TÉCNICO LISBOA

Optimization in Architecture

Catarina Garcia Belém

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. António Menezes Leitão

May 2019

Acknowledgments

I would like to express my respect and gratitude to my supervisor and friend Dr. António Menezes Leitão. He proposed an interesting theme, which proved to be intriguing and challenging. His efforts to arrange research grants and to supply better computational resources were inspiring and encouraged me to fight the difficulties found along the way. His constant support, preoccupation and first-class guidance were invaluable through this thesis. Thanks for everything, especially for encouraging me to pursuit my dreams and for providing me with the flexibility and free-will to tackle this theme as something that I would be proud of.

I would like to thank the members of the research group oriented by my supervisor, the Grupo de Arquitetura e Computação (GAC), for their support and valuable ideas and discussions which undoubtedly improved the practicality of this work - especially, Inês Caetano, Inês Pereira, Renata Castelo Branco, Guilherme Ilunga, and Luís Silveira Santos.

I would also like to thank the Department of Computer Science and Engineering at Instituto Superior Técnico, Universidade de Lisboa for providing me with the foundations for completing this work, as well as for the opportunities to lecture as a teaching assistant during my MSc Thesis. I would also like to thank the Instituto de Engenharia de Sistemas e Computadores - Investigação e Desenvolvimento (INESC-ID) for the financial support provided to me in the form of Bachelor's Research Grants.

I am also grateful to the staff and teachers of the Computer Engineering and Information Systems course for their effort, dedication, availability, and for providing an interesting working environment.

To all my friends whose support was invaluable during this period and which encouraged me to constantly push my limits when the task felt too large, I thank you deeply from my heart - especially, Carolina Pereira, Cristiana Tiago, Diogo Magalhães, Filipe Magalhães, Gonçalo Rodrigues, Guilherme Ilunga, Nuno Afonso, Pedro Simão, Rita Amaro, and Telma Correia.

Last but not least, I would like to thank my parents for their friendship, encouragement and caring over all these years, for always being there for me through thick and thin and without whom this project would not be possible. I would also like to thank my sister, brother, and sister-in-law, for their understanding, support, and preoccupation throughout this year.

To each and every one of you – Thank you.

Contributions

The development of this thesis resulted in several scientific contributions exploring different perspectives of optimization problems:

1. Caetano, I., Ilunga, G., **Belém, C.**, Aguiar, R., Feist, S., Bastos, F., and Leitão, A. (2018). Case Studies on the Integration of Algorithmic Design Processes in Traditional Design Workflows. Proceedings of the 23rd International Conference of the Association for CAADRIA, 1(Giedion 1941), 111–120.

2. **Belém, C.**, and Leitão, A. (2018). From Design to Optimized Design An algorithmic-based approach. Proceedings of the 36th eCAADe Conference - Volume 2, Lodz University of Technology, Poland, 549-558

3. Martinho, H., Leitão, A, **Belém, C.**, Loonen, R, and Gomes, M. G. (2019). Algorithmic Design and Performance Analysis of Adaptive Façades. Proceedings of the 24th Annual Conference of the Association for CAADRIA, Welligton.

4. **Belém, C.**, and Leitão, A. (2019). Conflicting Goals in Architecture: A study on Multi-Objective Optimization. Proceedings of the 24th Annual Conference of the Association for CAADRIA, Welligton.

5. **Belém, C.** , Santos, L. S., and Leitão, A. (2019). On the Impact of Machine Learning: Architecture without Architects?. 18th International Conference CAAD Futures 2019, Korea (Submitted).

Abstract

Keywords

Algorithmic Design; Black-Box Optimization; Machine Learning; Surrogate-based Modelling.

Resumo

Palavras Chave

Design Algorítmico; Otimização de caixa-preta; Modelos baseados em aproximações; Aprendizagem Máquina.

Contents

List of Figures

List of Tables

List of Algorithms

Listings

Acronyms

AD	Algorithmic Design
AA	Algorithmic Analysis
BIM	Building Information Modelling
BPO	Building Performance Optimization
BPS	Building Performance Simulation
CAD	Computer-Aided Design
HVAC	Heating, Ventilation, and Air Conditioning
MOO	Multi-Objective Optimization
SOO	Single-Objective Optimization

1

Introduction

Contents

The act of making something as fully perfect, functional or effective as possible is a behavior that is constantly sought by us, Humans, in a process known as optimization [Online, 2018]. Intuitively, through optimization one aims to improve a system in terms of different quantitative measurable aspects. Although usually striving to fully optimize these systems, i.e., to obtain *perfect* systems, it is often the case, that finding a better one or a near-optimal system suffices.

Generally, optimization processes are composed of two main parts: (1) the model of the system to be optimized and (2) the algorithm responsible for finding the optima. Conceptually, the model is a description of the system that is defined in terms of: a set of the system's characteristics, known as variables or unknowns, a set of quantitative measures of the system's performance, referred to as objectives or criteria, and, optionally, by a set of conditions that have to be satisfied to guarantee the system's feasibility, i.e., the system's constraints [Nocedal and Wright, 2011]. The objectives are usually functions of the variables being defined. Subsequent to the model definition, the obtained description can be interpreted as an optimization problem for which the optimal solutions are to be found, thus entering in the second part of an optimization process. In the second part, one executes an optimization algorithm, which encloses a description of the steps necessary to attain optimal solutions, which according to the user's intentions can be the maximization or minimization of the model's objectives.

Depending on the model representation, one is able to classify optimization problems differently with respect, for example, to its objective functions, variables, and determinacy. Due to their relevance in the developed work, in the next two paragraphs, we describe four different optimization classifications. However, we refer the interested reader to [Nocedal and Wright, 2011] for a more detailed and complete description of the different classifications.

One important classification is regarding the cardinality of the solutions sought by optimization processes, thus yielding the continuous and discrete optimization categories. In the former, optimal solutions lie in a potentially infinite set of candidate solutions, whereas in the latter, optimal solutions lie in a finite set. Optimization problems can also be classified as constrained or unconstrained, depending on whether the models explicitly define constraints or not. Moreover, optimization can also be distinguished in terms of the aim of the search that is performed, particularly, whether it is global or local. In local optimization, the search process strives to find a solution that is locally optimal, i.e., for which its value is better than all other points in its vicinity. The points that satisfy the previous property are known as local optima. On the other hand, there are optimization processes that strive to find the globally optimal solutions, i.e., the best of all the local optima.

Optimization is frequently required to address problems involving more than one objective. It is often the case that people face daily decisions involving two or more conflicting objectives, either to effectively manage resources, or just to ponder several factors associated with certain decisions. A simple example is the act of purchasing something, e.g., a computer or a car. In these settings, people

often ponder different aspects to determine the best choice, i.e., to optimize its decision. In general, when looking for a computer, one is usually interested in minimizing the price of the computer, while maximizing the processing speed, the RAM and storage capacity and speed, and the battery autonomy. However, while some aspects are independent of each other, e.g., processing speed and RAM capacity, other aspects are conflicting and preclude the simultaneous satisfaction of all the objectives, e.g., the higher the processing, storage or RAM speed, the higher the price, the higher the processing speed, the worse the battery autonomy. While in this example, we only consider a subset of aspects many other might be considered, possibly complicating the decision process. Because this example considers the optimization of more than one aspect, it belongs to a subset of problems, known as Multi-Objective Optimization (MOO) problems. Whereas the set of problems focused in the optimization of a single aspect are known as Single-Objective Optimization (SOO).

In addition to day-to-day life decisions, optimization can also be used with different decision and analysis purposes. As a result, several areas, including economy, science, engineering, among others apply optimization as auxiliary tool to maximize the efficiency of the decisions involved. Particularly, architecture is one of the areas where the potential of optimization becomes more visible. The architectural practice can benefit from optimization to reduce the building sector's economic and ecological footprint through the finding of more efficient buildings variants before their construction. Given its importance to the world's sustainability and economy, this thesis focus on the application of optimization processes to enhance the architectural practice, providing, in the following sections, an overview of the involvement and the evolution of such processes in architecture. We end this chapter by highlighting our research goals and by outlining the upcoming document's structure.

1.1 From design to Optimized design

The usefulness of optimization goes beyond architectural design applications, benefiting other engineering fields like Mechanics or Electronics, through the optimization of its components and circuits designs, respectively.

In the architectural practice, optimization has been gaining relevance for the past few years [Cichocka et al., 2017], especially due to the impact of building construction and building maintenance in the economy and environment. For this reason, designers are shifting from a pure aesthetically-based to performance-based design, where buildings are being optimized to achieve the best possible values regarding different aspects of their design, such as thermal comfort, energy consumption, lighting comfort, structural behavior, cost, among others.

This has only been possible due to the technological improvements in the architectural practice over the last few decades. The adoption of computer science techniques was responsible for the dissemina-

tion of digital modelling tools, which allowed the more accurate and efficient design of highly complex buildings. These tools enabled a shift from traditional paper-based approaches to more computerized ones, such as Computer-Aided Design (CAD) and Building Information Modelling (BIM) applications, where changes to designs are trivial and do not require manually erasing and redrawing parts of the original design [Ferreira and António, 2015]. Figure 1.1 illustrates the general view of this computer-aided design process, as well as an example of a 3D modeling tool. The architect interacts directly with these modeling tools to incrementally concretize his design ideas.

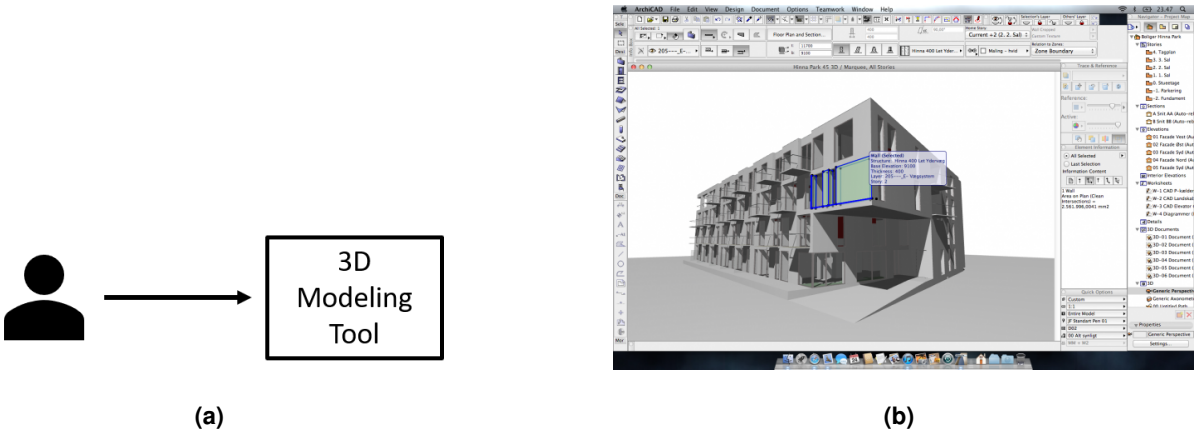


Figure 1.1: (a) Computer-aided design workflow (b) Building design example in a 3D modeling tool

Shortly after, the development of computer-based simulation tools allowed designers to simulate their building's behavior regarding specific criteria, and, thus get a measurement of its performance [Malkawi and Kolarevic, 2005]. Through this process, called Building Performance Simulation (BPS), designers could easily validate whether their building's performance satisfied the efficiency requirements and, ultimately, optimize their design by iteratively generating multiple variations of the same design, assessing their performance, and selecting the better ones. Albeit being very primitive, architects now had the elementary mechanisms required for optimizing their building's designs.

1.1.1 Building Performance Optimization

Building Performance Optimization (BPO), a simulation-based optimization approach, treats the results produced by the simulation tool as the functions to optimize. Although invariably suffering from some degree of imprecision and inaccuracy, using these simulations it becomes possible to estimate the performance of complex designs. Particularly, these estimates are beneficial in designs for which analytical solutions are often very difficult or even impossible to derive [Kolda et al., 2003]. In these cases, the objective function, i.e., the function to optimize, is derived from the simulations' results. These objective functions have a domain which corresponds to the range of acceptable designs as specified by the

architect.

A known drawback of simulation-based approaches is the time required to achieve reasonable results for complex systems [Law and Kelton, 1991] which is associated with different aspects of the problem, namely (1) its **domain** which, depending on the nature of the problem, might use different methodologies to produce the corresponding estimates (e.g., thermal *versus* structural); (2) its **intrinsic structure** which, depending on the attributes and relations of the system, might lead either to simpler or to more complicated computations (e.g., skyscraper *versus* a small house); and (3) its **analytical model**, which has the essential properties of the system we are trying to simulate and that will be used as input to the simulation tool. Generally, the domain and structure do not change for the same problem, albeit there are numerous ways to produce multiple analytical models. Depending on the level of detail of the analytical model, both the computational time and the result of the simulation might change.

Dar exemplo??

In architecture, the generation of each analytical model is a time-consuming and complicated task. On the one hand, it is often necessary to generate multiple models of the same design because of the simulation tools' specificity, i.e., in order to evaluate a design, each simulation tool requires a specialized model of the same design. On the other hand, simulation tools often yield time-consuming processes, where a single simulation can take up to seconds, minutes, hours, days, or even weeks to complete.

In addition to the simulations' specificity and complexity, architectural designs are inherently complex, thus leading to less predictable objective functions, for which mathematical forms are difficult to formulate [Machairas et al., 2014]. For this reason, information about the derivatives of such functions cannot be extracted and methods depending on function derivatives cannot be used to address architectural optimization problems. As a result, classical gradient-based optimization methods can not be used. Instead, other methods that do not rely on the existence of an explicit mathematical form should be used, i.e., methods that treat the optimization functions as black-boxes, relying uniquely on the outputs of numerical simulations.

Despite the flexibility provided by CAD and BIM tools, architects often face difficulties when modeling complex geometry. A BPO methodology requires to experiment with different design variations, which implies spending a large amount of time to manually make changes to the design. Since an optimization process requires evaluating several variations of the same design, the manual execution of the required changes implies a lot of effort, hence making the whole optimization process unviable.

1.1.2 Algorithmic Design

An approach capable of creating forms through algorithms is crucial for overcoming the aforementioned limitations. An example of an algorithm approach is the Algorithmic Design (AD) approach [Branco and Leitão, 2017], which is depicted in Figure 1.2. In this approach, the architect entails an algorithmic description of the intended design. After elaborating the algorithm, executing it will automatically generate

the corresponding 3D model in a CAD or BIM tool. Algorithmic approaches are inherently parametric, thus enabling the generation of different variations of the same design by making simple modifications to the values of the parameters [Leitão et al., 2014].

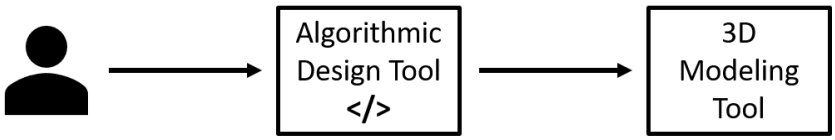
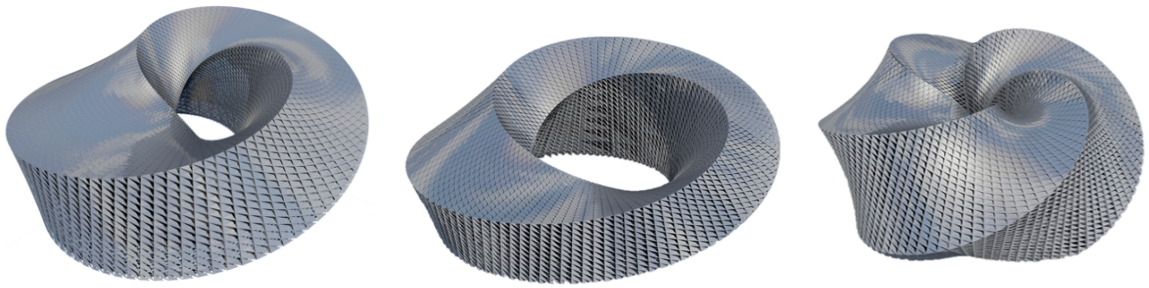


Figure 1.2: Algorithmic-based design workflow

As an example consider the algorithmic design of the Astana's National Library from BIG architects, illustrated in Figure 1.3(a). Initially, the architect selects an AD tool providing the necessary design primitives. Then, he uses these primitives to create a computational program enclosing the relative relations among the different design elements, so that when a modification occurs in one element, that modification is propagated throughout that element's dependencies. In the end, the architect creates a procedure responsible for creating the whole design `astana(radius, nturns)`, which when executed will produce the corresponding Astana 3D model. Using this procedure, he can easily generate different variations of the Astana by simply setting different values for the parameters: `radius`, defining the width of the whole design, and `nturns`, defining the number of turns in the design. Figure 1.3 illustrates the original design and two design variations are represented: (a) the original design (b) design variation generated by doubling the radius of the original design, and (c) design variations generated by doubling the number of turns.



(a) **(b)** **(c)**

Figure 1.3: Astana's National Library Design variations: (a) Original (b) Larger diameter (c) Two Moebius Turns

Only recently has the algorithmic paradigm begun to settle in the architectural practice. The requirement for programming knowledge is often an obstacle to the adoption of these approaches, since it requires larger initial investments for the architect to learn how to program. Despite the larger initial investments, the benefits obtained with algorithmic approaches surpass the ones obtained when us-

ing CAD or BIM tools for the design of complex buildings. Particularly, the initial investments can be quickly recovered when the need for the incorporation of changes arises or when it becomes necessary to experiment different design variations [Leitão et al., 2014]. This is especially important when facing design processes characterized by continuously changing constraints and requirements of the design. In these scenarios, a manual-based approach requires to constantly change the design by hand, thus incurring in a dreadful and tiresome process, whereas an algorithmic approach enables the effortlessly generation of a broader range of design solutions, as well as the easy modification of the algorithmic model to comply with new requirements. As a result, with algorithmic approaches, architects are able to explore larger regions of the design solution space, as well as to explore innovative solutions which were not previous considered due to their time and effort complexity [Leitão et al., 2014].

The appearance of AD was crucial for the automation of optimization processes as it enables the automated generation of multiple designs by simply changing the values of the design's algorithmic model. However, to optimization such designs, it is necessary to create the analytical models, which can be very dissimilar to the 3D models produced by the AD tool. Therefore, to evaluate the 3D models produced by the AD tool, the architect must manually generate the corresponding analytical models. Particularly, for complex buildings, this task requires large time and effort investments, which make most optimization processes impracticable.

1.1.3 Algorithmic Analysis

Faster and broader design space exploration prompted the creation of increasingly complex building designs, which became less predictable with respect to different aspects [Branco and Leitão, 2017], such as thermal, lighting, acoustics, among others. Moreover, recent requirements for efficient and sustainable buildings led to the demand for buildings that not only are well-designed, but also exhibit a good performance at those aspects.

Nowadays, most of the available simulation-based analysis tools are single-domain toolsets, each analysing different parameters within their domain [Malkawi and Kolarevic, 2005], i.e., while a lighting analysis tool measures daylight and glare coefficients, an energy simulation tool measures other coefficients related to thermal, energy consumption, and Heating, Ventilation, and Air Conditioning (HVAC) systems. Unfortunately, this often implies the production of different analytical models for the corresponding simulation tools. Moreover, the 3D models produced by 3D modeling tools are generally dissimilar to the specialized models required by each analytical tool. To evaluate the design performance on different domains (e.g., lighting, energetic, structural), the corresponding analytical models have to be produced either by hand, or through translation processes that convert generic 3D models into specialized models required for analysis.

Currently available techniques for the production of analytical models exhibit a few limitations, includ-

ing: (1) hand-made analytical models might be a more faithful representation of the original model but they require a considerable amount of effort to create; (2) despite the existence of tools that attempt to convert a 3D model into the corresponding analytical model, this conversion is frequently fragile and can cause loss of information or errors; (3) ideally, the results of the analysis would then be used to guide changes in the original design, but these changes require additional time and effort to implement, as does redoing the analysis to confirm the improvements. This explains why performance analyses are typically postponed to later stages of the design process, only to assess the fulfillment of the performance requirements.

To overcome the time and effort limitations associated with the production of analytical models, one can exploit the ideas of algorithmic approaches to automatically generate the necessary analytical models from an algorithmic description. Algorithmic Analysis (AA) is an extension to the AD approach that besides enabling the automatic generation of analytical models from a design's algorithmic model, also automates the setup of the analysis tool and the collection of its results [Aguiar et al., 2017]. Following this approach, in an AD tool, the architect creates the algorithmic model that describes his design's intents and then changes its configurations as to reflect the analysis tool to use, for example, by changing the values of the configuration parameters. Figure 1.4 illustrates the AD and AA design workflow, as well as examples of the Astana's National Library models produced for each tool. Note that, even though the abstract description of the design is the same for 3D and analytical models, the produced models can be very different, e.g., a truss is represented as a graph of nodes and edges when submitted for structural analysis, whereas for lighting analysis the truss is represented by its surfaces, and, in the 3D model, the truss is represented by a set of masses representing its bars and nodes.

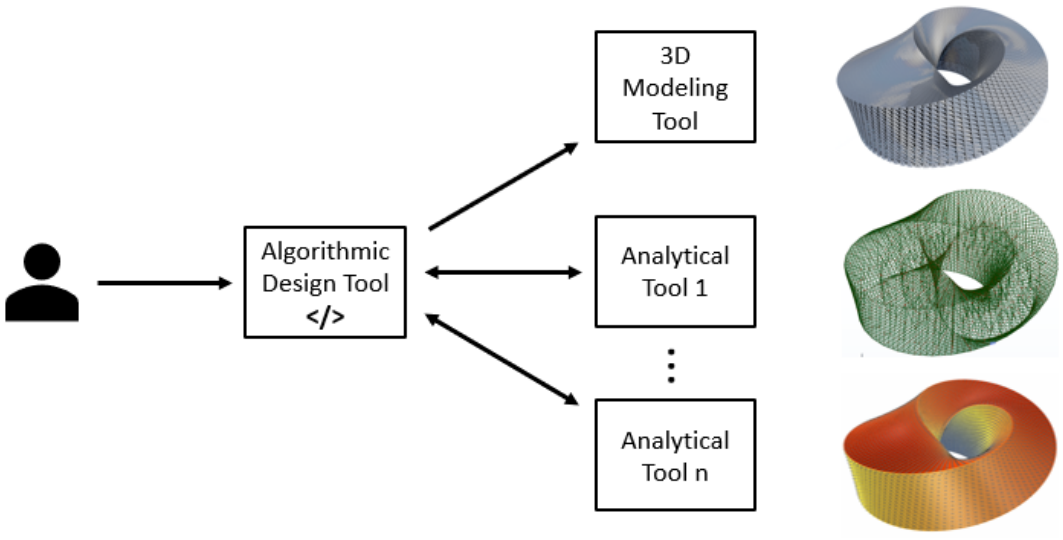


Figure 1.4: Algorithmic Design and Analysis design workflow with examples of the Astana's National Library design and analytical models: (top) 3D model; (center) Robot's structural analysis model; (bottom) Radiance's pos-radiation analysis model.

The AA approach is able to enhance performance-based design approaches, as it provides means to effortlessly perform analysis throughout the whole design process, instead of just at final stages. Depending on the performance requirements, architects might need to use different analysis tools: (1) for daylight analysis, DAYSIM and Radiance are very popular among the community, (2) for energy simulations, EnergyPlus, TRNSYS and DOE-2 are widely used [Nguyen et al., 2014], (3) for structural analysis, Robot Structural analysis is a well-known reputed tool, and (4) Olive Tree Lab and Pachyderm Acoustical Simulation are examples of good acoustic analysis tools.

The AA approach was also very important to allow the automation of optimization processes, as it abstracted the production of the analytical model, removing the need for direct human intervention and reducing the errors and information losses. Additionally, together with AD, it provides the mechanisms to quickly update a design, to generate the corresponding analytical model, to automatically evaluate the design in an analytical tool, and, finally, to collect the results and use them to guide the search for optimal solutions. Despite being possible to automate optimization processes, in order to do so, the architect must define a script entailing an optimization algorithm every time he intends to attain better designs. Besides lacking enough knowledge or expertise about optimization algorithms themselves, in order to entail one, architects would have to either create their own algorithm, or they would have to use one available in some optimization framework. Moreover, because of the efforts associated with the implementation of the optimization algorithm, architects will be tempted to adopt the same algorithm to optimize every design, thus instilling performance impacts in the overall optimization time. Among other obstacles, the large time and investment efforts, as well as the lack of knowledge are often main setbacks to the application of optimization processes in architectural design contexts.

1.1.4 Architectural Optimization Workflow

In architecture, design optimization might be approached differently. In the past, most optimization processes were comprised of different frameworks, which often had to be integrated with each other. Architects often attempted to integrate existing mathematical optimization and visualization frameworks within their workflow. Due to their specificity integration problems were often obstacles to the optimization process itself. More recently, the emergence of visual parametric approaches, such as Grasshopper, Dynamo, among others, coupled with the growing consciousness of both the limitations and the benefits of optimization in building design have led to the development of ready-to-use optimization toolsets (e.g., Galapagos, Goat, Octopus, Opossum, Optimo). However, despite enabling the optimization of several designs, visual parametric tools are known to scale poorly with the complexity of the design , thus diminishing and constraining its optimization capabilities.

COLOCAR REF

On the other hand, algorithmic approaches are known to scale well with designs' complexity. In addition to its scalability benefits, its growing popularity among building design practitioners [De Kestelier

and Peters, 2013], its models' flexibility, as well as its capacity to automate optimization processes allow the development of more robust and complete optimization tools. To successfully take advantage of such a tool, an AD-based design workflow optimization methodology must be followed. In this approach, the architect idealizes a design which he ought to produce in the corresponding AD tool. For this purpose, he creates the computational program, defining the parameters that represent the degrees of freedom in the design, i.e., the parameters which he is willing to manipulate in order to achieve more efficient designs. After the conception of the design's algorithmic model and, provided the values for the parameters, the AD tool generates 3D or analytical models for visualization and performance analysis purposes, respectively. Optionally, the architect may decide to optimize his design according to some particular aspects, potentially leading to the exploration of design solutions that were not previously considered. In that case, the optimization algorithm explores different design candidates, using the results produced by simulation tools as the functions to optimize. The execution of the optimization algorithm then yields optimal (or near optimal) design solutions.

Considering the previous view of an algorithmic-based design workflow, we identify four key dimensions in an optimization process:

1. **Analytical models**: when the optimization algorithm specifies a candidate design, i.e., a concrete configuration for the parameters of the model, analytical models are automatically generated by the AD tool. These models are then used as input for the corresponding analytical tools. These models can be improved, either through simplification of the analytical models, or by enriching them with context information. The former enables the simplification of the analysis itself and potentially reduces the simulation time, by providing an equivalent but simpler model to the tool, whilst the latter enables the attainment of a more detailed and realistic simulation, which is not always possible due to limitations in the AD tool.

2. **Optimization algorithms**: the algorithms that explore the design space in the quest for optimal (or near optimal) solutions. These algorithms use the results obtained from performance analysis of different design variations as the functions to optimize, i.e., as objective functions. Generally, the algorithm uses these inferred functions to guide the search for optimal solutions. The time complexity of the algorithm is typically dependent on the number of function evaluations. In architectural design, these functions entail time-intensive simulations, thus instilling optimization processes that may take minutes, hours, days, or even weeks to complete.

3. **Intelligibility of Results**: Cichocka et al. identify the need for intelligibility of the optimization processes within the architectural community [Cichocka et al., 2017]. Having access to an explanation regardig the quality of a design solution allows architects to make more informed decisions. With these explanations, the archtiect can not only provide valuable arguments for its implementation,

but also, depending on the quality of the explanations, learn with the process, thus fostering more efficient and faster future designs.

4. **Interactivity and Visualization**: interactive and visual aspects are highly important features in the context of an optimization process [Ashour, 2015]. On the one hand, an interactive optimization process enables its user to transfer knowledge about the problem at hand, for instance, by adding or removing constraints or by exploring different, yet unexplored regions of the design space, hence potentially increasing this process' performance. On the other hand, optimization processes providing better visualizations and representations of their own evolution can present their users with better feedback about the course of the search. This feedback is important, as it also allows the comparison of variable-objective correlations and the making of more informed decisions about the optimization process itself, e.g., whether the evaluations made so far suffice or if the algorithm is converging to non-conventional designs that he refuses to accept.

1.2 Goals

The interest in design optimization is evident within the architectural community, however the currently existing tools are often fragile or limited, frequently compromising the application of optimization in architecture. This thesis focus on optimization processes within the architectural domain, providing a framework for optimizing both single and multi-objective problems. The implementation of such framework requires the definition of: (1) a modeling language to support the specification of optimization problems, (2) a wide variety of optimization algorithms to solve optimization problems, and (3) a visual presentation of the obtained results to provide a more comprehensible feedback over the optimization results.

To achieve the goals that we proposed to, we reviewed different mathematical optimization modeling languages and optimization frameworks, pondering the benefits and obstacles of each one. Based on these languages and frameworks, we established the basis requirements for the framework that enable its seamless application within the architectural practice.

1.3 Organization of the Document

This thesis is is organized as follows:

Chapter 1 discusses optimization concepts and evidences its importance for different problems, ranging from simpler day-to-day decisions to more complex engineering problems, such as components, circuits, and building designs. Particularly, this chapter stresses the relevance of optimization in the architectural context, providing a comprehensive overview of the existing practices and the difficulties underlying the

adoption of optimization processes in architecture.

Chapter 2 provides an overview over the currently existing optimization practices in architecture and makes a balance of the benefits and drawbacks associated to each one.

Chapter 3 describes the architecture of the implemented framework and enumerates important design decisions that were made during its implementation.

Chapter 4 evaluates both quantitative and qualitative aspects of the proposed solution, evaluating its performance in the context of three real-world case studies.

Chapter 5 emphasizes the importance of optimization in architecture and draws some conclusions about the final work and how it can effectively influence the architectural practice. Finally, we reflect over future improvements for the proposed frameworks.

2

Background

Contents

The development of an algorithmic-based framework for optimization, applicable to architectural domains, requires a careful review over the current literature on BPO practices and limitations.

Firstly, the *ad-hoc* nature of the functions used for performance assessment in BPO motivates the application of a special class of optimization algorithms, the derivative-free algorithms. Within this class, different categories emerge, emphasizing algorithms' different properties and search strategies. Each algorithm is able to potentially increase the effectiveness of an optimization process, depending on the characteristics of the considered problem.

Secondly, there are multiple approaches to optimization that might be considered. Generally, BPO practices include the simultaneous optimization of multiple aspects. However, they often opt for simpler specifications, often disregarding all but one of the initial considered aspects.

Finally, currently available architectural design optimization tools explore the parametric models produced in visual programming environments, such as Grasshopper and Dynamo. These graphical environments are implemented as plug-ins, which are tightly integrated with a CAD and a BIM tools, respectively. As a result, the connection between optimization tools and visual design workflows becomes seamless and friendlier. These optimization tools usually expose a *ready-to-run* interface, which is very appealling to most BPO practitioners [Cichocka et al., 2017].

2.1 Derivative-Free Optimization

Different optimization algorithms can solve more or less efficiently specific optimization problems, depending on their characteristics. Particularly, classical gradient-based algorithms are very efficient solvers for optimization problems explicitly defined by mathematical formulations. This results from the fact that gradient-based algorithms explore information about the derivatives extracted from the mathematical formulation to guide the search for optimal solutions. However, when neither the mathematical form is easily available, nor is the information about the derivatives, it becomes necessary to explore other classes of algorithms. In these cases, the class of derivative-free algorithms is remarkably suitable for addressing these problems, as they do not use information about the objective functions' derivatives to find optimal solutions, instead, treating the objective functions as *black-boxes* and guiding the search based on the result of previously evaluated solutions [Rios and Sahinidis, 2013].

In architecture, it is often impossible to mathematically model the underlying objective functions for complex designs. Alternatively, architects use simulation tools as means to replace closed-form mathematical expressions that relate the design's parameters to the objective functions: simulation tools' results and other quantitative measures for different configurations of a design define the objective functions to optimize [Wortmann and Nannicini, 2016]. Additionally, information about underlying objective functions is not easily attainable, often requiring excessive amounts of resources. The absence of in-

formation about objective functions prompts the need for algorithms that treat these functions as *black-boxes*. One simple approach is to systematicly experiment with different parameter values until the best solutions are found, whereas a second, and more complex, approach is to use derivative-free optimization algorithms, also commonly referred to as black-box optimization algorithms within the architectural community [Wortmann and Nannicini, 2016]. Despite its simplicity, experimentation-based approaches, such as **MCS!** (**MCS!**) and **LHS!** (**LHS!**) [Giunta et al., 2003], might not always be advisable, particularly when dealing with time-consuming functions as is the case of architecture. In such cases, derivative-free algorithms might be more appropriate, yielding better solutions in less time.

Building design's complexity has been raising for the past few years, leading to more complex objective functions, for which analytical forms are difficult to derive [Machairas et al., 2014]. For this reason, derivative-free algorithms are sought as useful tools to optimize designs, having been applied extensively to optimize building designs' manifold aspects. Among the numerous studies that apply derivative-free optimization algorithms to optimize building designs, we refer to the distinct works of Wortmann [Wortmann and Nannicini, 2016, Wortmann et al., 2015, Wortmann et al., 2017, Wortmann, 2017], Evins [Evins and Vaidyanathan, 2011, Evins et al., 2012, Evins, 2013], and Waibel [Waibel et al., 2018] which cover the optimization of various aspects, including, among others, the structural, the lighting, the thermal, the energy consumption, and the carbon-emissions.

For the past decades, the constant development and improvement of derivative-free optimization algorithms led to a diversified tools' gamut, each with its own characteristics and limitations. While the main ideas behind each algorithm's category seem to be more or less recognized throughout the architectural community, the lack of standards make it difficult to decide which definitions to convey [Rios and Sahinidis, 2013, Wortmann and Nannicini, 2017]. The currently most relevant classifications are: (1) the one presented by Rios et al. [Rios and Sahinidis, 2013] that, based on the functions being used to guide the search process, classifies the algorithms into direct search or model-based algorithms; and (2) the classification provided by Wortmann et al. [Wortmann and Nannicini, 2017], which first subdivides the algorithms in two groups according to the number of solutions generated in each iteration, namely metaheuristics and iterative algorithms, and only then proceeds to classify iterative algorithms as direct search or model-based algorithms depending on the function that was applied during the search for optimal solutions.

This thesis will consider an approach similar to the one proposed by Wortmann [Wortmann and Nannicini, 2017] by exploring the concepts of metaheuristics, direct-search, and model-based algorithms. Albeit the apparent chasm between these classifications, some algorithms draw ideas from distinct classes, thus emphasizing not only the blurred lines of such categorizations, but also the difficulties that lie with the definition of more standardized classifications.

In the following sections, we describe each class of algorithms and its intrinsic characteristics, pro-

ceeded by a brief comparison of such classes in light of the architectural design practice.

2.2 Single-Objective Optimization

2.2.1 Optimization Tools in Architecture

2.2.1.A Galapagos

2.2.1.B Goat

2.3 Multi-Objective Optimization

Nunc tincidunt convallis tortor. Duis eros mi, dictum vel, fringilla sit amet, fermentum id, sem. Phasellus nunc enim, faucibus ut, laoreet in, consequat id, metus. Vivamus dignissim [**?**]. **??** is automatically compressed to fit text width. You can use `https://www.tablesgenerator.com` to produce these tables, and then copy the LaTeX code generated to paste in the document.

2.3.1 Experimental Approach

2.3.2 Priori Articulation Approach

2.3.3 Pareto-Based Approach

2.3.4 Metrics for Multi-Objective Optimization

2.3.5 Optimization Tools in Architecture

2.3.5.A Octopus

2.3.5.B Opossum

2.3.5.C Optimo

3

Solution

Contents

Donec gravida posuere arcu. Nulla facilisi. Phasellus imperdiet. Vestibulum at metus. Integer euismod. Nullam placerat rhoncus sapien. Ut euismod. Praesent libero. Morbi pellentesque libero sit amet ante. Maecenas tellus. Maecenas erat. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

3.1 Architecture Overview

3.2 Architecture Design Requirements

3.2.1 Problem Modelling

3.2.2 Simple Solver

3.2.3 Meta Solver

3.3 Architecture Design Implementation

3.3.1 Problem Modelling

3.3.2 Simple Solver

3.3.3 Meta Solver

4

Evaluation

Contents

- Relembrar o objectivo do trabalho e dizer como o vamos avaliar de um modo geral introduzindo os proximos subcapitulos.

4.1 Qualitative Evaluation

- Number and Heterogeneity of Available algorithms - Differences / Benefits / Disadvantages when compared to Grasshopper's frameworks

4.2 Quantitative of Applications

- Dizer que de um modo geral começámos de forma incremental por considerar problemas single-objective, nomeadamente a casa da ericeira, que remonta a primeira publicação. Depois evoluimos para a avaliação bi-objetivo de dois casos de estudo reais - Pavilhão Preto para exposições e de uma arc-shaped space frame.

- Comentar a facilidade c/ que alguém que já tem um programa AD consegue acopolar optimização a AD.

4.2.1 Ericeira House: Solarium

4.2.2 Black Pavilion: Arts Exhibit

4.2.2.A Skylights Optimization

4.2.2.B Arc-shaped Space Frame Optimization

5

Conclusion

Contents

Pellentesque vel dui sed orci faucibus iaculis. Suspendisse dictum magna id purus tincidunt rutrum. Nulla congue. Vivamus sit amet lorem posuere dui vulputate ornare. Phasellus mattis sollicitudin ligula. Duis dignissim felis et urna. Integer adipiscing congue metus.

Rui Cruz
You should always start a Chapter with an introductory text

5.1 Conclusions

5.2 System Limitations and Future Work

5.2.1 Optimization Algorithms

5.2.2 ML models

5.2.3 Constrained Optimization

Aliquam aliquet, est a ullamcorper condimentum, tellus nulla fringilla elit, a iaculis nulla turpis sed wisi. Fusce volutpat. Etiam sodales ante id nunc. Proin ornare dignissim lacus. Nunc porttitor nunc a sem. Sed sollicitudin velit eu magna. Aliquam erat volutpat. Vivamus ornare est non wisi. Proin vel quam. Vivamus egestas. Nunc tempor diam vehicula mauris. Nullam sapien eros, facilisis vel, eleifend non, auctor dapibus, pede.

Bibliography

[Aguiar et al., 2017] Aguiar, R., Cardoso, C., and Leitão, A. (2017). Algorithmic Design and Analysis Fusing Disciplines. pages 28–37.

[Ashour, 2015] Ashour, Y. S. E.-D. (2015). Optimizing Creatively in Multi-Objective Optimization.

[Branco and Leitão, 2017] Branco, R. C. and Leitão, A. M. (2017). Integrated Algorithmic Design: A single-script approach for multiple design tasks. *Proceedings of the 35th Education and research in Computer Aided Architectural Design in Europe Conference (eCAADe)*, 1:729–738.

[Cichocka et al., 2017] Cichocka, J. M., Browne, W. N., and Rodriguez, E. (2017). Optimization in the architectural practice. *Protocols, Flows and Glitches, Proceedings of the 22nd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2017*,, pages 387–397.

[De Kestelier and Peters, 2013] De Kestelier, X. and Peters, B. (2013). *Computation Works: The Building of Algorithmic Thought*, volume Computation Works: The Building of Algorithmic Thought.

[Evins, 2013] Evins, R. (2013). A review of computational optimisation methods applied to sustainable building design. *Renewable and Sustainable Energy Reviews*, 22:230–245.

[Evins et al., 2012] Evins, R., Joyce, S. C., Pointer, P., Sharma, S., Vaidyanathan, R., and Williams, C. (2012). Multi-objective design optimisation: getting more for less. *Proceedings of the Institution of Civil Engineers - Civil Engineering*, 165(5):5–10.

[Evins and Vaidyanathan, 2011] Evins, R. and Vaidyanathan, R. (2011). Multi-objective optimisation of the configuration and control of a double- skin facade. In *12th Conference of International Building Performance Simulation Association, Sydney*, number November, Sydney.

[Ferreira and António, 2015] Ferreira, B. and António, L. (2015). Generative Design for Building Information Modeling. *Proceedings of the 33rd Education and research in Computer Aided Architectural Design in Europe Conference*, 1:635–644.

[Giunta et al., 2003] Giunta, A., Wojtkiewicz, S., and Eldred, M. (2003). Overview of modern design of experiments methods for computational simulations. *Aiaa*, 649(July 2014):6–9.

[Kolda et al., 2003] Kolda, T. G., Lewis, R. M., and Torczon, V. (2003). Optimization by Direct Search - New Perspectives on Some Classical and Modern Methods. *Society for Industrial and Applied Mathematics*, 45(3):385–482.

[Law and Kelton, 1991] Law, A. M. and Kelton, W. D. (1991). *Simulation modeling and analysis*, volume 2.

[Leitão et al., 2014] Leitão, A., Santos, L., and Fernandes, R. (2014). Pushing the Envelope: Stretching the Limits of Generative Design. *Blucher Design Proceedings*, 1(7):235–238.

[Machairas et al., 2014] Machairas, V., Tsangrassoulis, A., and Axarli, K. (2014). Algorithms for optimization of building design: A review. *Renewable and Sustainable Energy Reviews*, 31(1364):101–112.

[Malkawi and Kolarevic, 2005] Malkawi, A. and Kolarevic, B. (2005). *Performative Architecture: Beyond Instrumentality*, volume 60.

[Nguyen et al., 2014] Nguyen, A.-T., Reiter, S., and Rigo, P. (2014). A review on simulation-based optimization methods applied to building performance analysis. *Applied Energy*, 113:1043–1058.

[Nocedal and Wright, 2011] Nocedal, J. and Wright, S. J. (2011). *Numerical optimization*. Number 2.

[Online, 2018] Online, M. W. (2018). Merriam Webster Online - Optimization Definition.

[Rios and Sahinidis, 2013] Rios, L. M. and Sahinidis, N. V. (2013). Derivative-free optimization: A review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293.

[Waibel et al., 2018] Waibel, C., Wortmann, T., Evins, R., and Carmeliet, J. (2018). Building Energy Optimization: An Extensive Benchmark of Global Search Algorithms. *Energy and Buildings*, under revi(October).

[Wortmann, 2017] Wortmann, T. (2017). Opossum. *Proceedings of the 22nd International Conference of the Association for Computer-Aided Architectural Design Research in Asia*, pages 283–292.

[Wortmann et al., 2015] Wortmann, T., Costa, A., Nannicini, G., and Schroepfer, T. (2015). Advantages of surrogate models for architectural design optimization. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 29(04):471–481.

[Wortmann and Nannicini, 2016] Wortmann, T. and Nannicini, G. (2016). Black-box optimization methods for architectural design. (April):177–186.

[Wortmann and Nannicini, 2017] Wortmann, T. and Nannicini, G. (2017). Introduction to Architectural Design Optimization. 125(December).

[Wortmann et al., 2017] Wortmann, T., Waibel, C., Nannicini, G., Evins, R., Schroepfer, T., and Carmeliet, J. (2017). Are Genetic Algorithms Really the Best Choice for Building Energy Optimization? (June):51–58.