FUNCTIONAL TRADE-OFFS IN THE MECHANICAL DESIGN OF INTEGRATED PRODUCTS - IMPACT ON ROBUSTNESS AND OPTIMISABILITY

Sigurdarson, Nökkvi S. (1,2); Eifler, Tobias (1); Ebro, Martin (2)

1: Technical University of Denmark (DTU); 2: Novo Nordisk A/S

ABSTRACT

It is generally accepted in industry and academia that trade-offs between functional design objectives are an inevitable factor in the development of mechanical systems. These trade-offs can have a large influence on the achievable robustness and performance of the final design, with many products only functioning in narrow sweet-spots between different objectives. As a result, the design process of multi-functional products can be prolonged when designers concurrently attempt to find sweet-spots between a number of potentially interdependent trade-offs.This paper will show that designers only have six different approaches available when attempting to manage a trade-off while trying to ensure robustness and a sufficient performance. These fall within one of three categories; accept, optimise, or redesign. Selecting the wrong approach, can result in consequences downstream which can be difficult to predict, amongst others a lack of robustness to geometric variation, constrained performance, and long development lead time.This points to a substantial potential in the synthesis of design methods that support the identification and management of trade-offs in early product development.

Keywords: Robust design, Embodiment design, Optimisation, Design trade-offs

Contact:
Sigurdarson, Nökkvi S,
Danish Technical University (DTU)
Department of Mechanical Engineering
Denmark
noksig@mek.dtu.dk

Cite this article: Sigurdarson, N.S., Eifler, T., Ebro, M. (2019) 'Functional Trade-offs in the Mechanical Design of Integrated Products - Impact on Robustness and Optimisability', in *Proceedings of the 22nd International Conference on Engineering Design (ICED19),* Delft, The Netherlands, 5-8 August 2019. DOI:10.1017/dsi.2019.356

1 INTRODUCTION

In mechanical engineering design it is generally accepted that trade-offs between design-objectives will inevitably need to be made in the development process. Most systems are integrated, meaning that their individual components and subsystems each contribute to several functions and requirements, and often do so simultaneously. This tendency, which seems to be growing, is driven by two factors:

1. Competitive pressure - the success of new products is generally reliant on increased functionality and/or improved performance, and product developers will therefore commonly strive to add new features and sub-functions that bring added value to the user or optimise the existing design.
2. Associated cost - production companies aim for driving as much functionality with as few components as possible, in order to ensure cost efficient and reliable products.

The result is products that are becoming increasingly complex to develop (Arthur, 1993), given a growing amount of constraints and interdependent design objectives. No product can be infinitely accurate, durable, efficient, robust, user friendly, manufacturable, cost effective, etc. In order to find the correct balance between the many objectives a product is designed toward, numerous trade-offs need to be either solved or managed systematically throughout the design process. As a consequence, product development can be a highly iterative process, where the design is gradually refined until the required functionality has been achieved and an acceptable compromise between all design objectives is reached. The aim of this contribution is to show that there is a lack of a comprehensive way for a design engineer, faced with a design trade-off, to decide between *accepting* the trade-off, *optimising* the design, or making a *design change* to avoid the trade-off. As a consequence, realising the design may require tighter tolerances and lesser performance than originally foreseen in the preliminary phases, with a design process that is correspondingly more prone to loopbacks that lead to delays.

2 THEORETICAL BACKGROUND

The development of new products often follows a structured process (Pahl & Beitz, 1996) with a defined set of phases, each becoming more specific and detailed until the product is complete. Decisions made early in this process are decisive for the downstream workload. The process of geometrically realising the functional intent and layout of the final product - often coined embodiment design - determines the limits of its achievable performance (Papalambros & Wilde, 2017), and robustness (Andersson, 1997). Yet as discussed by Andreasen & Howard (2012), this process is ill supported by existing methodology. As such, it seems that most late design changes and development lead-time can be traced back to decisions made during embodiment (Vianello *et al*., 2012).

Most optimisation and robust design methods are highly dependent on knowledge that is not necessarily available at an early stage of development and are therefore often applied at least after a preliminary embodiment has been developed (Papalambros & Wilde, 2017), (Ebro & Howard, 2016). As a result, a lot of existing design methodology aimed at the embodiment phase is based on heuristics e.g. (French, 1971), (Pahl & Beitz, 1996), (Matthiassen, 1996), (Anderson, 1997), and DfX (Olesen, 1992). Yet the challenge in most of these is that their application is mostly limited to single functions, single domains e.g. structural design, hydraulics, and single dispositions, (c.f. DfX). In the context of the embodiment of mechanical systems, there are however several domains (especially in multi-physical design situations) and dispositions to take into account, and as Mattiassen (1997) remarked, no universal design principle can meet all design requirements. This is then further complicated when the designer has to realise multiple functions and sub-functions in the same system.

Few specific embodiment heuristics exist within the question of functional integration, perhaps due to the infinite amount of combinations of working principles and design requirements that could feasibly exist in the same system. Yet the importance of allocation of functionality amongst the components in a system is covered by numerous sources from different perspectives, e.g. division of tasks (Pahl and Beitz, 1996), integration and differentiation (Matthiassen, 1997), merging and segmentation (Altshüller, 1984). However, none of these answer the question of how or when to integrate functionality into fewer components, and when to differentiate. Yet from a production perspective, integration can have significant benefits; cf. design for manufacture and assembly (Boothroyd, 2002).

Functional allocation in a system can have a significant detrimental impact, if the wrong functions are integrated into the same subsystems. Integration implicitly involves designing geometrical features or system properties that contribute to multiple design objectives - either simultaneously or in different

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

functional states. Increasing integration in other words creates dependencies. A natural consequence of integration in design, dependencies can both have a positive and negative impact on a design:

- Positive in the sense that they allow more functionality to be fulfilled without increasing the amount of sub-systems, parts, or even geometrical features.
- Negative in situations where the dependencies leads to a design trade-off, where two or more objectives have conflicting relationships to same geometry

Herein lies the reason as to why no universal design principle exists to meet all design requirements; as argued Pahl and Beitz (1996), it can be difficult or impossible to optimise the "carrier of several combined functions". Implicitly, multifunctional products will always require trade-offs to be made. Given that design principles are generally made with few objectives in mind, following these otherwise useful recommendations will inevitably come with a cost in form of a trade-off with other objectives within the system. Take the commonly cited "*Provide short direct force paths*" principle (French, 1971/Pahl & Beitz, 1996). While it definitely has benefits from a structural design perspective, it directly contradicts Matthiassens' (1997) principle *integrate for coordinated outputs* which aims at ensuring accuracy and coordination between subsystems in machine by using a shared power input, which of course results in longer force paths than otherwise. As a result, designers are left to rely on experience and intuition to create good designs, as shown by Ahmed *et al*. (2003), who found that experienced designers are much more likely to be aware of trade-offs.

Several existing frameworks address dependencies between requirements in design. All involve some form of dependency identification and assessment, with some aimed at supporting synthesis that aims at reducing the negative impacts of dependency. Dependency modelling methods are widely applied in design. An example of such, is the design structure matrix (DSM), which is used to qualitatively identify dependencies in complex system development, addressing aspects such as modularity, task coordination, system optimisation, and system integration (Eppinger & Browning, 2012).

More quantitative approaches to trade-offs are however widespread in the context of optimization, decision support and design space exploration. Examples include hybrid trade-off strategies (Otto *et al*., 1991), the compromise decision support (Mistree *et al*., 1993), and trade-space exploration (Ross *et al*., 2004). These however all aim at finding the best solution in a system influenced by trade-offs, rather than changing the system itself. The field of optimal design is aimed at identifying the best combination of parameter values given a set of objectives, constraints, and dependencies. Dependency mapping - for instance functional dependence trees (Wagner, 1993) - is used for model partitioning, and decomposition in optimisation, in an effort to simplify model definition and reduce computational cost. Optimisation often involves finding the best trade-off between two or more objectives within a set of constraints. Yet at the same time, the optimisation field does not necessarily question whether the design itself is worth optimising - whether it is sufficiently optimisable, or whether the objectives involved are so conflicting and hence limiting, that alternative embodiments might be worth exploring.

Methods related to design dependency aimed at the support of synthesis also exist. An example of such, is Axiomatic Design (AD) (Suh, 2001), which states that any form of dependency - termed coupling - is detrimental to how well a system will function, and should therefore be avoided or reduced. Another example is TRIZ (Altshuller, 1984), a knowledge-based inventive problem solving framework, which aims at supporting invention through identification and removal of so-called contradictions, which are incompatibilities between design objectives.

Interestingly, Ahmed *et al*. (2003), found that strategies to identifying or avoiding design trade-offs is largely based on tacit knowledge in industrial practice. This surely creates additional challenges in original design, given that experience is less useful when designers are met with design tasks and issues they have not faced before. When designing a multi-functional product, how does a designer then ensure that the decisions made during the embodiment phase, do not result in trade-off situations at a late stage of development, where the only recourse is constrained optimisation or acceptance? If the product could be embodied in such a way, that the aforementioned situation is avoided, then the development lead time for new complex products, and the importance of experience would be reduced. As discussed, heuristics are not necessarily sufficient in helping designers allocate functionality across a system, in a way that secures robustness and high performance. Meanwhile, the applicability of methods related to dependency can be limited in mechanical design given that:

1. Dependency modelling methods such as DSM, are qualitative and knowledge intensive in nature. Requiring a substantial insight into the workings of a design, DSM would also result in a substantial workload if one were to aim at identifying all the dependencies in a system, across

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

all levels of abstraction. While these methods are certainly valuable in the decomposition of complex systems, and the identification of dependencies, they do not necessarily aid the designer in then determining the impact of the dependency, and what to do about it.

2. Design optimisation methods and quantitative trade-off studies are mostly applied after a concept and embodiment has been defined, generally only adjusting the parameter design within the limits of the embodiment itself. Driving design and redesign activities with these is as such relatively time consuming, as it implies an iterative, knowledge intensive approach.
3. According to the first axiom of AD, the ideal design would be fully differentiated, meaning that each functional requirement is met by a unique set of design parameters. This implies little to no integration, and would therefore in practice often require a larger number of components to fulfil. This in turn increases the information content, in conflict with the second axiom. Similarly, Frey *et al*. (2007) found that increased coupling is not necessarily detrimental, when studying the relationship between part count and performance. What AD as such fails to capture, is that couplings can in fact have a positive effect on performance and robustness.
4. Being aimed at invention rather than mechanical design specifically, TRIZ is of a general nature, spanning any type of contradiction in any product. It is also fairly limited in its uptake in practice, often attributed to its' perceived enigmatic nature and complexity (Ilevbare *et al*., 2013). Frey *et al.* (2007) also did not find the *law of ideality* to be consistent with their observations from practice.

Given some of these challenges, Göhler and Howard (2015) put forward a metric, the contradiction index, introducing the notion that not all couplings need to have a negative impact. Their work combines AD, the notion of contradictions with system complexity considerations from DSM. The metric was meant as an indicator of robustness to be applied at an early stage of development- Göhler *et al*. (2016) later found a significant correlation between this index and system robustness. This begs the question; how can designers avoid distributing functionality in a way that results trade-offs?

3 TRADE-OFFS IN ROBUST MECHANICAL DESIGN

Achieving a design where all objectives are independent is unrealistic in practice. Design objectives will by interdependent be it due to design integration caused by e.g. manufacturing constraints, or simply due to inherent dependencies between physical phenomena. With this in mind, it seems that the understanding of how trade-offs between functional objectives arise in mechanical design, and how these can be managed, is essential to the embodiment of robust multi-functional products. In the following, a theoretical perspective is given on the reason behind the occurrence of robustness and performance reducing trade-offs, and what options are available to designers in such situations.

3.1 Occurrence and implications of trade-offs in mechanical design

As introduced by Göhler & Howard (2015), functional requirements from AD can be split in three categories; *min-is-best, nominal-is-best, and max-is-best*. This notion is also consistent with optimal design, where objective functions are aimed at minimisation, maximisation, while meeting a set of constraints. Conversely, when looking at the relationship between two coupled functional requirements, they concluded that there are three types of coupling, positive, negative, and nominal. Negative couplings always cause a trade-off between the two requirements, which will often result in a narrow design space, which in turn reduces both the achievable performance and the allowable variation, if both targets are to be met, as illustrated in figure 1. Negative couplings in other words reduce the feasible domain of a design, with the bound" stemming from the coupling rather than a constraint.

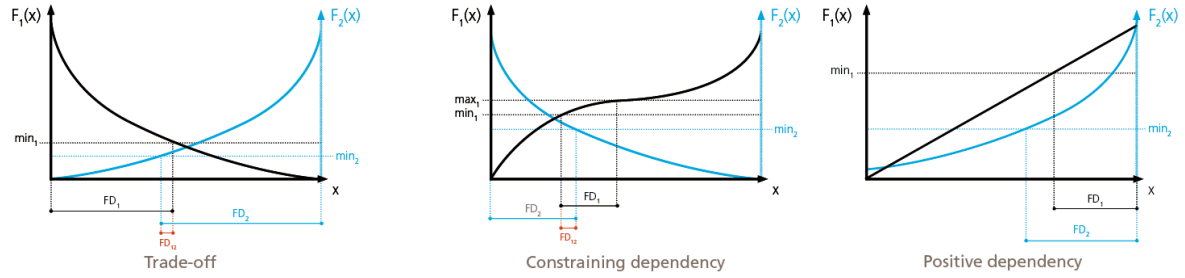

Figure 1 - The types of dependency between two objective functions F1 and F2, with a shared design parameter, x. Adapted from Göhler & Howard (2015)

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

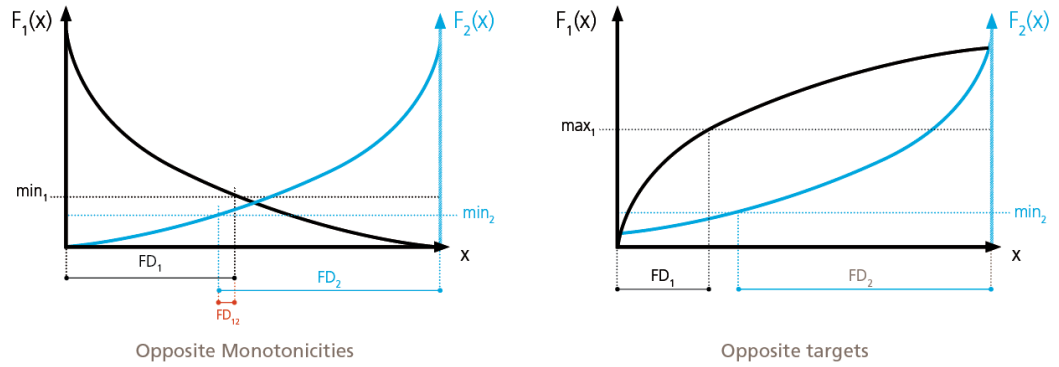

Figure 2 - The two types of functional trade-offs, Left: A narrow feasibility domain (FD12) requiring the design parameter x to have tight tolerance in order to stay inside a "sweet-spot". Right: An unfeasible design where no value of x exists where both objectives are met

Building on these notions, design trade-offs between two objective-functions must only be able arise in two situations. For two mutually dependant objectives F_1 & F_2 shown in fig 2, a trade-off occurs if:

1. F1 & F2 are of the same objective type (e.g. min-is-best vs. min is best), but are oppositely monotonic, either globally or locally within the feasibility domains of each objective.
2. F1 & F2 are of different objective type (e.g. max vs. min) but have the same monotonicities

What is more; these situations can occur in decoupled designs, if the order of the influence of the dependant parameter is sufficiently high, or if the independent parameters cannot be adjusted without violating other constraints. A wide range of analysis tools could be used to identify design trade-offs at a relatively early stage of design, e.g. through the use of monotonicity tables (Papalambros & Wilde, 1979) or using the DSM-based contradiction-index approach (Göhler & Howard, 2015).

The implication for mechanical design, is that systems that constrain themselves due to conflicting objective functions, must have a smaller feasible domain, than designs that are merely bound by geometry constraints. In other words, better optima must be achievable in a design without a trade-off, than in a design with one. Designs with positive dependencies also allow optima that can be achieved with lessened need for to tight parameter control (as the feasibility domain is wider), meaning that the design is more robust to variation. This is not to imply that multi-objective performance trade-offs are the same as the oft discussed and omnipresent performance vs. quality trade-offs, but rather that designs with performance trade-offs are more prone to being sensitive to variation, and will as such more commonly be at risk of variation-driven failure and loss of quality, i.e. that they are optimal but not robust. In summation, integrated mechanical systems can hence be made more optimal and robust if these negative dependencies are avoided, or their number at least reduced to the bare minimum.

3.2 Trade-off management strategies

If a trade-off situation affects a design, what can one then do? Looking at perspectives from prior research and industrial practice, a designer can generally speaking either accept the existence and influence of a trade-off, attempt to tweak/optimise the design, or change the design itself. Common for all three is that it can be difficult to predict the outcome of deciding on one approach rather than another. *Concept A or B? Optimise or redesign*? Accept compromise on *objective 1* or *objective 2*? In the authors' experience, these are all decisions designers can struggle with, and the right approach depends on a wide range of factors. Yet, it would seem that the decision on what approach to apply is not necessarily made explicitly industrial practice, but rather done through tacit and explorative means. As such the management of one or several trade-off scenarios in a design will in some cases be handled with simultaneous or parallel activities - e.g. with a team of design engineers implementing design changes while also performing parametric updates and investigating the impact of compromise. In the following, six generic strategies (c.f. figure 3) available to designers for the management of trade-offs between functional objectives are discussed, along with their benefits and limitations, and the methodological support available in applying these strategies. They are not necessarily independent, and one could as such argue that combinations of otherwise these otherwise different strategies exist.

A trade-off situation between two max-is-best objectives with opposite monotonicities is used as an illustrative example, but the strategies are equally applicable to opposite targets and to trade-offs

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

involving other types of objective functions. Real-world examples are used to clarify the nature of these strategies. While the examples are somewhat simplistic, they still serve to illustrate the value in- and the need for methods that support designers in selecting the best trade-off management strategy for a given context, and also point to the potential pitfalls involved.

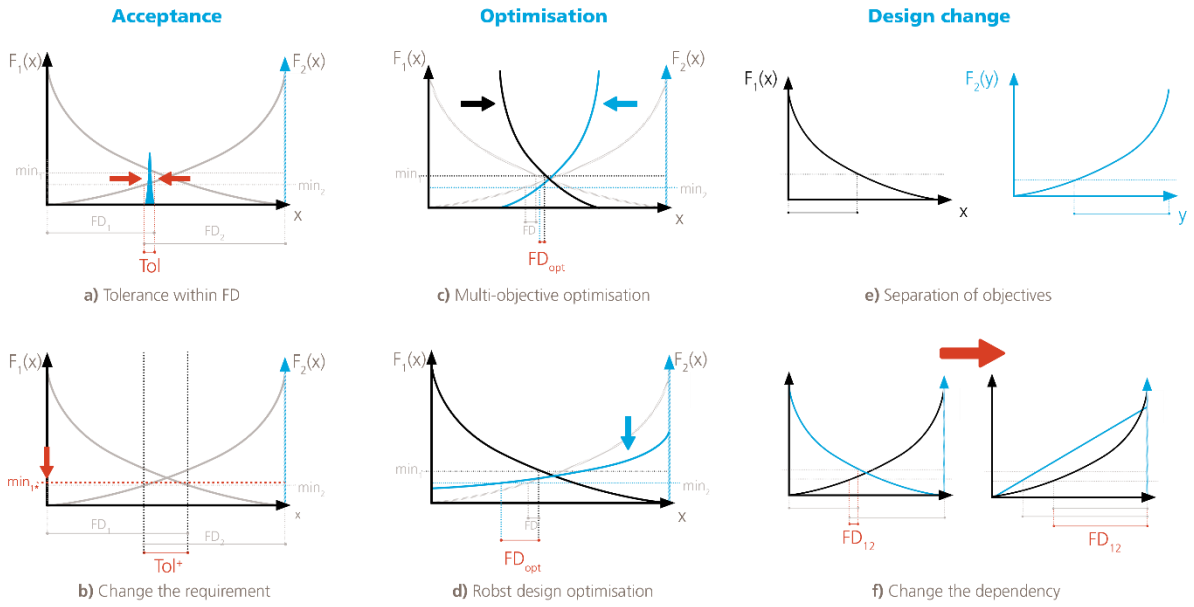

Figure 3 - Strategies to manage trade-off situations. a) Process control, b) Compromise, c) Improve performance d) Improve robustness e) Uncouple f) Change the dependency type

3.2.1 Trade-off acceptance

In accepting a trade-off, the designer does not alter the design in any way, rather relying on that the design can stay within specification in any conceivable state. In order to do so, it is necessary to ensure that the system always stays within the feasibility domain where both targets are met; in other words, the design parameter, x, cannot have a variance larger than the feasibility domain. This implies that the trade-off is managed through **tight tolerances/process control** (figure 3.a), the consequence being a potential increase in cost and a low robustness to degradation throughout the lifecycle. Entire books exist on the subject, with fields such as manufacturing engineering and process optimisation aimed at predicting and improving achievable tolerances and process quality. If it is impossible or costly to control the parameter sufficiently, an alternative would be to accept a **compromise by changing one or more of the requirements** (fig 3b). In doing so, the feasibility domain and therefore the allowable Parameter variance is widened. This implies a reduction in performance of the compromised objectives, and therefore potentially quality loss to the user. Methodologies such as compromise decision support problem (Mistree *et al.*, 1993) aim to support decision making in these situations, to help find the compromise that is most effective in increasing the overall performance.

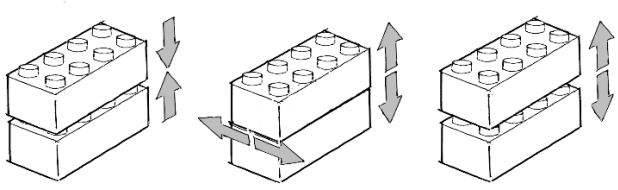

Figure 4 - LEGO should be easy to assembly and disassemble, yet stable once assembled

LEGO bricks are a good example where both approaches have been applied in design and production, to create a unique product offering. On one hand, LEGOs have to be easy to assemble and take apart, yet at the same time be highly stable once assembled. As such regular LEGO bricks are highly contradicting in that core objectives, assembly-, holding-, and disassembly force, have opposite objectives in relation to four controlling design parameters; pin diameter, thickness, interference fit, and material stiffness. Applying TRIZ would reveal that it is a physical contradiction, and solving it

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

involves separation in time, space, condition, or scale (Altshuller, 1984). These would require a solution that is vastly different from LEGOS, e.g. introducing a third joining component, some form of bi-stability, or a two-directional disengagement movement (e.g. pull and twist). Given that there is little added value to the user in achieving a lower assembly force and a higher stability, LEGO instead found a sweet-spot between all requirements, and managed it through tight process control and interface standardisation. This allows very simple use - single axis movement of two parts - yet in a more multi-functional product this would not necessarily be possible. While it is difficult to actually assert whether the inventor of LEGO consciously decided upon acceptance of the trade-off, it is still clear that the trade-off exists between two essential functional objectives exists. Given that the design still functions, variation in manufacture is surely being tightly controlled to ensure consistent quality.

3.2.2 Trade-off optimisation

Assuming that independent parameters exist that allow the objective functions to be adjusted individually, optimisation can be applied to reduce severity of the trade-off - either through systematic and formal optimisation techniques or through an iterative experience-driven approach where the designers tweaks and refines parameter values in the design to lessen the impact of the trade-off. Be it through formal or tacit means, applying **Multi-objective optimisation** (fig 3.c) and design optimisation techniques would ultimately change the gradients of the two objective functions locally within the limits created by the bounds and the trade-off itself. The result is an improved optimum of one or both of the objectives but a narrower feasibility domain, corresponding to a reduction in robustness. Alternatively, **Robust design optimisation** (fig 3.d) could widen the feasibility domain, but reduce the achievable optimum of one of the objectives. Both approaches are thoroughly described in a wide range of sources, yet in both cases, optimisation only changes the parametric design; the conflict still exits, and will limit the achievable optimum. The question when considering whether to optimise or redesign a system under trade-off, is whether the achievable optimum is sufficient, which is difficult to answer without performing the optimisation itself.

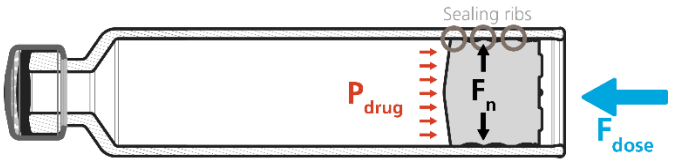

Figure 5 - Drug cartridges for medical injection devices are highly optimised designs, with a trade-off between avoiding leakage and achieving a low dosing force

Drug cartridges for medical devices (fig 5) are a simple example of a design where a trade-off has been mitigated through robustness- and multi-objective optimization. A dose is delivered by pushing a viscoelastic plunger forward, expelling the drug through a needle - the less resistance there is to this movement the faster the drug can be expelled. The plunger also acts as a hydraulic seal, preventing the drug from leaking out during dosing, with the plunger being pretensioned against the cartridge. Amongst the many design objectives in cartridges, are therefore a max-is-best drug sealing objective and a min-is-best dose force objective. A potential conflict arises in that the pretension of the plunger results in a normal force causing friction between the plunger and cartridge walls, which negatively influences the dosing force. This is handled through the minimisation of the coefficient of friction - an independent parameter that only influences the dose force - using a lubricant. Furthermore, the interface between plunger and cartridge walls has been reduced to three ribs to reduce the variation in sealing pressure that would otherwise occur in a continuous interface influenced by shape variation. Finally, the shape of the plunger is optimised towards achieving as homogenous a load distribution as possible while dosing, permitting an equal distribution of sealing pressure. Some of these features have been achieved through formal optimization, while others have been achieved through tweaking and experimentation (i.e. DoE), but it is some form of quasi-optimization nonetheless.

3.2.3 Trade-off mitigation through redesign

If a trade-off is to be avoided entirely, the design will inevitably need to be changed. This is the basic rationale behind theoretical frameworks such as Axiomatic Design and TRIZ. As discussed in section 2, the most common approach to trade-off avoidance or removal is to strive for designs that are

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

independent, i.e. uncoupled. In this context, the obvious approach to managing a trade-off in a design would be to change the design in a way that allows the **Separation of design objectives** (fig 3.e). By designing towards each objective being independent, there is no direct risk of trade-offs. However, this approach is not without limitations; sometimes objectives are inherently interdependent and a useful uncoupled solution therefore difficult to create. Furthermore, independence can require more components and sub-systems, to ensure that no elements of the design are shared between objectives. From a mechanical design perspective, this could - but does not necessarily - imply: 1) lowered mechanical efficiency and reliability, due to more contributors to losses and more features that can fail and 2) increased cost driven by extra components. Robustness wise, more parameters means longer tolerance chains, more load paths, and potentially more variation. A degree of integration (and ergo dependency) can as such be beneficial. With this in mind, aiming to **Change the dependency** (fig 3.f) between two objectives by design, to a positive dependency instead - thereby removing the trade-off - would allow simultaneous of optimisation of both objectives, and result in no reduction in the size of the feasibility domain of the system. g

An oft cited **example of separation** is the difference between the Newcomen steam engine and the Watt steam engine (Suh, 2001). The Newcomen engine (figure 6) is challenged in that the efficiency of the expansion and condensation cycles cannot be improved simultaneously, as both cycles occur inside the cylinder. Efficient and fast condensation relies amongst others on a high heat transfer through the outer wall of the cylinder, while the efficiency of expansion relies on no heat transfer at all; ideally the cylinder would always run warm. As such, the cylinder would ideally be infinitely thermally conductive in one state and infinitely insulated in another; a physical contradiction when viewed from a TRIZ perspective. In other words, this is a case of excessive integration, as two functionalities with opposite targets for the same parameter, with the Watt engine separating the condensator and cylinder, thereby separating the two in space, vastly improving the optimisability of the system.

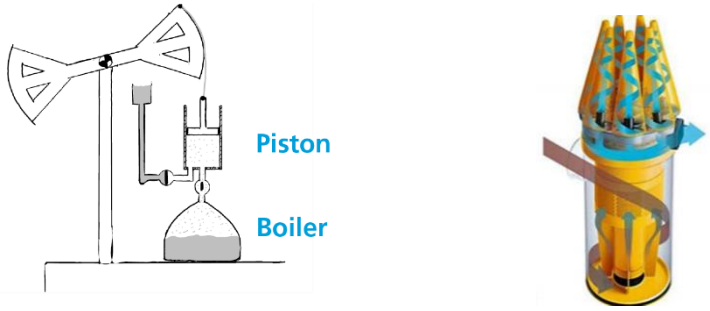

Figure 6 - Left) The Newcomen steam Engine (adapted from Suh (2001)), Centre) A bag-based vacuum cleaner (Right) A cyclonic separator from a Dyson Vacuum.

Changes in dependency can be difficult to exemplify, given that the design changes involved can be quite significant, not to mention the often large amount of design parameters and objectives involved. **A simple example of positive dependency** however, is the difference between vacuum cleaners with bags and bag-less vacuums. Regular vacuums created an airflow using an electric motor and fan, which then sucks air through a hose, with dust and debris being filtered out by an intermittent bag and secondary filter. The filtration in the bag is critical to avoid debris causing damage to the motor, but also to ensure that the dust is captured and not blown out again. Looking at two objectives, suction pressure and filtration quality which are of the type max-is-best, a trade-off reveals itself. The more efficient the bag is at filtration, the more resistance it creates, hindering the flow. In other words, the better the filtration, the more powerful a motor is required to generate a given suction pressure at the end of the hose. This also means that the suction pressure is reduced, the more the bag is filled. Bag-less vacuums meanwhile, commonly rely on cyclonic separation - a process that incurs less loss to the suction path. In fact, the filtration quality increases with the suction pressure, making the vacuums with cyclonic separation more optimisable and robust by design, with regards to these two objectives.

4 DISCUSSION - IMPLICATIONS OF TRADE-OFFS IN DESIGNS

As shown and discussed in the prior section, trade-offs can often be handled through process control in manufacture, requirement change, and optimisation. Such approaches are widely research and already supported by numerous well established methodologies and frameworks, (e.g. Papalambros & Wilde

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

(2017), Marler &Arora (2004), Mistree *et al*. (1994)). Yet, these are only truly applicable when the achievable feasibility domain is sufficiently wide and the achievable performance is sufficient. As such, trade-offs between functional objectives cause substantial issues relating to:

- **Tight tolerances or out-of-spec performance -** organisations that manufacture products that only function within too narrow sweet-spots between design objectives, will either have to rely on tight tolerances, or live with scrap, an increased functional dispersion, and the added reliability issues given the of two-sided variation driven failure that follows these dependencies.
- **Reduced performance** - trade-off situations limit the feasible domains of a design due to dependencies rather than constraints, reducing the achievable optimum.
- **Increased lead time** - With an increase in functional integration in a system, the risk of trade-offs grows proportionally. Designing highly integrated products is in the authors' experience often a matter of managing a "*system-of-sweets-spots"*, where each design change with one objective has a negative cascading effect across the system. The result is a highly iterative design process where time is spent on tweaking the design to continuously find new sweet-spots.

In other words, improperly managed trade-offs can result in less robust products that are less optimisable and more challenging to design. The decision-making involved in whether to accept, optimise or change a design, is in other words a cardinal process in designing high-performance, robust products. Yet it does not seem that there is any approach to support this process - i.e. helping the designer identify and classify the functional trade-offs in a design, and subsequently decide how to manage it (i.e. select one of the outlined approaches). Instead, the only way for the designer to identify the right strategy, is to actually apply all of them in parallel, and then compare the outcome.

Given that this is a time-consuming approach, experience shows that the *accept-optimise-redesign* decision is made based on tacit knowledge, and sometimes based on comparative analysis of the consequences of some of the options (e.g. assessing the tolerance required to accept the design, vs. the impact of changing the requirement). Based on the authors' experience from practice in numerous industries, this leads development teams to unconsciously applying all three approaches simultaneously in an unstructured manner, with substantial coordination complexity to follow. It is for instance not uncommon for parts of a team to start investigating the influence of requirement or specification change, while some colleagues investigate potential redesigns, and others attempt to tweak parameter values and evaluate the design through simulation or experiments. This is further complicated in multi-disciplinary applications such as electro-mechanics and mechatronics, where these trade-off scenarios become more multi-dimensional. While the strategies discussed could also apply in these cases, they are perhaps not exhaustive, with other options existing, given that redesign in e.g. the software or control domains do not necessarily imply radical changes to the system itself.

In lieu of the above, and the obvious benefits in designing functionally integrated products, why not attempt to design systems in a way that reduces the risk of functional trade-offs impacting robustness and performance? As discussed by Matthiassen (1997), a more comprehensive approach to integration is required, but no research has so far pointed to when to integrate and when to separate, beyond to do what is beneficial to the system. It would seem obvious that integration should be performed with trade-off management in mind, with a focus on embodying systems in a way that avoids severe trade-offs by design. In this context, the following conclusions can be drawn based on the prior sections:

- **Optimality and robustness by design** - A mechanical design will be optimisable and often robust, when all the objectives can be improved on without detriment to others. As such, the theoretically ideal integrated mechanical system only has positive dependencies.
- **Allocation of functionality** - Functionality integration and part reduction should, when possible, only be performed when the types and monotonicities of the objective functions are either the same, or when both type and monotonicity are simultaneously opposite.

How is this achieved? Systematic methodological support within this domain is scarce (c.f. sec. 2), which is surprising given the amount of methodological support within optimisation and trade-off acceptance. While TRIZ and axiomatic design both address the notion of avoiding detrimental dependency, they do necessarily not support the designer in deciding between acceptance, optimisation, or redesign, both in principle prescribing that all detrimental dependency should be solved by design. Furthermore, they primarily focus on approaches to removing the underlying the dependency (figure 3.e), rather than changing it (figure 3.f). There is in other words a potential in providing methodological support for designers aimed at managing trade-offs, specifically on how to decide between *acceptance, optimisation,* and *redesign*, and how actually perform said redesign.

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press

5 CONCLUSION

Trade-offs between functional objectives can have a significant impact on the performance of a mechanical system and the complexity and lead time of designing it. In the context of ensuring robustness and optimisability, six different approaches to managing trade-offs have been identified, falling within one of three categories; *accept*, *optimise*, or *redesign*. Yet there is no approach to assist designers in identifying the approach that best suits a given design, nor is the aspect of how to perform redesign to remove a trade-off well described. This can drive designers to select the wrong approach at the cost of robustness and/or optimisability. What this points to, is a vast potential in further research within methods for the identification, classification and management of trade-offs between objectives, and embodiment design methods that support optimal functional integration in mechanical systems.

REFERENCES

Ahmed, S., Wallace, K. and Blessing, L. (2003), "Understanding the differences between how novice and experienced designers approach design tasks", *Research in Engineering design*, Vol. 14 No. 1-1, pp. 1.

Altshuller, G.S. (1984), *Creativity as an exact science: The theory of the solution of inventive problems*, Taylor & Francis Group.

Andreasen, M.M. and Howard, T.J. (2011), "Is Engineering Design Disappearing from Design Research?", Chapter 2, *Future of Design Methodology*, Springer-Verlang, London.

Andersson, P. (1997), "On Robust Design in the Conceptual Design Phase: A Qualitative Approach", *Journal of Engineering Design*, Vol. 8, pp. 75–89.

Arthur, W.B. (1993), "Why do things become more complex?" *Scientific American*, May 1993, p. 144.

Boothroyd, G., Dewhurst, P. and Knight, W. (2002), *Product design for manufacture & assembly*, Taylor & Francis.

Ebro, M. and Howard, T.J. (2016), "Robust Design principles for reducing variation in functional performance", *Journal of Engineering Design*, Vol. 26 No. 1-3, pp. 75–92.

Eppinger, S.D. and Browning, T.R. (2012), *Design structure matrix methods and applications*, MIT press.

French, M. (1971), *Conceptual Design for Engineers*, Springer, Berlin.

Frey, D., Palladino, J., Sullivan, J. and Atherton, M. (2007), "Part Count and Design of Robust Systems", *Systems Engineering*, Vol. 10 No. 3, pp. 2007.

Göhler, S.M. and Howard, T.J. (2015), "The Contradiction Index (CI): A New Metric Combining System Complexity and Robustness for Early Design Stages", *Proceedings of the ASME IDETC/CIE 2015*.

Göhler, S.M., Frey, D. and Howard, T.J. (2016), "A model based approach to associate complexity and robustness in engineering systems", *Research in Engineering Design*, pp. 1–12.

Ilevbare, I.M., Probert, D. and Phaal, R. (2013), "A review of TRIZ, and its benefits and challenges in practice", *Technovation*, Vol. 33, pp. 30–37.

Marler, R.T. and Arora, J.S. (2004), "Survey of multi-objective optimization methods for engineering", *Structural & Multidisciplinary Optimisation*, Vol. 26, pp. 369–395.

Matthiassen, B. (1997), *Design for Robustness and Reliability - Improving Quality Consciousness in Engineering Design*, Technical University of Denmark 1997.

Mistree, F., Hughes, O. and Bras, B. (1993), "The Compromise Decision Support Problem and the Adaptive Linear Programming Algorithm", *Structural Optimization: Status and Promise*, AIAA, Washington, DC.

Olesen, J. (1992), *Concurrent Development in Manufacturing – based upon dispositional mechanisms, PhD-Thesis*, Technical University of Denmark.

Otto, K. and Antonsson, E. (1991), "Trade-off strategies in engineering design", *Research in Engineering Design*, Vol. 3 No. 2, pp. 87–104.

Pahl, G. and Beitz, W. (1996), *Engineering Design: A systematic approach*, Springer, Berlin.

Papalambros, P. and Wilde, D. (2017), *Principles of Optimal Design - Modelling and Computation*, 3rd edition, Cambridge University Press, Cambridge.

Ross, A., Hastings, D. and Warmkessel, J. (2004), "Multi-attribute Tradespace Exploration as Front End for Effective Space System Design", *Journal of Spacecraft and Rockets*, Vol. 41 No. 1.

Suh, N.P. (2001), *Axiomatic Design - Advances and Applications*, Oxford University Press.

Wagner, T.C. (1993), *A general decomposition methodology for optimal system design, Doctoral dissertation, Dept. of Mechanical Engineering*, University of Michigan.

Vianello, G. and Ahmed-Kristensen, S. (2012), "A comparative study of changes across the lifecycle of complex products in a variant and a customised industry", *Journal of Engineering Design*, Vol. 23 No. 2.

ACKNOWLEDGMENTS

The authors would like to thank the Danish Innovations Fund and the Novo Nordisk STAR-programme for funding this industrial research project (grant nr. 7038-00221B).

https://doi.org/10.1017/dsi.2019.356 Published online by Cambridge University Press