From Design to Optimized Design

An algorithmic-based approach

Catarina Belém[1], António Leitão[2]
[1,2]INESC-ID/Instituto Superior Técnico, Universidade de Lisboa
[1,2]{catarina.belem|antonio.menezes.leitao}@tecnico.ulisboa.pt

Stringent requirements of efficiency and sustainability lead to the demand for buildings that have good performance regarding different criteria, such as cost, lighting, thermal, and structural, among others. Optimization can be used to ensure that such requirements are met. In order to optimize a design, it is necessary to generate different variations of the design, and to evaluate each variation regarding the intended criteria. Currently available design and evaluation tools often demand manual and time-consuming interventions, thus limiting design variations, and causing architects to completely avoid optimization or to postpone it to later stages of the design, when its benefits are diminished. To address these limitations, we propose Algorithmic Optimization, an algorithmic-based approach that combines an algorithmic description of building designs with automated simulation processes and with optimization processes. We test our approach on a daylighting optimization case study and we benchmark different optimization methods. Our results show that the proposed workflow allows to exclude manual interventions from the optimization process, thus enabling its automation. Moreover, the proposed workflow is able to support the architect in the choice of the optimization method, as it enables him to easily switch between different optimization methods.

Keywords: *Algorithmic Design, Algorithmic Analysis, Algorithmic Optimization, Lighting optimization, Black-Box optimization*

INTRODUCTION

Since the creation of the digital computer, architecture adopted computer science tools and techniques to change its own practices. This lead to the dissemination of digital modelling tools, which simplified the design of highly complex buildings. However, these days, it is not enough to have a well-designed building, it is also necessary to ensure that it has a good performance at various levels, such as, thermal, lighting, structural, and environmental, among others. To this end, several simulation-based analysis tools were developed, which take a specialized model of a building (designated analytical model) and simulate its behavior regarding the intended metric. However, this process has several problems: (1) despite the existence of tools that attempt to convert a 3D model

into its corresponding analytical models, this conversion is frequently fragile and can cause loss of information or errors; (2) hand-made analytical models might be a more faithful representation of the original model but they require a considerable amount of effort to create; (3) ideally, the analysis' results would afterwards be used to guide changes in the original design, but this requires additional time and effort to implement, as also does redoing the analysis to confirm the improvements. This explains why performance analysis is typically postponed to the later stages of the design process, only to verify if the performance meets the standard requirements. Unfortunately, by following such process, it becomes very difficult to optimize a design, which, nowadays, has become an important requirement for ensuring the design of efficient and sustainable buildings.

In order to fully support optimization processes, several changes should be implemented in the design process. First, we need to be able to quickly change a design and, almost simultaneously, produce its corresponding analytical model. Secondly, we need to automate the analysis process and to use it as an objective function for an optimization algorithm.

In this paper, we discuss and evaluate these changes in the context of a lighting optimization problem, proposing a more efficient and flexible workflow for optimization processes

Algorithmic Design

As mentioned, the implementation of changes to a design should be simplified, and, in fact, one way of speeding up design changes is to use parametric models. These models have parameters representing degrees of freedom in the design, which the architect is willing to manipulate in the search for a better performing design. A particularly good approach to produce parametric models is through Algorithmic Design (AD) (Terzidis 2006). Here, instead of directly creating a 3D model in a CAD or BIM tool, the architect creates a program that can be executed with different values of its parameters to produce the corresponding 3D models. This considerably improves the implementation of design changes, allowing for a broader exploration of the design space.

Algorithmic Analysis

Similarly to AD, in Algorithmic Analysis (AA), the analytical model is produced by an algorithm, instead of being created by hand or by relying on fragile conversion tools (Leitão et al. 2017). The novelty is that this algorithm is the same that is used for AD, but configured differently, so as to match the requirements of the analysis tools being used. For example, for lighting analysis tools, a truss might be represented by its surfaces, while for structural analysis tools, it might be represented by a graph of nodes and edges. In either case, the same algorithm is capable of generating different models for different types of analysis.

Moreover, AA is also concerned with the automation of the whole analysis process. In practice, this means that not only is the generation of the analytical model automated, but so is the setup of the analysis tool and the collection of its results.

Algorithmic Optimization

With the ability to quickly update a design, to generate the corresponding analytical model, and, finally, to automatically evaluate the design in an analytical tool, it becomes possible to implement automated optimization processes. We name these processes Algorithmic Optimization (AO) and they are the main focus of this paper.

AO treats the analyses' results for different variations of the design as the functions to optimize. These functions, also known as objective functions, have a domain which corresponds to the range of acceptable designs as specified by the architect. Moreover, since we do not know their mathematical form, these objective functions are often treated as black-boxes, and, as a result, information about the their derivatives cannot be extracted. For this reason, methods depending on function derivatives cannot be used to address this problem. Instead, we use black-box (or derivative-free) methods, which can be divided into three distinct subclasses: direct search

methods, model-based methods, and metaheuristics (Koziel and Leisson 2011; Wortmann and Nannicini 2016; Wortmann et al. 2017).

Direct Search Methods. Although there seems to be no precise definition for direct search methods (Kolda et al. 2003), they are often identified as methods that iteratively: (1) evaluate a finite sequence of candidate solutions, proposed by a simple deterministic strategy; and (2) select the best solution obtained up to that time. They are regarded as valuable tools to address complex optimization problems, not only because most of them were proved to rely on solid mathematical principles (Kolda et al. 2003), but also due to their good performance at initial stages of the search process (Rios and Sahinidis 2013; Wortmann and Nannicini 2016).

The main limitations of these methods are, on the one hand, the performance deterioration with the increase on the number of input variables and, on the other, the slow asymptotic convergence rates as these methods approach the optimal solution (Kolda et al. 2003).

Metaheuristics Methods. Metaheuristics (Glover and Kochenberger 2003) are methods that employ simple mechanisms, called heuristics, to locate good solutions in complex design spaces, while considering the trade-off among precision, quality, and computational effort of the solutions. These methods often rely on randomization, and biological or physical analogies to perform robust searches and to escape local optima. Additionally, through these heuristics, the designer is able to increase the overall performance by adding domain-specific knowledge. Moreover, their non-deterministic and inexact nature confer them the ability to effortlessly handle complex and irregular objective functions (Wortmann et al. 2017).

Nevertheless, metaheuristics are only effective when provided with large numbers of function evaluations (Conn et al. 2009). However, in the architectural practice, it is often the case that a small evaluation budget is available, hence lessening the convergence and performance guarantees (Hasançebi et al. 2009).

Model-based Methods. Model-based methods are known to be very efficient methods to approach time-consuming processes, allowing to reduce the computation time. These methods are able to provide instant estimates of the design's performance, by supplementing or replacing the original objective function by its approximation (Wortmann and Nannicini 2016). This approximation (or surrogate) model is generated from a set of known objective function values, and is then used to determine the promising candidate solution to evaluate next. The result of this candidate solution is then used to improve the surrogate and this process is repeated until a stopping condition is satisfied (Koziel and Leisson 2011).

Additionally, black-box methods might be classified as local or global (Nocedal and Wright 2011). Local algorithms seek a point in a feasible region for which the corresponding objective value is higher than the value of any other point in its vicinities. This point, known as local optimum, is not guaranteed to be the the globally optimal, i.e., the best of all locally optimal solutions in the whole design space. Since these methods look for optima within neighbouring regions, the initial point of the search may be crucial to find the global optimum, i.e., if the initial point is closer to a region containing a global optimum, then the algorithm is likely to converge to that global point, otherwise it might be trapped in a region with a local optimum. Global methods avoid this entrapment by exploring a broader area of the search space before yielding a solution, to ensure they are not caught in locally optimal solutions.

In this paper, we propose to complement AD with the algorithmic evaluation and optimization of the design's performance. As a case study, we evaluated our proposal in the context of lighting performance. To this end, we combined different optimization methods available in existing software libraries, which allowed us to effortlessly test different algorithms. Moreover, we used the AO approach to compare the performance of ten black-box methods, covering both local and global methods of the previously

mentioned subclasses. Additionally, we tested multiple variants of some of the methods to assess their influence on the quality of the obtained solutions. At the same time, we compared the impact of hot-starting local methods in their performance.

METHODOLOGY

Direct Search Methods. We tested the implementation of four direct search methods from the NLopt library ([1]- Johnson 2009). Three of which are local optimization method, namely, SUBPLEX (Rowan 1990), PRAXIS (Brent 1972), and Nelder-Mead Simplex (NMS) (Nelder and Mead 1964), and one global method, DIRECT (Jones et al. 1993).

SUBPLEX subdivides the design space into subspaces to overcome the difficulties of NMS with respect to high-dimensional problems. SUBPLEX iteratively selects a set of subspaces to move to and to seek for a better solution. These subspaces are explored by NMS.

PRAXIS moves from one region to other by iteratively applying line search algorithms to each direction. The goal of these line search algorithms is to determine a better solution along a certain dimension.

NMS constructs a geometric figure, known as simplex, to envelope a region of the design space. After creating it, the algorithm moves the geometric figure across the design space by iteratively changing its vertices.

DIRECT is an algorithm that recursively subdivides the design space into smaller multi-dimensional hyper-rectangles, estimating the quality value of each rectangle. DIRECT uses these values to focus the search on more promising regions of the design space and to further subdivide those in smaller hyper-rectangles. Besides DIRECT, we also test a local variant -DIRECT-L- claimed to be more efficient for functions with few local optima (Glabonsky and Kelley 2001).

Metaheuristics Methods. We tested the implementation of three global metaheuristics methods, namely, the CSR2, ESCH, and ISRES. All these methods are available in the NLopt library as well.

CRS2 (Price 1983) is a metaheuristic algorithm that starts with a random set of designs and randomly applies heuristic rules to the solutions in the set.

ESCH (Santos 2010) creates an initial population that is then iteratively recombined according to a specific distribution function, instead of the uniform distribution, and then mutated. The mutations are the result of combining two different mutation operators -the gaussian variation and the duplication gene mutations.

ISRES (Runarsson and Yao 2005) is a global evolution strategy method, i.e., a method that maximizes the suitability of the candidate designs given by a fitness ranking function. The candidate designs are evolved iteratively by combining the application of a mutation rule and differential variation.

Model-based Methods. We tested two implementations of global model-based methods available in the pySOT open-source library ([2]- Eriksson et al. 2015). In both methods, the surrogate model is generated from nine initially sampled design candidates, and then explored by a perturbation strategy to determine which candidate to evaluate next. The selected methods are based on Radial Basis Functions (RBF) (Gutmann 2001) and the Gaussian Process Regression (GPR) (Rasmussen and Williams 2006). Additionally, we tested a linear interpolation for RBF (RBF-LL), and cubic interpolations for RBF (RBF-CL, RBF-CC).

On the other hand, from the NLopt library, we selected two local model-based implementations, i.e., methods that rely on the construction of simple, partial models of the objective function (Koziel and Leisson 2011): COBYLA (Powell 1994) and BOBYQA (Powell 2009). COBYLA uses the concept of simplex to iteratively generate linear approximations of the objective function, whereas BOBYQA generates quadratic approximations instead.

Lighting Optimization Problem

We combined the three previously referred approaches -AD, AA, and AO- to address the optimization of lighting conditions of a room in an isolated pri-

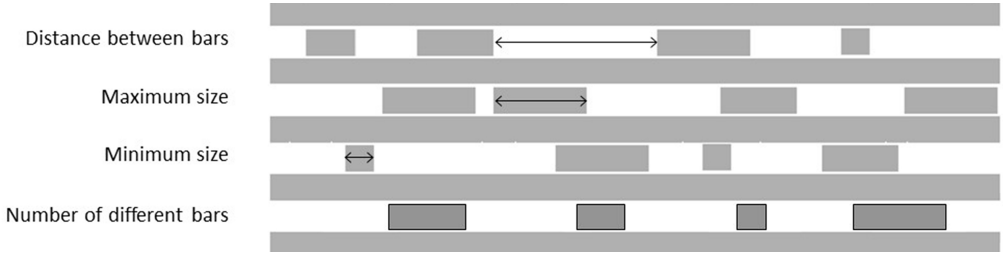


Figure 1
Representation of the shading panels' geometric pattern and its design variables.

vate house in Portugal, previously addressed by Caetano et al. (2018).

The room was designed with a set of façade shading panels that modulates the daylight conditions on the interior. The panels were composed of a set of horizontal wood bars of different sizes (see Figure 1), which alternated between one full-length bar and a set of smaller bars. For aesthetic reasons we randomized the size and the position of the small bars along the panel's width. These bars were restricted by their minimum and maximum lengths, the length's step, and the maximum distance separating two consecutive bars.

To evaluate the quality of the lighting conditions inside the room, we used Radiance, a ray tracing software that enables accurate and physically valid lighting and daylighting simulations (Ward et al. 1998), and we measured the performance results in terms of the spatial Useful Daylight Illumination (sUDI) (Nabil and Mardaljevic 2006), which we then optimized. Figure 2 represents some variations of our case study with different sUDI values.

BENCHMARKING

For this particular problem we subdivided the tests in two phases: (1) we compared both global and local model methods' performance; and (2) we analyzed the impact of the starting point's choice in the overall performance of local methods. The impact of the starting point was tested by providing each algorithm with two initial guesses: both a good and a bad solution, i.e., with values of sUDI of 78% and 7%, respectively.

The performance was accessed according to the time necessary to reach high-quality solutions, i.e., solution with sUDI values greater than 95%. According to the current practice (Cichocka et. al. 2017), we have considered the methods within different time frames, hence evidencing the strength of each method.

To simulate the time constraints and the lack of knowledge about the optimization methods, we used the default parameters for all methods and we considered the number of evaluations to be 60 and 15 for the first and second phases, respectively. For the second phase of tests, we further constrained the number of evaluations to simulate a real scenario, where the designer runs a few evaluations to determine a reasonable solution, which he then uses to hot-start the local method.

After running the first phase of tests, we chose a design with a reasonable sUDI value and used it as the starting point for the local optimization methods. All tests were run on an Intel *Xeon* CPU ES-2670 with 2.60 GHz where each evaluation takes approximately 7 minutes.

RESULTS AND DISCUSSION

This section presents and discusses the quantitative results for the conducted tests.

Test phase 1 - Global and local methods

The obtained results for 60 function evaluations reveal that NMS, CRS2, and all global model-based methods, except RBF-CL, are the most promising methods, achieving sUDI values above 95%, while

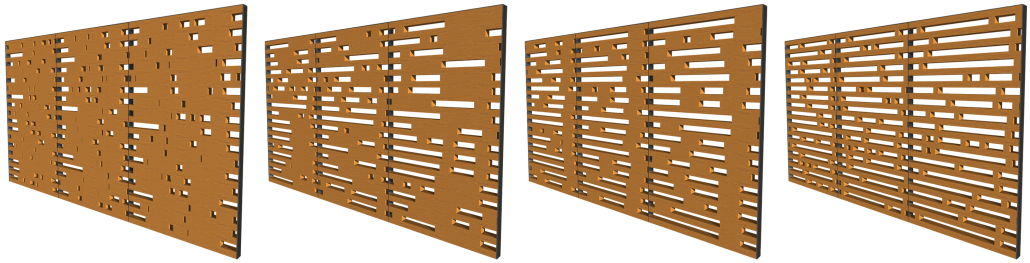


Figure 2
Representation of shading panels' geometric pattern with different sUDI values (from left to right, 7%, 62%, 90% and 100%).

the least promising methods are PRAXIS, COBYLA and BOBYQA, achieving sUDI values inferior to 75%.

Figure 3 shows that, even though some methods have performed poorly throughout the entire process, they still managed to improve the initial design. Also, it is possible to observe that, in general, global model-based methods exhibit very good performance in the early phases of the optimization process, immediately converging to near optimal solutions. However, after a few iterations, CRS2, a metaheuristic method, temporarily outperforms model-based methods, achieving a sUDI value of 98%.

Additionally, the obtained results seem to evidence the sensitivity of local methods to the presence of local optima, as both direct search and model-based local methods converged to sUDI values inferior to 80% after approximately twenty evaluations. Curiously, SUBPLEX, an improved version of the NMS method that is capable of handling higher-dimensional problems more effectively, performed poorly throughout the whole process, increasing the initial value of sUDI by only 9%. In contrast, NMS confirmed to be a very efficient option to address problems with few dimensions (Rowan 1990), having found a design with a sUDI value of 99% after eighteen iterations.

Figure 4 demonstrates the improvements of each optimization method throughout the time of execution. The performance of all methods seems to stagnate after two hours. GPR clearly outperforms

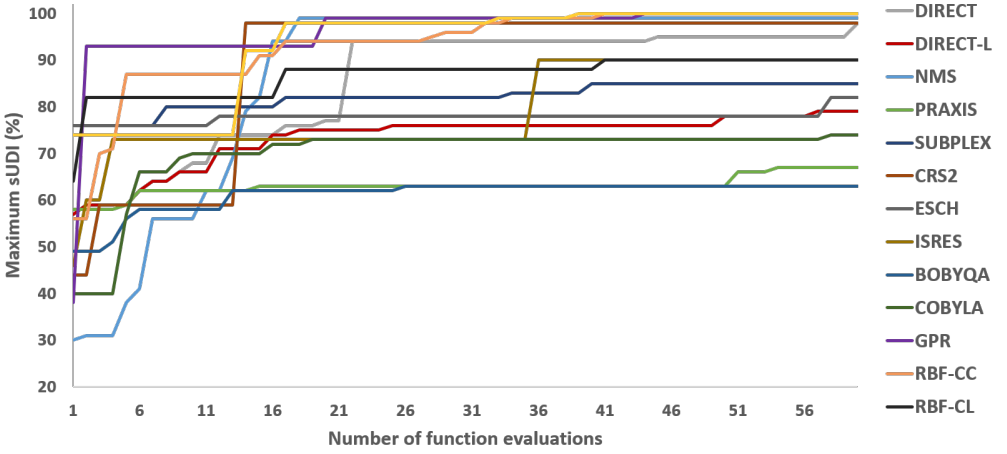
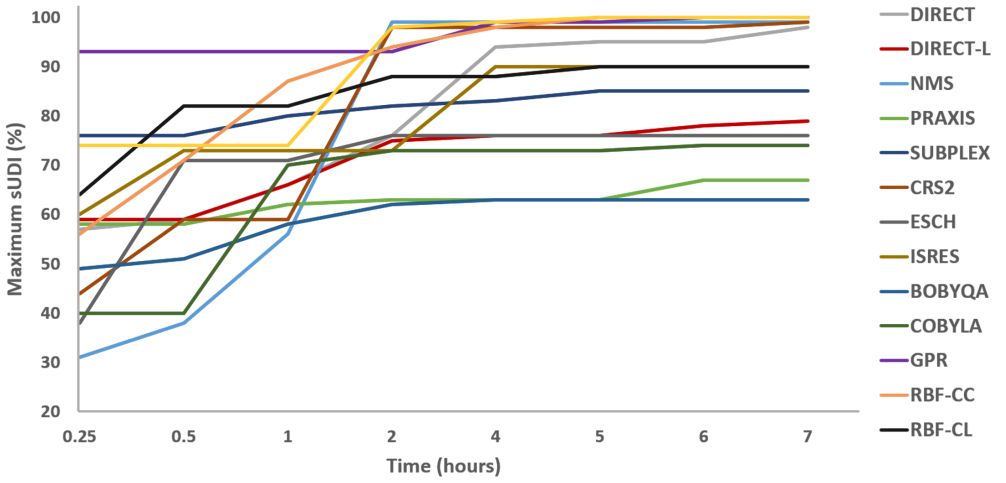


Figure 3
Test phase 1- Maximum spatial Useful Daylight Illuminance (sUDI) as a function of the time in evaluations.

Figure 4
Test phase 1- Maximum spatial Useful Daylight Illuminance (sUDI) as a function of the time in hours.



all other methods for the first two hours, after which CRS2, NMS, and RBF-LL yield better designs achieving values of sUDI of 98%, 99%, and 98%, respectively. For time frames of one hour, RBF-CL and RBF-CC are approaches to consider, producing designs with values of sUDI of 82% and 87%, respectively.

After seven hours, DIRECT achieves its best design with a sUDI value of 98%, while its locally biased variant, DIRECT-L, achieves a mild value of 78%, hence emphasizing the sensitivity of local searches in optimization processes. The metaheuristic ESCH exhibits a similar performance to DIRECT-L, whereas ISRES finds a reasonable design with a value of sUDI of 90% after four hours.

Test phase 2 - Hot-starting local methods

Figure 5 exhibits the results of running five local algorithms for 15 function evaluations. On the one hand, when provided with a mild initial design, both COBYLA and NMS found the best designs achieving a sUDI value of 99%. On the contrary, PRAXIS found the worse, and showed no relevant improvement over the initial design in terms of daylight illuminance. Nevertheless, it initially managed to outperform other methods, achieving values of sUDI of 80%. After the eighth iteration, COBYLA and NMS quickly converged to near optimal designs, with sUDI values of 99% and 98% respectively. BOBYQA and SUBPLEX struggled to improve from the initial design.

On the other hand, when provided with a bad initial design, the best daylight illuminance result has a sUDI value of 15% and was found by NMS after the ninth iteration. NMS, COBYLA, and SUBPLEX exhibit similar performances, stagnating in a design with a sUDI value of 11% after three iterations, with NMS being able to further improve the design after five iterations. PRAXIS exhibits the worst performance among all methods, showing no significant improvements throughout the whole optimization process.

CONCLUSIONS

In the past, Caetano et al. (2018) optimized the room's lighting conditions by consecutively experimenting with multiple designs. However, that mechanism lacks rigor and flexibility, as it provides no guarantee that an optimal solution will be found in

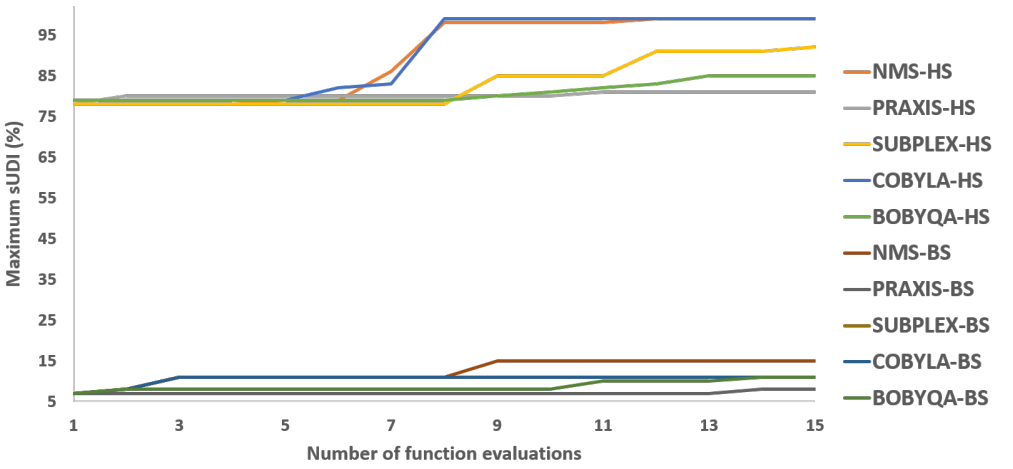


Figure 5
Test phase 2- Maximum spatial Useful Daylight Illuminance (sUDI) as a function of the number of function evaluations. Hot-start (HS) - methods that start with a mild sUDI value (78%). Bad-start (BS) - methods that start with a bad sUDI value (7%).

the least amount of time possible. Instead, an algorithm capable of directing the search towards the global optima should be used. In this paper we proposed to improve the AD process by extending it to include the algorithmic assessment and optimization of the design's performance. This allows to quickly generate different types of analytical models, which are then inputted to different simulation engines, thus enabling the application of optimization processes regarding different aspects, such as structural and lighting, among others.

As it has been previously stated by Wortmann and Nannicini (2016), the selection of the most effective method is important to allow an efficient design exploration when complex and time-consuming simulations are necessary. Consequently, the choice of the optimization method should be based on the results of several tests with different methods for a fixed number of evaluations or a fixed amount of time (Hamdy et al. 2016).

Considering the need to test multiple optimization methods, this work applied AO, which allowed us to easily combine optimization techniques available in existing software libraries, such as NLopt and pySOT. We were, thus, able to effortlessly test different methods, and assess their performance in terms of the trade-off between performance and time. Additionally, we concluded that it is possible to further improve local methods' performance by hot-starting them with the knowledge we have about the problem (i.e., by providing them with an initial good design): in our particular case, if we initialize them with a design that has higher values of sUDI, it is more likely that these algorithms will show better performance.

In this paper, two test phases were conducted. The first phase considered both global and local methods, for which a limit of 60 function evaluations and default parameters were used. The GPR and RBF-CC exhibited good performance throughout all the optimization process, both achieving the maximum sUDI values after forty evaluations. When constrained by hourly time frames, CRS2 becomes the best method to apply, converging to a value of sUDI of 98% in less than an hour. In general, all local methods struggled, except for NMS that quickly converged to 99% after twenty iterations. The second phase tested the impact of the starting points on

local methods by running each local method twice, providing them with two different starting points. The impact of the starting point on the methods' performance is evident, with NMS and COBYLA achieving the best results. In both cases, BOBYQA and PRAXIS struggled to improve the sUDI value of the initial design. Surprisingly, and unlike previous benchmarks (Wortmann et al. 2016), SUBPLEX did not excel at any of the tests, nor did DIRECT.

In sum, this work emphasized the importance of testing different methods in the initial stages of the design, specially when facing tight time constraints. Depending on the characteristics of the problem, certain methods exhibit better performance than others, and it might be advantageous to dedicate a small amount of the evaluations to determine which algorithm performs best. To this end, AO reveals itself as a flexible and effective approach with several advantages for architectural optimization problems.

FUTURE WORK

Future research should consider the application of the proposed algorithmic approach to different tests cases (e.g., different performance criteria, different number of variables, different designs). Moreover, to further support benchmark's results, it is necessary to perform more optimization runs per algorithm, which requires a larger number of experiments. In the case of lighting analysis, these experiments are highly time-consuming, hence lessening the number of experiments. Nevertheless, we plan to enlarge the benchmarking by including more test cases, as well as more optimization methods, and, ultimately, incorporate them in an AD tool. Finally, we also plan to extend AO to address other optimization problems, namely, Multi-Objective Optimization.

ACKNOWLEDGMENTS

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013.

REFERENCES

Brent, RP 1973, *Algorithms for Minimization without Derivatives*, Dover Publications, Mineola, New York

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', *Learning, Adapting and Prototyping - Proceedings of the 23rd CAADRIA Conference*, Beijing, China, pp. 129-139

Cichocka, JM, Browne, WN and Rodriguez, E 2017 'Optimization in the Architectural Practice', *Protocols, Flows, and Glitches - Proceedings of the 22nd CAADRIA Conference*, Suzhou, China, p. 387–397

Conn, A, Scheinberg, K and Vicente, L 2009, *Introduction to Derivative-Free Optimization*, PA: Society for Industrial and Applied Mathematics

Gablonsky, JM and Kelley, CT 2001, 'A locally-biased form of the DIRECT algorithm', *Global Optimization*, 21, pp. 27-37

Glover, FW and Kochenberger, GA 2003, *Handbook of Metaheuristics*, Kluwer

Gutmann, HM 2001, 'A Radial Basis Function Method for Global Optimization', *Journal of Global Optimization*, 19(3), p. 201–227

Hamdy, M, Nguyen, A and Hensen, JLM 2016, 'A performance comparison of multi-objective optimization algorithms for solving nearly-zero-energy-building design problems', *Energy and Buildings*, 121, pp. 57-71

Hasançebi, O, Carbas, S, Dogan, E, Erdal, F and Saka, MP 2009, 'Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures', *Computers and Structures*, 87(5-6), pp. 284-302

Jones, DR, Perttunen, CD and Stuckman, BE 1993, 'Lipschitzian optimization without the Lipschitz constant', *Journal of Optimization Theory and Applications*, 79(1), p. 157–181

Kolda, TG, Lewis, RM and Torczon, V 2003, 'Optimization by Direct Search- New Perspectives on Some Classical and Modern Methods', *Society for Industrial and Applied Mathematics*, 45(3), pp. 385-482

Koziel, S, Ciaurri, DE and Leifsson, L 2011, 'Surrogate-Based Methods', in Koziel, S and Yang, X (eds) 2011, *Computational optimization, methods and algorithms*, Springer-Verlag Berlin Heidelberg, pp. 33-59

Leitão, A, Castelo-Branco, R and Cardoso, C 2017 'Algorithmic-based analysis: Design and Analysis in a Multi Back-end Generative Tool', *Protocols, Flows*

and Glitches, Proceedings of the 22nd CAADRIA Conference, Suzhou, China, p. 137–146
Nabil, A and Mardaljevic, J 2006, 'Useful daylight illuminances: A replacement for daylight factors', *Energy and Buildings*, 38(7), p. 905–913
Nelder, J and Mead, R 1964, 'A Simplex Method for Function Minimization', *The Computer Journal*, 7(4), p. 308–313
Nocedal, J and Wright, SJ 2011, *Numerical Optimization*, Springer-Verlag New York
Powell, MJD 2009 'The BOBYQA algorithm for bound constrained optimization without derivatives', *Centre for Mathematical Sciences, University of Cambridge*, Cambridge
Powell, MJD 1994, 'A Direct Search Optimization Method that Models the Objective and Constraint Functions by Linear Interpolation', in Gomez, S and Hennart, JP (eds) 1994, *Advances in Optimization and Numerical Analysis*, Springer-Verlag Netherlands, p. 51–67
Price, WL 1983, 'Global optimization by controlled random search', *Journal of Optimization Theory and Applications*, 40(3), p. 333–348
Rasmussen, CE and Williams, CKI 2006, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge
Rios, LM and Sahinidis, NV 2013, 'Derivative-free optimization: A review of algorithms and comparison of software implementations', *Journal of Global Optimization*, 56(3), p. 1247–1293
Rowan, TH 1990, *Functional Stability Analysis of Numerical Algorithms*, Ph.D. Thesis, The University of Texas, Austin
Runarsson, TP and Yao, X 2005, 'Search biases in constrained evolutionary optimization', *IEEE Trans. on Systems, Man, and Cybernetics Part C: Applications and Reviews*, 35(2), pp. 233-243
Santos, CHS 2010, *Parallel and Bio-Inspired Computing Applied to Analyze Microwave and Photonic Metamaterial Structures*, Ph.D. Thesis, University of Campinas
Terzidis, K 2006, *Algorithmic Architecture*, Architectural Press, Oxon and New York
Ward, JG 1994 'The RADIANCE lighting simulation and rendering system', *SIGGRAPH '94 Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pp. 459-472
Wortmann, T and Nannicini, G 2016 'Black-box optimization methods for architectural design', *Living Systems and Micro-Utopias: Towards Continuous Designing, Proceedings of the 21st International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2016)*, Melbourne, pp. 777-786
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?', *Symposium on Simulation for Architecture and Urban Design*, p. 51–58
[1] http://ab-initio.mit.edu/nlopt
[2] www.github.com/dme65/pySOT