorous and accurate. However, it
is computationally more expensive than the first approach. Therefore, the use of surrogate
models, e.g., artificial neural networks, is imperative. The main idea presented in this
article is to combine these methodologies using the first approach as a filter to reduce the
platform selection problem size and the second approach to maximize commonality while
minimizing individual performance deviations.

The article is organized as follows: The two approaches developed previously are re-
viewed first. The combined methodology is then formulated, and subsequently demonstrated on a family of automotive engines. Results are discussed and conclusions are drawn.

2 Problem formulation

The following definitions will be used in reviewing the previous methodologies and describing the combined strategy:

- \( \mathcal{P} = \{ p_1, p_2, \ldots \} \): set of \( m \) products
- \( \mathbf{x}^p \): column vector of design variables for the product \( p \in \mathcal{P} \)
- \( \mathcal{C}^p = \{ c^p_1, c^p_2, \ldots \} \): set of components that form a particular product \( p \)
- \( \mathbf{x}^{c^p} \): column vector of design variables for the component \( c^p \)
- \( \mathcal{S}^{pq} \): set consisting of index pairs of elements that are shared between two products \( p \) and \( q \)
- \( \mathcal{S} = \{ \mathcal{S}^{pq} \mid p, q \in \mathcal{P}; \ p < q \} \): set describes element sharing throughout the family
- \( \mathcal{S}^* \): set describing the “optimal” product platform
- \( \mathbf{x}^{p,0} \): null-platform optimal design of product \( p \), solution of Problem (1) below
- \( f^{p,0} \): null-platform optimal objective function value of product \( p \)
- \( \mathbf{x}^{p,*} \): platform-based optimal family design of product \( p \), solution of Problem (2) below
- \( f^{p,*} \): platform-based optimal family objective function value of product \( p \)
The individual optimal design problem for product variant \( p \) can be formulated as the general optimization problem

\[
\max_{x^p} f^p(x^p) \\
\text{subject to } g^p(x^p) \leq 0 \\
h^p(x^p) = 0.
\]

The multiobjective family design problem is then formulated as

\[
\max_{x = [x^{p_1}, x^{p_2}, ...]} \{ f^p(x^p) \} \quad \forall p, q \in \mathcal{P}, (i, j) \in \mathcal{S}^{pq}, p < q
\]

\[
\text{subject to } g^p(x^p) \leq 0 \\
h^p(x^p) = 0 \\
x^p_i = x^q_j
\]

where the additional equality constraints represent commonality. In general, a platform selection methodology can be summarized as follows: Quantify performance deviations by considering individual optimal designs and decide which components to be shared (i.e., determine the platform) with minimal performance deviation (cf. Figure 1); optimally design the product family around the chosen platform.

### 3 Platform selection strategy and optimal family design

In this section we will review the previously developed approaches for making commonality decisions and formulate a combined strategy for platform selection and optimal family design.

#### 3.1 Platform selection using null-platform optimality information

Optimality and sensitivity information obtained from individual product optimization is used to assess the potential deviation from the null-platform optimal design incurred by
sharing parts with other product variants (Fellini et al., 2002a). Component sharing is represented by common design variables. The following assumptions are made:

1. Self-sharing (i.e., component sharing within the same variant) is not possible.

2. Null-platform optimal designs lie “close enough” to each other.

3. The family optimum lies in the convex hull of the individual solutions, i.e., the null-platform optima.

4. Constraint inactivity remains unchanged between null-platform and family optimal designs.

We refer to the design solutions that satisfy these assumptions as “mild variants”. The formulation is derived based on a first order Taylor series approximation. Therefore, the general condition that the individual optimal designs lie relatively close to each other so that the linear approximation is valid.

We consider two product variants A and B without loss of generality. Under Assumption 3, the relation between the shared variables and the null platform can be rewritten as

\[ (x^*_i - x_i^{A,0}) = (1 - \lambda_i) \left( x_i^{B,0} - x_i^{A,0} \right), \]

with \( i \in S \). The deviation of the objective \( f \) in one variant \( A \) due to sharing of the variables \( x_i, \ i \in S \), is approximated by

\[ f^A(x^*) - f^A(x_i^{A,0}) \approx \sum_{i \in S} \nabla_i f^{A,0} \left( x_i^* - x_i^{A,0} \right) \approx \sum_{i \in S} (1 - \lambda_i) \nabla_i f^{A,0} \left( x_i^{B,0} - x_i^{A,0} \right). \]

We can then estimate the upper bound on the total performance variation of product A by

\[ \Delta^A \leq \sum_{i \in S} (1 - \lambda_i) \left( \left| \nabla_i f^{A,0} \right| \delta_i + \sum_{j \in G^A} \max \left( \nabla_i g_j^{A,0} \delta_i, 0 \right) \right), \]

where \( \delta_i = |x_i^{B,0} - x_i^{A,0}| \) and \( G^A \) is the set of indices of the active constraints at the null-platform optimum of product A. A similar upper bound can be obtained for product B.
We define next a performance deviation vector $\Pi$, whose entries correspond to performance deviations due to sharing, as follows

$$
\Pi_i = (1 - \lambda_i) \left( \mid \nabla_i f^{A,\circ} \mid \delta_i + \sum_{j \in G_A} \max(\nabla_i g_j^{A,\circ} \delta_i, 0) \right) + \lambda_i \left( \mid \nabla_i f^{B,\circ} \mid \delta_i + \sum_{j \in G_B} \max(\nabla_i g_j^{B,\circ} \delta_i, 0) \right). \tag{4}
$$

The $l_1$ norm of the vector $\Pi$ provides an upper bound on the actual performance deviation $\Delta$ of the product family

$$
\Delta = \Delta^A + \Delta^B \leq \|\Pi\|_1. \tag{5}
$$

Equations (4) and (5) can be adjusted straightforwardly for more than two products. The design variables are arranged in order of increased performance deviation value and the number of variables to share is determined by a limit on acceptable design deviations. Figure 2 illustrates a typical use of this information in making commonality decisions (Fellini et al., 2002a). From a total of 63 variables, the first 24 first variables are shared "naturally", i.e., they have the same values at the null-platform optima. Next, choosing a threshold of 0.01, the designer decides to share the first 54 variables. The information provided by the performance deviation vector allows the designer to choose in what order to begin sharing components. The method is iterative in that the designer will likely have to increase or decrease the level of sharing until the desired level of product performance is achieved. The drawbacks are the heuristic manner used to choose the components to share (according to some threshold) and the approximate nature of the method.

### 3.2 Platform selection by solving a relaxed combinatorial problem

This methodology integrates platform selection under bounds on performance deviation with optimal product family design (Fellini et al., 2002b). The designer can choose what performance deviations are acceptable relative to null-platform optima. Component sharing is determined through the solution of a relaxed commonality maximization combinatorial problem subject to these performance bounds.
The commonality decision problem is formulated as a mixed-discrete programming problem

\[
\max_{\eta, x = [x^p_1, x^p_2, ...]} \{ \{ f^p(x^p) \}, \sum_{(i,j) \in S^{pq}} \eta^p_{ij} \} \quad \forall \ p, q \in P, \ (i, j) \in S^{pq}, \ p < q \quad (6)
\]

subject to

\[
g^p(x^p) \leq 0
\]

\[
h^p(x^p) = 0
\]

\[
\eta^p_{ij} (x^p_i - x^q_j) = 0
\]

\[
\eta^p_{ij} \in \{0, 1\}.
\]

The set \(S^{pq}\) consists of component candidates for sharing between two products. The sharing decision variables \(\eta^p_{ij}\) multiplied by the commonality constraint can take a value of either 1 or 0 depending on whether a component is shared or not. The multiobjective problem formulation reflects the tradeoff between individual product performance and commonality.

Problem (6) is reformulated to include bounds on individual performance deviations and replace the sharing decision variables by a distance function \(D_\circ\):

\[
\max_{x = [x^p_1, x^p_2, ...]} \sum_{pq} |S^{pq}| - \sum_{(i,j) \in S^{pq}} D_\circ(x^p_i - x^q_j) \quad \forall \ p, q \in P, \ (i, j) \in S^{pq}, \ p < q \quad (7)
\]

subject to

\[
g^p(x^p) \leq 0
\]

\[
h^p(x^p) = 0
\]

\[
f^p(x^p) \geq (1 - L^p) f^{p,\circ},
\]

where

\[
D_\circ(x^p_i - x^q_j) = \begin{cases} 
0 & \text{if } x^p_i = x^q_j \\
1 & \text{otherwise}
\end{cases}
\]

(8)

We approximate the binary function \(D_\circ\) by the continuous function

\[
D_\alpha(x^p_i - x^q_j) = 1 - \frac{1}{\left(\frac{x^p_i - x^q_j}{\alpha}\right)^2 + 1}.
\]

(9)
This function is constructed as a measure of the distance between designs and approaches the function $D_\alpha$ as $\alpha$ goes to zero. Figure 3 shows $D_\alpha$ for $\alpha = 0.05$. Since $D_\alpha$ is continuously differentiable, gradient-based algorithms can be used to solve the approximate commonality problem.

Problem (7) can be solved in cases where components are described by a vector of design variables by computing the $l_2$ norm of the design difference

$$\min_{x} \sum_{(i,j) \in S_{pq}} D_\alpha(\| x^p_i - x^q_j \|_2) \quad \forall p, q \in \mathcal{P}, (i, j) \in S_{pq}, p < q.$$ 

Solution of Problem (7) will return just the “optimal” platform, but not necessarily the optimal design of the product family. Therefore the final step is to design the product family by solving Problem (2). The design process is visualized in Figure ??.

### 3.3 Combining the approaches

The motivation for an integrated approach is to take advantage of the strengths of each of the above two methods. The first approach will be used as a filtering step to reduce the problem size, acting as an upper bound computation in determining components considered good candidates for sharing. The second approach is then applied to the remaining candidates to complete the commonality selection. This reduced-size problem allows an efficient solution of the relaxed combinatorial problem. The combined methodology is then linked to an artificial neural network, due to computational expense, when using simulation-based models.

The first method requires a minor modification so that the two approaches can be integrated efficiently, specifically to allow components described as a vector of design variables. This is done by aggregating the performance deviation values of the design variables that
define the component into a single value, $\Pi^c$.

In the following demonstration study we will look at various ways to present the performance deviation vector. Previously, the sorted performance deviation vector values were plotted as was shown in Figure 2. We will also look at the increasing cumulative value of the performance deviation as more variables are shared.

The combined approach can be summarized in the following steps:

1. Determine the optimal null-platform design $x^{p,0}$ for each individual product $p \in P$ by solving the individual optimal design problem (Problem (1)).

2. Identify the components that are good candidates for sharing among products, i.e., define the candidate platform set.

3. Define the performance deviation $L^p$ acceptable for each of the products.

4. Compute the performance deviation vector using Eq. (4). Choose a subset of the candidate platform set as components to share.

5. Solve the relaxed combinatorial problem (Problem (7)) for the remaining components of the candidate platform set.

6. Based on the results obtained by solving the relaxed combinatorial problem, make a final selection of components to be shared.

7. Solve the family optimal design problem (Problem (2)).

In the following section we demonstrate the application of the combined strategy on a family of automotive engines.

4 Automotive engine family design

This case study will examine designing a family of engines using the combined strategy formulated above. Engine variants are defined based on different functional requirements.
4.1 Simulation tools and design variables

GT-Power by Gamma Technologies is used as the simulation tool (GTI, 2001). A 24-valve 2.5L V6 engine model, previously validated at various operating points, is used to generate the family. Analysis is performed at a specified operating point given engine speed and fuel rate, specifically at 5000 RPM and wide open throttle (WOT). The operating point characteristics are:

\[
\begin{align*}
N_e &= 5000 \text{ RPM (mean crank speed)}, \\
\theta_p &= 90^\circ \text{ (throttle angle)}, \\
i_{vo} &= 331.0^\circ \text{ (intake valve open wrt. CA (crank angle))}, \\
ivc &= -103.0^\circ \text{ (intake valve close wrt. CA (crank angle))}, \\
e_{vo} &= 101.0^\circ \text{ (exhaust valve open wrt. CA (crank angle))}, \\
e_{vc} &= 397.0^\circ \text{ (exhaust valve close wrt. CA (crank angle))}.
\end{align*}
\]
The geometry of components from the intake manifold through the exhaust system are modeled in the simulation. The design variables of particular interest in this study are:

\[ x_1 : \text{Bore (} b \text{)}, \]
\[ x_2 : \text{Stroke (} s \text{)}, \]
\[ x_3 : \text{Connecting rod length (} l \text{)}, \]
\[ x_4 : \text{Compression ratio (} c_r \text{)}, \]
\[ x_5 : \text{Intake valve diameter (} d_i \text{)}, \]
\[ x_6 : \text{Intake cam-timing angle (} i_{cta} \text{)}, \]
\[ x_7 : \text{Intake angle multiplier (} i_{am} \text{)}, \]
\[ x_8 : \text{Exhaust valve diameter (} d_e \text{)}, \]
\[ x_9 : \text{Exhaust cam-timing angle (} e_{cta} \text{)}, \]
\[ x_{10} : \text{Exhaust angle multiplier (} e_{am} \text{)}, \]
\[ i_1 : \text{Number of cylinders (} n_c \text{)}. \]

The two exhaust valves are modeled as a single valve by using the equivalent area

\[(\text{input diameter})^2 = 2(\text{valve diameter})^2.\] (10)

In addition, the theoretical height of the combustion chamber is computed as a function of the stroke and compression ratio

\[ h_c = s/(c_r - 1).\] (11)
The responses to be computed are:

\[ R_1 : \text{Brake Power (performance)}, \]
\[ R_2 : \text{Brake Torque (performance)}, \]
\[ R_3 : \text{BSFC (efficiency)}, \]
\[ R_4 : \text{NO}_x \text{ (emissions)}, \]
\[ R_5 : \text{dPmx/DCA (NVH,knock)}, \]
\[ R_6 : P_{\text{max}} \text{ (stress/durability)}. \]

The brake power and brake torque are measures of the product dynamic performance. The brake specific fuel consumption (BSFC) is a measure of efficiency, and measured emissions are through NO\(_x\) produced. Finally, dPmx/DCA, the mean pressure rise with respect to crank angle, and P\(_{\text{max}}\), the maximum cylinder pressure, contribute to various measures such as NVH, knock, stress, and engine durability.

The components we focus on are the following: exhaust cam(s), intake cam(s), exhaust valve(s), intake valve(s), cylinder head, piston(s), connecting rod, and engine block (c.f. Figure 5). Finally, we map the design variables to their respective components:

\[ x^{c_1} = [x_{10}], \]
\[ x^{c_2} = [x_7], \]
\[ x^{c_3} = [x_8], \]
\[ x^{c_4} = [x_5], \]
\[ x^{c_5} = [x_1, h_c, i_1], \]
\[ x^{c_6} = [x_1], \]
\[ x^{c_7} = [x_3], \]
\[ x^{c_8} = [x_1, x_2, x_3, i_1]. \]

Note that the number of cylinders, \(i_1\), will be fixed at six for all engines.
Simulation is computationally quite expensive. Therefore, we developed a surrogate model by training a radial basis function artificial neural network. This model was constructed using a 2500-point data sample from a Latin-hypercube design of experiments. The upper and lower bounds of the design variables were set as follows:

\[
\begin{align*}
 x_{1,l}, x_{1,u} &= 70.0, 95.0 \text{ [mm]}, \\
 x_{2,l}, x_{2,u} &= 70.0, 95.0 \text{ [mm]}, \\
 x_{3,l}, x_{3,u} &= 105.0, 237.5 \text{ [mm]}, \\
 x_{4,l}, x_{4,u} &= 8.0, 10.0, \\
 x_{5,l}, x_{5,u} &= 22.0, 35.0 \text{ [mm]}, \\
 x_{6,l}, x_{6,u} &= -10.0^\circ, 10.0^\circ, \\
 x_{7,l}, x_{7,u} &= 0.9, 1.1, \\
 x_{8,l}, x_{8,u} &= 30.0, 43.0 \text{ [mm]}, \\
 x_{9,l}, x_{9,u} &= -10.0^\circ, 10.0^\circ, \\
 x_{10,l}, x_{10,u} &= 0.9, 1.1.
\end{align*}
\]

A 500-point data sample of random design points was used to validate the model. The computed average errors $\mu$ and standard deviations $\sigma$ are shown in Table 1. All reported results have been obtained using the artificial neural network in place of the GT-power model simulation.

### 4.2 Optimal design model

The first step in the design process is to define the optimal design model. Various engine design rules of thumb on bore to stroke ratio, connecting rod to stroke ratio, etc. are available in the literature (Heywood, 1988). In addition to geometric constraints, limits are placed on pressure gradients, in-cylinder pressures, and mean piston velocities to maintain the reliability of the engine. The following inequality constraints must be satisfied by all
family products:

\[ g_1^p, g_2^p : 0.8 \leq b/s \leq 1.2, \]
\[ g_3^p, g_4^p : 350 \leq \pi b^2 s/4 \times 10^{-3} \leq 650 \text{ [cm}^3\text{]}, \]
\[ g_5^p, g_6^p : 1.5 \leq l/s \leq 2.5, \]
\[ g_7^p : d_i \leq 0.37 b \text{ [mm]}, \]
\[ g_8^p : d_e \leq 0.45 b \text{ [mm]}, \]
\[ g_9^p : (2 s N_e)/(60 \cdot 1000) \leq 15.0 \text{ [m/s]}, \]
\[ g_{10}^p : s/(c_r - 1) \geq 5.0 \text{ [mm]}, \]
\[ g_{11}^p : l + s + s/(c_r - 1) + 0.5 b \leq 350.0 \text{ [mm]}, \]
\[ g_{12}^p : \text{dPmx/DCA} \leq 3.0 \text{ [bar/deg]}, \]
\[ g_{13}^p : P_{max} \leq 110 \text{ [bar]}, \]

where

\[ g_1^p, g_2^p : \text{Bounds on bore-to-stroke ratio}, \]
\[ g_3^p, g_4^p : \text{Bounds on displacement of cylinder}, \]
\[ g_5^p, g_6^p : \text{Bounds on connecting rod length-to-stroke ratio}, \]
\[ g_7^p : \text{Upper bound on intake valve diameter with respect to bore}, \]
\[ g_8^p : \text{Upper bound on exhaust valve diameter with respect to bore}, \]
\[ g_9^p : \text{Upper bound on mean piston speed}, \]
\[ g_{10}^p : \text{Upper bound on clearance height above piston crown}, \]
\[ g_{11}^p : \text{Upper bound on overall engine height}, \]
\[ g_{12}^p : \text{Upper bound on pressure rise rate}, \]
\[ g_{13}^p : \text{Upper bound on cylinder pressure}. \]

We define three variants by means of three functional requirements. The first engine variant is designed to maximize power, the second to minimize fuel consumption, and the third to
minimize emissions. The optimal design problems are formulated as

\[
\max_{x^p} \quad f^p = \text{Power [kW]} \tag{13}
\]

subject to

\[
g_1^p, g_2^p, \ldots, g_{13}^p \\
g_{14}^p : \text{NO}_x \leq \text{NO}_{x,\text{max}} \text{ [ppm]} \\
g_{15}^p : \text{Power} \cdot \text{BSFC} \leq 30,000 \text{ [g/h]}
\]

\[
\min_{x^p} \quad f^p = \text{Power} \cdot \text{BSFC [g/h]} \tag{14}
\]

subject to

\[
g_1^p, g_2^p, \ldots, g_{13}^p \\
g_{14}^p : \text{NO}_x \leq \text{NO}_{x,\text{max}} \text{ [ppm]} \\
g_{15}^p : \text{Power} \geq 80 \text{ [kW]}
\]

\[
\min_{x^p} \quad f^p = \text{NO}_x \text{ [ppm]} \tag{15}
\]

subject to

\[
g_1^p, g_2^p, \ldots, g_{13}^p \\
g_{14}^p : \text{Power} \geq 80 \text{ [kW]} \\
g_{15}^p : \text{Power} \cdot \text{BSFC} \leq 30,000 \text{ [g/h]}
\]

The value of \(\text{NO}_{x,\text{max}}\) is based on the baseline value of 25,546 ppm multiplied by 110%, namely, we do not want to produce more than 10% additional emissions with respect to the baseline model.

5 Results and discussion

Following the steps of the combined methodology we first solve Problem (1) to obtain the null-platform optima for each of the three products independently. The results are shown in Table 2. For the engine designed for maximizing power we find that three of the inequality constraints are active. These are the upper bounds on the mean piston speed, engine height, and \(\text{NO}_x\) emissions. For the engine designed for maximizing fuel efficiency four
inequality constraints are active. These are the lower bounds on cylinder displacement and power, and the upper bounds on connecting rod to stroke ratio and intake valve diameter with respect to bore. For the engine designed for minimizing emissions three inequality constraints are active. These are the upper bounds on the connecting rod to to stroke ratio and intake valve diameter with respect to the bore, along with the lower bound on power. For the maximum power and fuel efficiency engines the compression ratio is maximized, which in turn increases combustion efficiency. For the low emissions engine the compression ratio is minimized, which corresponds to the strong correlation of NO\textsubscript{x} production with the increased heat due to higher compression. There is relatively little natural commonality between the three individually optimized engines. The only component that is common is the exhaust valve between the engine designed for fuel efficiency and low emissions.

We computed the performance deviation vector for the product family. All functions are normalized and design variables are scaled to be in the range [0,1]. Figure 6 depicts the sorted performance deviation vector in terms of both individual Π\textsubscript{i} and cumulative values. Figure 7 illustrates the sorted performance deviation vector with respect to engine components (aggregating deviations that correspond to component variables) in terms of both individual Π\textsubscript{c} and cumulative values. The performance deviation vector values sorted with respect to the design variable illustrate which variables we could share if variable sharing were the objective. Our focus is on the deviations sorted with respect to components. From the performance deviation vector values sorted with respect to components, we find that the Π\textsubscript{c} values are relatively low for the connecting rod, intake and exhaust valves, and the intake cam. We also observe on the cumulative deviation plot that after sharing the first four components (components 7, 4, 2, and 3, the performance deviation increases significantly. Therefore, these first four components will be shared and the relaxed combinatorial problem will be solved for the remaining components.

In the second step we solved the relaxed combinatorial optimization problem for a deviation factor of 5%, 6%, 7%, 8%, 9%, and 10%. The most interesting results were
obtained for 6% and 7%, and are presented in Table 3, where 0 and 1 denote no sharing and sharing in product pairs, respectively. Note that if a component is shared in all product pairs, then it is shared among all products. The results indicate that we can share the intake and exhaust valves along with the connecting rod and intake cam across all engine variants for both performance deviation bounds. Depending on the allowable deviations from the optimal designs it is also possible to share the exhaust cam across the family. The piston is consistently shareable only between the high power and low emissions engines. With slightly more performance deviation it is also possible to share the entire engine block between these two engines. Note that the cylinder head, piston, and engine block are “modules” that happen to be shared in two cases (the piston and engine block between products A and C). Exchanging these components produces the variety in this engine family.

The final step is to design the product family. We design each of the engines to minimize the performance deviation from the null-platform optima. Therefore, the objective function

$$\min_{x = [x^{p1}, x^{p2}, ...]} \left\{ \left( \frac{(f^{p^\circ} - f^p(x^p))}{f^{p^\circ}} \right)^2 \right\}$$

is used when solving Problem (2). The optimal family designs for the 6% and 7% deviation bound cases are presented in Tables 4 and 5, respectively. Performance optima and associated deviations are summarized in Table 6; they demonstrate that the bounds on performance deviation due to commonality are not violated. The results have been validated by solving the entire problem using the relaxed-problem formulation. The optimization results are consistent to the combined approach. Additionally, the sharing order that is computed by the first-order method is confirmed.

## 6 Conclusions

A methodology was proposed for making commonality decisions and designing optimal product families, by combining two previously developed approaches. The methodology
was applied successfully to the design of an automotive engine family consisting of three variants with different functional requirements modeled by means of artificial neural networks. The advantage of the new method is improved computationally efficiency. This allows for commonality decisions on larger product sets. The limitations include the heuristic nature still inherent to the first-order method. However, the consequence of this limitation has been reduced through the combination of the strategies. Future work for the methodology could include adding search heuristics in an attempt to further increase efficiency of the utilized algorithms. The mixed-discrete case could be explored by implementing evolutionary algorithms together with the continuous algorithm. Future work for the engine application includes looking at engine variants defined by different power specifications and using the number of cylinders as a design variable. An important next step is to integrate the engine model into a full vehicle model to obtain a more comprehensive operating cycle. From a methodology viewpoint, the next step is to link the engineering decisions with financial analysis models (Georgiopoulos et al., 2002) so that product families can be examined in the context of a combined business-engineering product development and assessment process.

Acknowledgments

The authors would like to thank Nestor Michelena and Alexis Perez-Duarte for their numerous helpful suggestions and discussions regarding the commonality strategy, and George Delagrammatikas, Terry Wagner, and Guangquan Wu for their assistance with the engine simulations and design model. This research was partially supported by the Automotive Research Center, a US Army Center of Excellence in Modeling and Simulation of Ground Vehicles, by a US Army Dual-Use Science and Technology Project, and by the General Motors Collaborative Research Laboratory at the University of Michigan. This support is gratefully acknowledged.
References


Ryan Fellini is a Post-Doctoral Research Fellow in the Department of Mechanical Engineering at the University of Michigan. He has received a B.S. from Virginia Tech (1996) and a M.S. and a Ph.D. from the University of Michigan at Ann Arbor (1998, 2003). His research interests include design of product families, optimal design of large systems, and distributed design environments.

Michael Kokkolaras is an Assistant Research Scientist in the Department of Mechanical Engineering at the University of Michigan. He received a Diploma in Aerospace Engineering from the Munich University of Technology (1992) and a Ph.D. in Mechanical Engineering from Rice University (1998). Besides product families, his interests include multidisciplinary optimization and design under uncertainty.

Panos Y. Papalambros is a Professor of Mechanical Engineering at the University of Michigan. He received a Diploma in Mechanical and Electrical Engineering from the National Technical University of Athens (1974) and a M.S. and a Ph.D. from Stanford University (1976, 1979). His interests include design methodology, optimization and systems integration, and ecologically conscious design. He has published over 140 papers and is co-author of the textbook Principles of Optimal Design: Modeling and Computation (Cambridge University Press, New York, New York, 1988, 2000).
Table 1: Artificial neural network model average errors and standard deviations

<table>
<thead>
<tr>
<th>Response</th>
<th>$\mu$</th>
<th>$\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_1$</td>
<td>6.5521</td>
<td>5.6571</td>
</tr>
<tr>
<td>$R_2$</td>
<td>6.5520</td>
<td>5.6570</td>
</tr>
<tr>
<td>$R_3$</td>
<td>1.2758</td>
<td>1.3417</td>
</tr>
<tr>
<td>$R_4$</td>
<td>1.7756</td>
<td>3.4038</td>
</tr>
<tr>
<td>$R_5$</td>
<td>4.3606</td>
<td>4.3559</td>
</tr>
<tr>
<td>$R_6$</td>
<td>3.5425</td>
<td>3.8723</td>
</tr>
</tbody>
</table>
Table 2: Null-platform optima

<table>
<thead>
<tr>
<th>Engine:</th>
<th>Power (A)</th>
<th>Fuel Usage (B)</th>
<th>Emissions (C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>84.43</td>
<td>75.39</td>
<td>95.00</td>
</tr>
<tr>
<td>s</td>
<td>79.95</td>
<td>78.40</td>
<td>90.00</td>
</tr>
<tr>
<td>l</td>
<td>199.87</td>
<td>196.00</td>
<td>201.02</td>
</tr>
<tr>
<td>c_r</td>
<td>10.00</td>
<td>10.00</td>
<td>8.84</td>
</tr>
<tr>
<td>d_i</td>
<td>31.24</td>
<td>27.90</td>
<td>27.55</td>
</tr>
<tr>
<td>i_cta</td>
<td>-10.00</td>
<td>10.00</td>
<td>-10.00</td>
</tr>
<tr>
<td>i_am</td>
<td>0.99</td>
<td>1.08</td>
<td>1.03</td>
</tr>
<tr>
<td>d_e</td>
<td>35.57</td>
<td>30.00</td>
<td>30.00</td>
</tr>
<tr>
<td>e_cta</td>
<td>-10.00</td>
<td>10.00</td>
<td>-10.00</td>
</tr>
<tr>
<td>e_am</td>
<td>1.10</td>
<td>1.00</td>
<td>0.90</td>
</tr>
<tr>
<td>h_c</td>
<td>8.88</td>
<td>8.71</td>
<td>11.48</td>
</tr>
<tr>
<td>disp.</td>
<td>2686</td>
<td>2100</td>
<td>3828</td>
</tr>
<tr>
<td>Power</td>
<td>114.29</td>
<td>80.00</td>
<td>80.00</td>
</tr>
<tr>
<td>Torque</td>
<td>218.27</td>
<td>152.79</td>
<td>152.79</td>
</tr>
<tr>
<td>BSFC</td>
<td>263.50</td>
<td>265.82</td>
<td>331.17</td>
</tr>
<tr>
<td>NO_x</td>
<td>26089.02</td>
<td>25717.78</td>
<td>17853.09</td>
</tr>
<tr>
<td>dPmx/DCA</td>
<td>1.40</td>
<td>1.28</td>
<td>0.88</td>
</tr>
<tr>
<td>P_max</td>
<td>52.36</td>
<td>47.97</td>
<td>35.43</td>
</tr>
<tr>
<td>Fuel Usage</td>
<td>30000.00</td>
<td><strong>21265.32</strong></td>
<td>26493.90</td>
</tr>
</tbody>
</table>
Table 3: Relaxed combinatorial optimization problem results (0 and 1 denote no sharing and sharing, respectively)

<table>
<thead>
<tr>
<th>Shared between:</th>
<th>Deviation = 6%</th>
<th>Deviation = 7%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exhaust Cam</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Intake Cam</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Exhaust Valve</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Intake Valve</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cylinder Head</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Piston</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Connecting Rod</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Engine Block</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Table 4: Product family designs for 6% deviation

<table>
<thead>
<tr>
<th>Engine:</th>
<th>Power (A)</th>
<th>Fuel Usage (B)</th>
<th>Emissions (C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>95.00</td>
<td>75.03</td>
<td>95.00</td>
</tr>
<tr>
<td>s</td>
<td>79.17</td>
<td>79.17</td>
<td>90.00</td>
</tr>
<tr>
<td>l</td>
<td>197.92</td>
<td>197.92</td>
<td>197.92</td>
</tr>
<tr>
<td>c&lt;sub&gt;r&lt;/sub&gt;</td>
<td>10.00</td>
<td>10.00</td>
<td>8.69</td>
</tr>
<tr>
<td>d&lt;sub&gt;i&lt;/sub&gt;</td>
<td>26.65</td>
<td>26.65</td>
<td>26.65</td>
</tr>
<tr>
<td>i&lt;sub&gt;cta&lt;/sub&gt;</td>
<td>-5.32</td>
<td>10.00</td>
<td>-10.00</td>
</tr>
<tr>
<td>i&lt;sub&gt;am&lt;/sub&gt;</td>
<td>1.05</td>
<td>1.05</td>
<td>1.05</td>
</tr>
<tr>
<td>d&lt;sub&gt;e&lt;/sub&gt;</td>
<td>30.00</td>
<td>30.00</td>
<td>30.00</td>
</tr>
<tr>
<td>e&lt;sub&gt;cta&lt;/sub&gt;</td>
<td>10.00</td>
<td>10.00</td>
<td>-10.00</td>
</tr>
<tr>
<td>e&lt;sub&gt;am&lt;/sub&gt;</td>
<td>0.99</td>
<td>0.90</td>
<td>0.90</td>
</tr>
<tr>
<td>h&lt;sub&gt;c&lt;/sub&gt;</td>
<td>8.80</td>
<td>8.80</td>
<td>11.71</td>
</tr>
<tr>
<td>disp.</td>
<td>3367</td>
<td>2100</td>
<td>3828</td>
</tr>
<tr>
<td>Power</td>
<td>109.62</td>
<td>80.54</td>
<td>80.00</td>
</tr>
<tr>
<td>Torque</td>
<td>209.36</td>
<td>153.81</td>
<td>152.79</td>
</tr>
<tr>
<td>BSFC</td>
<td>273.67</td>
<td>269.50</td>
<td>330.72</td>
</tr>
<tr>
<td>NO&lt;sub&gt;x&lt;/sub&gt;</td>
<td>25795.82</td>
<td>25252.87</td>
<td>18110.20</td>
</tr>
<tr>
<td>dPmx/DCA</td>
<td>1.29</td>
<td>1.26</td>
<td>0.87</td>
</tr>
<tr>
<td>P&lt;sub&gt;max&lt;/sub&gt;</td>
<td>48.86</td>
<td>47.29</td>
<td>35.22</td>
</tr>
<tr>
<td>Fuel Usage</td>
<td>30000.00</td>
<td>21704.10</td>
<td>26457.97</td>
</tr>
</tbody>
</table>
Table 5: Product family designs for 7% deviation

<table>
<thead>
<tr>
<th>Engine:</th>
<th>Power (A)</th>
<th>Fuel Usage (B)</th>
<th>Emissions (C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>91.02</td>
<td>74.13</td>
<td>91.02</td>
</tr>
<tr>
<td>s</td>
<td>90.00</td>
<td>81.09</td>
<td>90.00</td>
</tr>
<tr>
<td>l</td>
<td>202.72</td>
<td>202.72</td>
<td>202.72</td>
</tr>
<tr>
<td>c_r</td>
<td>10.00</td>
<td>10.00</td>
<td>8.64</td>
</tr>
<tr>
<td>d_i</td>
<td>27.43</td>
<td>27.43</td>
<td>27.43</td>
</tr>
<tr>
<td>i_cTA</td>
<td>3.42</td>
<td>10.00</td>
<td>-10.00</td>
</tr>
<tr>
<td>i_am</td>
<td>1.02</td>
<td>1.02</td>
<td>1.02</td>
</tr>
<tr>
<td>d_e</td>
<td>30.00</td>
<td>30.00</td>
<td>30.00</td>
</tr>
<tr>
<td>e_cTA</td>
<td>10.00</td>
<td>10.00</td>
<td>-10.00</td>
</tr>
<tr>
<td>e_am</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>h_c</td>
<td>10.00</td>
<td>9.01</td>
<td>11.78</td>
</tr>
<tr>
<td>disp.</td>
<td>3513</td>
<td>2100</td>
<td>3513</td>
</tr>
<tr>
<td>Power</td>
<td>106.29</td>
<td>84.39</td>
<td>80.28</td>
</tr>
<tr>
<td>Torque</td>
<td>203.00</td>
<td>161.16</td>
<td>153.31</td>
</tr>
<tr>
<td>BSFC</td>
<td>282.25</td>
<td>268.83</td>
<td>324.71</td>
</tr>
<tr>
<td>NO_x</td>
<td>24638.78</td>
<td>25237.59</td>
<td>18883.23</td>
</tr>
<tr>
<td>dPmx/DCA</td>
<td>1.22</td>
<td>1.28</td>
<td>0.89</td>
</tr>
<tr>
<td>P_max</td>
<td>46.20</td>
<td>48.15</td>
<td>35.86</td>
</tr>
<tr>
<td>Fuel Usage</td>
<td>30000.00</td>
<td>22685.43</td>
<td>26066.62</td>
</tr>
<tr>
<td>Variant</td>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>Null platform</td>
<td>114.29 kW</td>
<td>21,265.32 g/h</td>
<td>17,853.09 ppm</td>
</tr>
<tr>
<td>Platform with $L_p = 6%$</td>
<td>109.62 kW</td>
<td>21,704.10 g/h</td>
<td>18,110.20 ppm</td>
</tr>
<tr>
<td>Performance loss</td>
<td>4.09%</td>
<td>2.06%</td>
<td>1.44%</td>
</tr>
<tr>
<td>Platform with $L_p = 7%$</td>
<td>106.29 kW</td>
<td>22,685.43 g/h</td>
<td>18,883.23 ppm</td>
</tr>
<tr>
<td>Performance loss</td>
<td>7.00%</td>
<td>6.68%</td>
<td>5.77%</td>
</tr>
</tbody>
</table>
Figure 1: Performance deviations due to commonality
Figure 2: Sorted performance deviation vector (reproduced from (Fellini et al., 2002a))
Figure 3: Continuous approximation of binary function (reproduced from (Fellini et al., 2002b))
Figure 4: Product family design process (reproduced from (Fellini et al., 2002b))
Identify Candidates for Sharing
Component sharing decisions, and assigning sharing preference

Figure 5: Engine components of interest

\[ c_1 = \text{Exhaust Cam} \ [e_{am}] \]
\[ c_2 = \text{Intake Cam} \ [i_{am}] \]
\[ c_3 = \text{Exhaust Valve} \ [d_e] \]
\[ c_4 = \text{Intake Valve} \ [d_i] \]
\[ c_5 = \text{Cylinder Head} \ [b, h_c, n_c] \]
\[ c_6 = \text{Piston} \ [b] \]
\[ c_7 = \text{Connecting Rod} \ [l] \]
\[ c_8 = \text{Engine Block} \ [b, s, l, n_c] \]
Figure 6: Sorted performance deviation vector with respect to design variables (the plot on the top shows the sorted values of $\Pi_i$, while the plot on the bottom shows cumulative values of the deviation vector)
Figure 7: Performance deviation vector with respect to engine components (the plot on the top shows the sorted values of $\Pi^c_i$, while the plot on the bottom shows cumulative values of the deviation vector.