Learning to be a Vault

Implementing learning strategies for design exploration in inter-scalar systems

David Stasiuk[1], Mette Ramsgaard Thomsen[2]
[1,2]The Royal Danish Academy of Fine Arts, Schools of Architecture, Design and Conservation (KADK) Centre for Information Technology and Architecture (CITA)
[1,2]http://cita.karch.dk/
[1,2]{david.stasiuk|Mette.Thomsen}@kadk.dk

Parametric design models enable the production of dynamic form, responsive material assemblies, and numerically and geometrically analytical feedback. The value potential for design produced through the procedural transformation of input parameters (or features) through algorithmic models has been repeatedly demonstrated and epistemically refined. However, despite their capacity to improve productivity and iteration, parametric models are nonetheless prone to inflexibility and reduction, both of which obscure processes of invention and discovery that are central to an effective design practice. This paper presents an experimental approach for the application of multiple, parallel computational design modelling strategies which are tested in the production of an inter-scalar model array that synthesises design intent, the simulation of material behaviours, performance-driven adaptation, and open-ended processes of discovery and categorical description. It is particularly focused on the computational potentials embedded in interdependent applications of simulation and machine learning algorithms as generative and descriptive drivers of form, performance, and architectural quality. It ultimately speculates towards an architectural design modelling method that privileges open model topologies and emergent feature production as critical operators in the generation of flexible and adaptive design solutions.

Keywords: *parametric design, computational modelling, machine learning, multi-objective optimisation, k-means clustering*

INTRODUCTION

Although parametric modelling allows designers to dynamically produce variable geometries and execute sophisticated analyses of performance attributes, certain problems of both *rigidity* and *reduction* endemic to such design systems are well documented. Parametric models are limited because they necessitate a predetermination of both model topology and the feature domains that constitute the model's parameter space. As such, the designer either must explicitly define all parameters and relationships between model elements at the start of the design project or risk breaking the model during any ensuing reconfiguration (Davis 2013). This rapid calcification of the design space is antithetic to the experience of design as a process of invention and discovery. A second problem is associated with risks in the oversimplification of descriptive parameters such that the defining design criteria ignore or suppress potentially useful information embedded in the model output. While this implicit reduction preserves design control and makes the creation of the model tractable (Davis 2013), the resulting limitations act as an impediment to a more exploratory design practice. These problems are only compounded by an increasing sophistication in design models' ability to represent in both the production and analysis of a number of useful characteristics, such as material assembly behaviours, occupation patterns, or energy-related performances. An increased facility in designing for and with these considerations is central to continued innovation in design modelling (Tamke et al. 2011). However, this increased capacity also offers the opportunity to explicitly describe the complexity of the design problem at hand, and in response develop open-ended design systems that have the capacity to address them (Cariani 2008). For this reason there remains a significant need for the ongoing development of tools through which we can flexibly capture and understand complex interstitial dependencies across model elements for directed performances and as a means to enhance the pursuit of invention and discovery in the development of design models.

This paper traces the exploration of new methodologies in this pursuit, specifically addressing the related desires to maintain open feature domains for both design and evaluation, in order to alleviate requirements for the pre-configuration of both value and element connectivity. It also describes the related pursuit of simultaneously embedding material behaviours in morphogenesis and producing multiple targets for performance optimisation. The aim for the project is to establish methods for designing with open topologies in which the dependencies between parameters are both emergent and changeable during the design process. To this end, an experimental approach is implemented for multiple computational strategies, applied in parallel and sequential operation in the digital environment.

Background

This project takes its point of departure from an intensive workshop with the Digital Matter Master's studio, led by Areti Markopoulou at the Institute for Advanced Architecture Catalonia (IAAC) in early 2014. This workshop introduced multi-objective evolutionary solvers operating in a simulated material context. Through a series of physical models using simple rattan splines and connector ties, seven teams focused on the development of generative design algorithms that synthesise material behaviours, the topological transformation of connectivity between constituent elements, and quantitative, multi-objective optimisation design goals. Through the exploration of these networks as morphogenetic rule-driven systems for incremental formation, a series of variables available for deployment in a multi-objective evolutionary model were then developed. The set of simple goals that emerged from this process of rapid physical prototyping were related to material usage, the generation of space and structural performance and capacities. Measures related to connectivity and material deflection define those goals related to structural performance, and the coupling of the conflicting goals of minimizing material

use while maximizing envelope size provided simple spatial performance goals. What emerged through these iterations was a series of rigorous pseudo-code algorithms for both generation and evaluation. The distinct approaches of each team established a framework for developing a more general design system capable of generating multiple, highly varied configurations, that nonetheless retain formal and organisational legibility.

Learning to be a Vault

"Learning to be a Vault" is an experiment consisting of a significant digital exploration and consequent representation of the design space, along with the production of 25 1:25 models and a 1:1 demonstrator approximately 5 x 5 x 2.5 m. It has been installed as part of the "What does it mean to make an Experiment" exhibition at the Royal Academy of Fine Arts School of Architecture during the Spring of 2014. The IAAC workshop operates as a baseline for the formulation of this experiment, which focuses on the synthesis of multiple computational strategies implemented in an inter-scalar modelling environment. Of particular interest is the deployment of *machine learning* algorithms. One possible strategy for advancing the flexibility of parametric models as performative instruments is tied to rethinking how parameters (or *features*) are treated in the modelling space. Typically, feature domains are established as inputs for the geometrical outputs of an architectural design model (Davis 2013). However, there are opportunities to recast certain features in a parametric model as either dynamic or descriptive entities, enabling the designer to engage in alternative means to search the design space. For this experiment, the focus is tied to using machine learning algorithms to *classify* large collections of model outputs generated through a multi-objective evolutionary optimisation solver into legible groups - or species - of response.

In "Learning to be a Vault", there is a tight integration of three distinct modelling systems. The first is a generative model, constructed with the considerations developed through the workshop described above. It recursively builds up networks of actively-bent splines that have the capacity to achieve great variety in form. Intrinsic to this model development is not only the topological connectivity established through the generative algorithm, but also a spring-based form-finding simulation of the design material, which is rattan, a soft but highly flexible wood-like plant. The second model is a multi-objective evolutionary optimisation solver, which leverages the generative model input parameters as its *genotypes* and its outputs as its *phenotypes*. These phenotypes are subsequently analysed for five distinct performance measures. Finally, the outputs from this model are analysed using k-means clustering, an unsupervised learning algorithm that identifies intrinsic relationships between the data elements of its input data points. This analysis takes on the form of *classification*, which allows for an intuitive understanding of many outputs as belonging to legible and distinct groups. The following section will briefly discuss machine learning in general, and in the context of computational design models, will then offer focus more on the relevant machine learning strategies used in this particular experiment.

MACHINE LEARNING IN COMPUTATIONAL DESIGN MODELS

Machine learning is a field of research and practice related to developing computer programs that are configured to improve their **performance** at a given **task** through **experience** (the acquisition and processing of incremental data) (Mitchell 1997). It is focused on three primary categories of interest: *task-oriented studies*, *cognitive simulation*, and *theoretical analysis* (Carbonell, Michalski & Mitchell, 1983). For task-oriented studies, machine learning is largely interchangeable with the field of statistics, as many predictive algorithms (such as linear or logistic regressions) are used extensively in both. Task-oriented machine learning is primarily concerned with the classification of data points as function values within some descriptive domain. These domains are most often discrete (e.g. Boolean, integer, or categorical val-

ues) but can also be continuous (e.g. real numbers). Learning models are divided into two main types: supervised and unsupervised.

Supervised learning models rely on training data where the predicted outcomes are known, so that the model develop input (or feature) transformations and weights that will allow it in the future to more effectively predict the outcomes for unknown data. As each new evaluated data point used (or instance) for training has its actual outcome determined, the learning model gains new intelligence about its performance and recalibrates itself according to its driver algorithm. Unsupervised models operate differently. They allow for the features of data sets to self-organize. These are primarily used for the discretisation of data into descriptive or functional categories, without the need for training data. In effect, when the outcomes are unknown, unsupervised learning algorithms are designed to allow for the internal relationships within the features of a data set to create emergent descriptions of each instance. Both supervised and unsupervised learning models are often used discretely to resolve subcomponents of larger problem spaces. In effect, the outcome of any learning model is simply a new feature for application in a new decision space.

Both supervised and unsupervised learning approaches have precedent in architectural design modelling. Their applications have been theorised for nearly as long as CAD systems have been in use, but in the last twenty years - and increasing in step with advances to the processing power of personal computers - their implementation has become tractable and in many cases standardised. The most commonly used learning algorithms in design modelling are *genetic* or *evolutionary* solvers, whose epistemological and functional maturity in the design sciences far outreaches that of other modes. However, several of those other, lesser-used approaches - which include neural networks of multiple types and k-means clustering algorithms - have nonetheless been successfully implemented in the past and present excellent opportunities for the future extension of learning algorithms in design modelling. For the purposes of this experiment, we will focus on two of these: multi-objective evolutionary solvers and the k-means clustering algorithm for classification. Additionally, spring-based particle simulation systems will be examined according to the criteria used for establishing that an algorithm constitutes a learning system.

Multi-Objective Optimisation

Evolutionary algorithms have been applied in the field of architecture for over twenty years, most popularly with the pioneering work of John and Julia Frazer in Department 11 at the Architectural Association, starting in the late 1980's (Frazer 1995). In a rough summary of their operation, they rely on a generative system for the parsing of inputs (genotypes) into outputs (phenotypes), which are then subjected to a performance analysis - most often numerically represented. They begin with a pool of randomly assigned genotypes, test the performance of resulting phenotypes, and then "breed" the most successful offspring through the crossover (and potential) mutation of genotypes. Over time, this process allows for successful genotypes to thrive and pass on their parameters to their children, and is repeated through a number of generations until satisfactory objective values have been achieved by the resulting phenotypes. Because evolutionary solvers rely on setting explicit targets for performance measurement, they can be understood as *supervised* learning models, with the optimisation objectives used for training.

In an architectural design context, there are key considerations for making this approach tractable. There must be a balance struck between the model's ability to produce an artefact that is recognisable as a built object, but also a capacity for the model not overly limit the range of possible outcomes (Janssen et al. 2000). This can be understood as a simultaneous desire for *legibility* and *variety* in the outputs of such models.

As the computational tools available for executing evolutionary algorithms have become more ac-

cessible, so has the capacity to introduce many parallel objectives into a single model. The concept of *n*-dimensional objectives in such design models is keyed around the *Pareto front*, which operates as a means to understand the outcomes of the design space (Figure 1). The Pareto front is defined by a boundary of phenotypes whose optimisation values define a convex hypervolume in the *n* th dimension, with n being the total number of optimisation objectives. For example, single-objective search will result in a point, with a single, optimum phenotype. A two-objective search will result in a convex polyline, and a three-objective search will result in a convex triangulated hull. Each phenotype that constitutes the Pareto front can then be said to be "optimised" in some capacity. The front communicates the trade-offs that exist between phenotypes: as a phenotype demonstrates better optimisation for one objective, it loses value for another. The Pareto front is convenient to visualise these performance trade-offs in up to three dimensions (Caldas 2003), but becomes significantly more difficult to understand once it defines a hypervolume in four or more dimensions. Compounding this difficulty in navigating performance objectives is the fact that as the front's dimensionality increases, it is comprised of a great many more constituent phenotypes, potentially resulting in thousands of Pareto-optimised candidates (Winslow, et al. 2010).

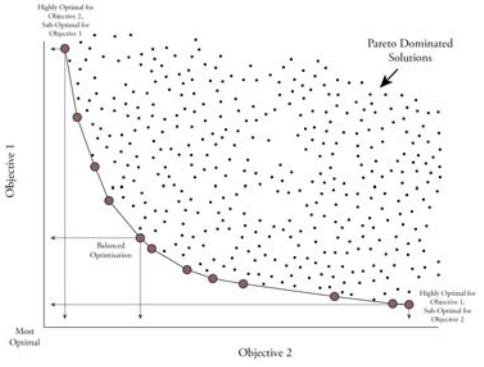


Figure 1
Example of a Pareto front for a two-dimensional objective optimisation

It is, however, this volume of phenotypic output that must be addressed if the continued advances in analytical computational methods are to be synthesised in multi-objective architectural design models. This experiment then deliberately identifies an objective space that produces such a volume of output, and engages in the use of a second learning algorithm to allow for the designer to engage in a form of search that encourages a process of discovery.

K-Means Clustering

K-means clustering - also known as Lloyd's algorithm - is a distance-based, unsupervised learning algorithm that uses intrinsic relationships between data points in large sets to discretise them into clusters containing the most similar instances. K-means clustering achieves this by using the Euclidean distance between data points in *n*-dimensional space (where *n* is the number of features used to drive the clustering algorithm) to find solutions for similarity for a user-specified *k* number of clusters. It does so through an iterative process of computing the "nearest-distance centroid rule", which can be visualised as a voronoi diagram. The algorithm begins by assigning *k*-number of random points in the data as centroids for the diagram. It then evaluates all of the other points in the set, and assigns them to a "cluster" based on the voronoi boundary containment. Then, the average value of all contained data points is used to recalculate the centroid of the voronoi diagram. This process is repeated until the cells achieve a stable state such that data points no longer move between clusters, and thus centroids no longer require recalculation.

As in the case of the Pareto front, this process is easiest to visualise in lower-dimensional space. Figure 2 illustrates a k-means clustering algorithm run on a two-dimensional data set as a collection of XY points. The iterative readjustment of the voronoi bounding space according to subsequent containment tests for data points finally results in cells that describe boundary conditions easily identified by the eye as being optimally discretised.

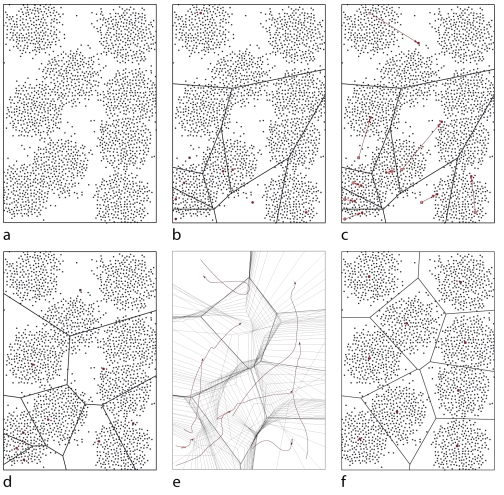


Figure 2
K-means algorithm on a two-dimensional data set with k=9: a) data array, b) boundaries for k random sample points, c) test for inclusion and move centroids, d) redefine boundaires, e) repeat c & d until stability, f) stable solution

K-means clustering is used extensively to find unknown relationships between data points in large data sets. Even though it is difficult to understand euclidean distances within data sets of high dimensionality in the abstract, clusters produced by such analyses - so long as the data being used to structure them possesses useful descriptive capacities - generally makes intuitive sense when examining outcomes. For example, k-means clustering is regularly used in internet search engines for the purpose of clustering similar search elements together, where based on such data elements as key words and the geography of a story's origin, related content can be grouped together that can readily be grasped by an observer.

K-means clustering has been implemented in architectural models as drivers for fabrication rationalisation. A prominent example is the facade panelisation process developed by Gehry Technologies for the Soumaya Museum in Mexico City. This freeform structure's facade is comprised of more than 16,000 panels, each of which, in the original design geometry, is unique. However, by defining a maximum acceptable tolerance for each panel, the team was able to deploy the k-means algorithm to reduce the number of moulds required for fabrication to 49. More generally, the architects and engineers for Evolute software make use of k-means clustering in their plug-ins for facade panel rationalisation.

For this experiment, k-means is not used to rationalise elements for fabrication, but in an entirely different capacity: to effectively understand, search, and *discover* the complex and varied design outputs that high-dimensional multi-objective optimisation algorithms produce.

Simulation Systems

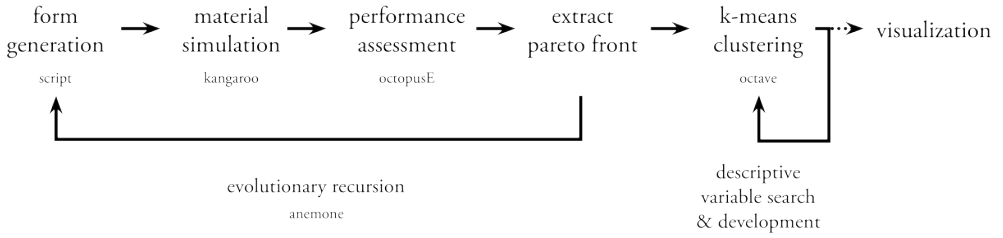
Spring-based particle simulations are frequently used in computational models for form finding and the integration of material behaviours into the design process. They use the same principles as agent-based models, with individual particles acting as agents, influencing each other according to definitions of connectivity and a rules that define the consequence of interaction. Interestingly, they meet the explicit conditions that Mitchell details in his definition of a machine learning program: through the experience of their interaction with one another over time, simulation particles improve their performance at the task of describing material or behavioural consequences for their constituent elements.

Extensive research has been made regarding swarm intelligence and the effectiveness of encoding material capacities into simulation-based design models (Thomsen et al. 2010), but it is worth extending this frame of reference to consider the parallels that exist between such design systems and machine learning algorithms. This experiment relies on the use of material simulation for both form-finding and the production of evaluation objectives for the evolutionary algorithm.

METHOD

"Learning to be a Vault" (Figure 3) is set up to synthesise the modelling systems and considerations described above, including 1) a morphogenetic script capable of producing a wide variety of forms grounded in a tendency toward legibility, 2) a spring-

Figure 3
Process diagram for "Learning to be a Vault"



based particle simulation for form-finding and loading behaviour of the actively-bent rattan member, 3) a multi-objective evolutionary solver for optimising the solution set and producing results on the Pareto front, and 4) a k-means clustering algorithm for searching the design space. The evaluation criteria for the experiment lies in the effectiveness of this integration, and the legibility, searchability and meaning inscribed by the final clusters.

Generation and Optimisation

The morphogenetic script implements a recursive array of elements. There are two types of elements considered: *primary* members that attach both ends to the floor in an elastic arch, and *secondary* members that span between two primary members. Secondary members connect to primary members in two consecutive locations such that active-bending forces are interdependently shared between them. A number of transformations are available to primary and secondary elements through each recursion such that many forms of varying symmetry and spatial consequence are producible (Figure 5). Once formed, each collection is passed through a spring-based simulation engine (the Kangaroo plug-in for Grasshopper) to execute both form finding, and deflection under self-loading. Next, using a multi-objective optimisation solver (the Octopus plug-in), each collection is evaluated for suitability for continued breeding according to the following objectives: 1) minimal deflection, 2) restriction of tight radii, 3) targeted area covered, 4) targeted connectivity, and 5) targeted height for secondary members. Subsequent generations are then recursively passed through the same generative and evaluative process until suitable results have been produced.

Searching the Design Space

Based on this collection of objectives, over 80 generations more than 2000 unique Pareto optimised phenotypes are produced (Figure 4, 5). These phenotypes are then analysed according to series of descriptive paremeters developed by the designer. In this case, the descriptive parameters applied are a collection of numerical transformations of phenotype geometry. These are then passed through the software Octave (a numerical solver capable of rapidly executing complex machine learning algorithms) for computing the k-means algorithm and back into Grasshopper for visualisation and consideration of results.

Through an iterative implementation of this process, the designer uses both heuristics and intuition to identify further possible transformations to the data that might result in improved clustering differentiation. In this case, 18 distinct numerically descriptive variables were developed, including: total materials used, average variance in height of secondary members, the radius of a circle inscribed in each primary member's start point, and the same for end points, and the distance variance between primary member base points. On their own, such parameters appear banal. However, when parsed through the clustering algorithm, they combine with

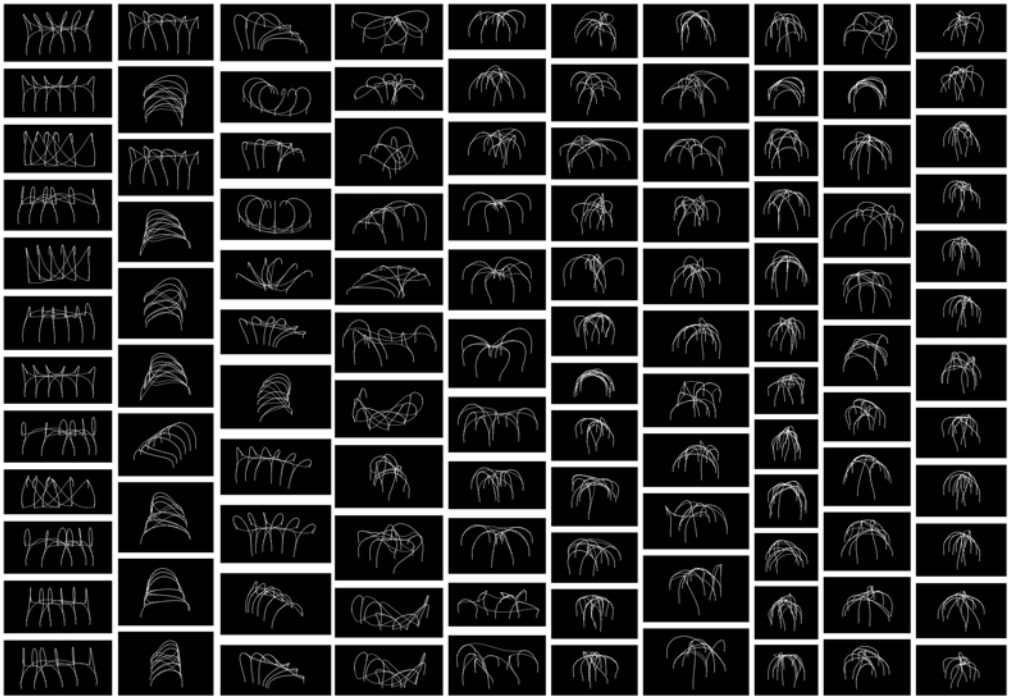


Figure 4
Partial section of ten clusters (out of 80, representing over 2000 total phenotypes)

one another to discretise a seemingly fragmented and intractably varied set of solutions into a collection of legible clusters.

It is here a new process of discovery emerges. Through the invention and application of these descriptive variables, one is able to steer the intrinsic relationships between relatively simple aspects of the geometry's underlying data structures toward a coherent and easily searched design space. This process of iteration is critical and effectively operates as a second stage for design execution. The clustering of suitably large collections of phenotypes - particularly those that have been optimised for a variety of objectives - may ultimately result in a form of speciation, wherein the designer is able to identify highly distinct solutions that may be suitable for diverse architectural applications.

The development of effective data visualisation tools is essential for managing the information about clusters that is produced. One measure for determin-

Figure 5
selected 1:25 models

ing the efficacy of a clustering analysis is the average *variance* for all variables of each constituent data point relative to its cluster centroid. Here, a smaller variance indicates clusters that better describe their constituent data points. As such, a zero variance requires that there be exactly one cluster per data point. This would offer no simplication, and therefore the purpose of the process is for the designer to establish a threshold that effectively finds the least (and most searchable) number of clusters that retain a high level of descriptive capacity. For this experiment, a dashboard is developed that allows for the designer to rapidly understand the dynamics of different clustering *passes*, with each incremental pass adding another descriptive cluster. The dashboard indicates how adding numbers of clusters from one pass to the next decreases variance for the entire solution, but does so at a decreasing rate, and at the expense of searchability. To facilitate finding an effective threshold, it allows for the designer to select individual clusters, see the individual phenotype most representative of the cluster centroid, read the relative average values of the descriptors used for executing the algorithm, and see which clusters are similar to it both within the same clustering pass and in those both preceding and subsequent to it (Figure 6).

DISCUSSION

Increased computing power and a proliferation of tools make more advanced multi-objective optimisation design models more tractable and create a need for new methods for managing phenotype outputs emerges. Attendant to having more complex collections of objectives is a higher volume of outputs. So not only does this make it difficult to visualise these outcomes on the Pareto front, but the sheer number of phenotypes to be processed makes a case-by-case review difficult. This experiment represents a first look at applying unsupervised learning algorithms for the purpose of organising such high-dimensional data output into coherent segments and as a result enabling the designer to engage in a secondary operation of discovery through the development of descriptive parameters that steer this segmentation process.

There are a number of opportunities to further refine and enhance the utility of the approach presented here. First of all, applying the clustering algorithm between generations in the evolutionary solver would enable the designer to implement a form of directed breeding, with non-desirable clusters removed from the pool of available solutions. This would more tightly couple the different con-

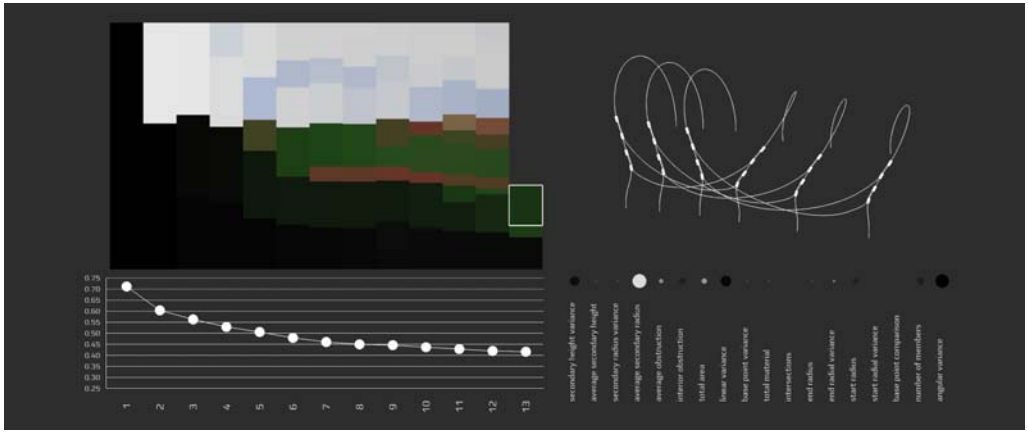


Figure 6
dashboard examining a single cluster, and its orientation within a solution set containing several passes of increasing cluster counts

tributing model elements and lend a higher degree of control to the designer. Also, there are multiple approaches for numerically processing descriptive features to heighten their ability to produce better distinguished clusters, as well as for identifying thresholds for ideal number of clusters for a given data set. These approaches can be further explored and implemented.

Conclusions

Machine learning algorithms present a number of opportunities for the enhancement of computational design practice. There are varying degrees of maturity between different methods currently applied. For example, while the use of evolutionary algorithms in the design sciences has an established history and advanced epistemology, algorithms such as k-means clustering have had more limited application. Neural networks of multiple type have been deployed. "Growing Neural Gas" networks have been implemented for the development of use and occupation analysis in urban environments, and have been deployed for developing way-finding analyses for space plans at multiple scales (Langley et al. 2007). Backward propagating neural networks have been used to drive robotic instruction for resilient structural systems (Mehanna 2013). Computational designers have long applied spring-based particle and agent-based simulation systems in their models, which can be seen to exhibit functional equivalence to machine learning models, but for problems particularly well-suited to the design sciences.

Machine learning models designed to handle the most difficult problems are often conceived of and constructed as a complex of multiple distinct approaches, operating in some combination of parallel and sequential synthesis (Flach 2012). In addition to the operational benefit of using machine learning algorithms directly to enhance the design capacities of their models, such an outcome should be compounded through the conception of their models as inter-scalar ecologies of such knowledge-acquiring algorithms.

REFERENCES

Caldas, L.G. 2003 'Shape Generation Using Pareto Genetic Algorithms Integrating Conflicting Design Objectives in Low-Energy Architecture', *Proceedings of CAADRIA 2003*

Carbonell, J.G., Michalski, R.S. and Mitchell, T.M. 1983, 'Machine learning: a historical and methodological analysis', *AI Magazine*, 4.3, pp. 69-79

Cariani, P. 2008, 'Design Strategies for Open-Ended Evolution', *Artificial Life*, XI, pp. 94-101

Davis, D. 2013, *Modelled on Software Engineering: Flexible Parametric Models in the Practice of Architecture*, Ph.D. Thesis, RMIT

Flach, P. 2012, *Machine Learning: the Art and Science of Algorithms that Make Sense of Data*, Cambridge University Press

Frazer, J. 1995, *An Evolutionary Architecture*, Architectural Association Publications

Janssen, P., Frazer, J. and Tang, M.X. 2000 'Evolutionary design systems: A conceptual framework for the creation of generative processes', *Proceedings of Design Decision Support Systems in Architecture and Urban Planning 2000*

Langley, P., Derix, C. and Coates, P.S. 2007 'Meta-Cognitive Mappings: Growing Neural Networks for Generative Urbanism', *Proceedings of th 10th Generative Art Conference 2007*

de Leon, A.P. 2012 'Two Case-Studies of Freeform-Facade Rationalization', *Proceedings of eCAADe 2012*

Mehanna, R. 2013 'Resilient Structures through Machine Learning and Evolution', *Proceedings of ACADIA 2013*

Mitchell, T.M. 1997, *Machine Learning*, McGraw-Hill Science/Engineering/Math

Tamke, M., Burry, M., Ayres, P. and Thomsen, M.R. 2011 'Design Environments for Material Performance', *Proceedings of Design Modelling Symposium Berlin 2011*

Thomsen, M.R., Tamke, M. and Riiber, J. 2010, 'Sea Unsea - Lamella Flock', in Leach, N. and Snooks, R. (eds) 2010, *Swarm Intelligence: Architectures of multi-agent Systems*, LATP

Winslow, P., Pellegrino, S. and Sharma, S.B. 2010, 'Multi-objective optimization of free-form grid structures', *Structural and Multidisciplinary Optimization*, 40, pp. 257-269