Student ID Number: I6045016
Student Name: Emily Hazel Jane Sellman
Entry Date to MSC: 2012 September
Paper: Bachelor Thesis
Word Count: 10995 words

Utilising Genetic Algorithms as a Method of Improving Risk Within Aerospace Engineering Design

Author: Ms EHJ Sellman

Internal Supervisor: Dr S.Williams [1]
Research Supervisor: Dr N.Ball [2]

[1]Maastricht Science College, Maastricht University, the Netherlands
[2]Department of Engineering, Cambridge University, United Kingdom.

Contact: ehjs2@cam.ac.uk, or esellman@student.maastrichtuniversity.nl

11th of August 2015

Contents

Abstract

Designing and creating complex projects can be a challenging task, especially when satisfying multiple varied requirements. Systems and their subsystems often interconnect, and aspects of these can depend directly or indirectly upon them, thus creating intricate interfaces. As altering a single element can impact numerous others within a system, it is important to understand which elements carry the most responsibility. This research aims to aid designers in their work by quantifying the amount of risk presented by a single element in a system, and specifying its characteristics. Genetic algorithms were used to find optimal solution by experimenting with parameter values. These produced different solutions which organise the system in such a way that risk is minimised overall.

1 Introduction

Within any industry it is important that a company can create products which are more desirable than those of their competitors. Aerospace engineering is no different[1]. Companies must ensure that their products are the best by establishing a good reputation for safety, efficiency, and cost effectiveness, while satisfying many other requirements and constraints[2]. Naturally, these restrictions are varied and complex; often a single feature of a product may have to serve a variety of different purposes - thus calling for designs whose development reflect this. It is also common that different solutions may conflict, meaning that compromises may be necessary. It is these types of complex multipurpose projects which are subject to interconnecting dependencies, where features of a system can impact other sections of the design[3]. The research focuses upon this matter and looks into the way that risk is controlled within a system. One of the biggest challenges is making changes to the system, and managing the effect this has upon the rest of the system. The aim of this research is to use optimisation techniques to help designers have a more specific and accurate understanding of connectivity and dependency within systems, especially when these can sometimes be hugely complex, intricate, and function on both a large and small scale. This been carried out by implementing genetic algorithms to create a method to optimise the overall simplicity of designs by measuring the different risk factors.

1.1 Key Research questions

This research builds upon the wealth of work which has previously been carried out within this field. However there are sub-sections of research which could potentially be improved upon. The future of this field rests upon improving three identifying areas, these being: *Visualisation*; the assessment of *Risk*; and the available support to designers through *software*[4]. These guiding areas have led this research project to come to the conclusion that the *software* element of the research could be developed further.

- Question one: By using past data for existing projects, how can design advice becomes automated to a higher degree?

  This question is based around the idea of advancing the current support tools for designers in a manner which can be quick to produce by a non-specialist and understood by designers who belong to different specialisms. This is one of the overall goals of the research because it is important to consider how current systems which exist academically can be placed within industry in practice.

- Question two: How can this be carried out in terms of the software? Are the optimisation mechanisms effective?

  The second question is centred around the concept of making the tool generic - enabling easy implementation. However it can be recognised that abstraction is necessary for this task, so to make the current model of risk analysis an intuitive experience for the designers, and would give more specified support. The latter part of this question is based around understanding how valuable this tool is.

Literature overview

This literature review looks to collate and consider prominent and relevant work pertaining not only to the specific application of this project, but also to the mechanisms and techniques used within the research. By reviewing and critiquing past work, this work may establish a setting within academia.

1.1.1 Change Management

Creating changes within systems appear to be simple: as the specific element to alter is identified, the alteration is planned and then executed. However, it is because of the network of dependent structures which make systems so complex, that Maull[5] proposed the engineering change process:

- Organisation of the change to be made
- Solution development
- Evaluation of the solution
- The approval and application of the change
- Implement the change to the system

This approach can occasionally have a further step: the critique of change after implementation. The purpose of which is to review. This style of change management is the basis of this research, because the work is carried out on systems which already exist. However the results of this research can also be applied to the design process of new projects and their changes; so each step of this process is important to consider. Another notable perspective upon change management is given by Fricke[6], who suggests factors for designers to consider:

- Less. This is the approach of avoiding making changes if at all possible.
- Earlier. The idea which states that if an alteration is made early in the process, then it causes the change to cost less. This concept is shown in figure 1.
- Effective. It was noted by Deubzer[7] that within their research, 39% of design alterations could have been avoided. This indicates that with better support tools, this value could be plausibly reduced to 0%.

- Efficient. Any addition to the design process should be easy to implement into the process with minimum disruption.
- Better. Studying past work can give mechanisms to improve those in the future, making each generation an improvement.

These ideals are well established in engineering design theory, especially when considering the influence of industry. The graph in figure 1 presents the consequences of alterations to designs and the point at which these occur during progress.

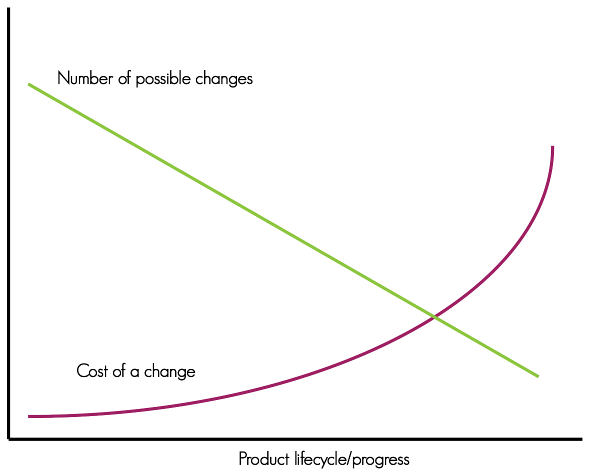


Figure 1: This illustrates the commodities which are at risk (time, money) during the design process

The relationship shown in figure 1 is a way of illustrating the general relationships occurring within engineering.

1.1.2 The Need for Support

Past work carried out by Clarkson and Jarratt [3] has shown that the base approach which this paper builds upon has been proven successful. The degree of success can be measured by the criteria shown below:

- The methodology used is able to create predictions, and these have proved to be correct having been carried out.
- The predictions which have been created, agree with the evaluation of designers

This type of assessment proves that the theoretical advantages exist in practice by implementing this methodology. Thus grounding the importance of the support available to designers.

1.2 Visualisation

1.2.1 The Importance of Visuals

When working upon a complex system it is necessary that many different agents will be involved in the process. This is because large scale projects require multiple varied skill sets, and in complex projects such as aerospace engineering - skills are typically highly specialised and unique[8]. This particular level of expertise tends to be narrow, and thus communication between different sectors of a project can be challenging as some agents understand issues in a way which others do not[9]. For this reason, it is of paramount importance that the communication between sections of projects are as efficient and effective as possible[10]. As an example, it could be important that teams are aware of certain boundaries which can only be seen by one team and not the other due to expertise differences. These types of scenarios limit the interdisciplinary power to overview a large project efficiently[11].

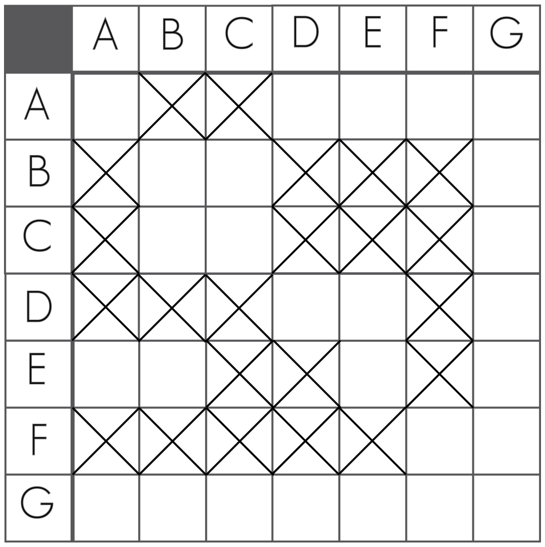
The key to making safe and informed decisions about design and ultimately creating a product which can work to the best of its abilities, is information and its communication[12]. In order to combat these issues, it is important to establish systems which make this process faster, easier, and less prone to expensive and time-consuming mistakes while reducing the need for experimental iterations which could otherwise be avoided. As Bucciarelli recognises, "Each designer comes from a different object world[9]. In order to communicate well, information should be clear, logical, easily grouped into topics for clarity, and well structured[13]. Some types of information can be more challenging to express in words than in visuals, and thus requires a different method to disseminate the information. Visualising information is also an advantage because little or no guidance is required in order to perceive information in a new way, as a brief description can be sufficient. Making communication of intricate topics more efficient, easy, and clear, can reduce the presence of misunderstandings or mistakes. Shai suggests that using graphs as a model for consistently styled communication method within engineering to address this issue[14].

1.2.2 Use in this Research

It is valuable to specify how different areas of a project interact with one another, leading to another level of detail to illustrate. This can be shown in a binary manner or by showing the strength and degree of the connection - which divulges more information. These dependencies can then be used to study a system and understand where the most vital aspects of a design lie, and by how much they are connected to the surrounding systems[15]. Using this process designers can have a clearer understanding and therefore more accurate and detailed information about a system, which ultimately helps them make better decisions[16]. These systems are large and complex, meaning that demonstrating the information textually would be problematic. It is also common that certain styles of visual representation becomes so convoluted that graphical depiction is no longer a viable method to demonstrate the information[17], it is because of this that a system which can demonstrate small and large scales of data is of high value. Using techniques such as node-link diagrams are helpful because they show the connections present, but can easily become convoluted when many connections are demonstrated[18]. Therefore, for this research, Design Structure Matrices are used.

1.3 DSM & Connective Maps

Design Structure Matrices [1] (DSM) presents a system by which complex systems can be collated and presented. An example of a DSM is shown in figure 2 These systems could be organisation structures, processes, or products. The steps typically taken to produce a DSM is as follows[20]:



	A	B	C	D	E	F	G
A		☒	☒				
B	☒			☒	☒	☒	
C	☒			☒	☒	☒	
D	☒	☒	☒			☒	
E			☒	☒		☒	
F	☒	☒	☒	☒	☒		
G							

Figure 2: Figure displaying a typical DSM graph[20].

1. Breakdown the subjects into smaller systems.
2. Determine the nature of sub-system relationships.These can be created depending upon different categorisation, i.e. function, location, organisation.
3. Construct the DSM using both input and output relationships and demonstrate the relevant influence.

[1]Which can also be called such names as: dependency source matrix, dependency structure matrix/method, incidence matrix, dependency map, or n^2 graph. A digraph if time-based, or an adjacency matrix in Graph Theory if static[19]

A DSM acts as a visual presentation of information or data, this is shown by using a square matrix, as shown in figure 2.The graphic demonstrates how elements are displayed horizontally and vertically[20]. Elements which meets other elements show the pertaining information in that cell of the matrix. In most cases this takes the form of a binary denotation which shows if there is a connection present[21]. An element will encounter itself upon the diagonal can be considered a loop, or are disregarded depending upon the specific system[22].

Systems can be decomposed in this manner gaining a set of subsystems which have each individual elements listed. These relationships can then be represented within the DSM - and then be built back up[21]. This specific type of DSM is known as a product model[20]. As Browning appreciated, DSM's can present the information concurrently, known as a static DSM. However, if a system has chronology, it is time based - elements placed closer to the first cell of the matrix (top-left) occur earlier than those in the last cell[19]. Other variations are also possible for static DSM's, these being considered object or component based, or a processed based or organisational system[23]. Time-based DSM's similarly have two further distinctions: activity/schedule based, or parameter-based systems[19]. Due to the nature of this research, it is only relevant to consider static DSM's.

It is because DSM's are adaptable and can be altered to comply with the requirements of the system efficiently, that they are prevalent in describing the degree of interdependent systems[24]. It is also possible within a DSM, to visualise an entire system, because of this it becomes more feasible to quickly recognise patterns by sight rather than carrying out operations to gauge the presence of relationships or patterns quantitatively[25]. However, this approach requires that a certain balance is reached so that the DSM is still usable, so using more than 50 components is no longer advantageous[4]. Feedback behaviour can also be shown in DSM's, while in other modelling systems, such as PERT modelling or Gantt Charts[26], do not allow for feedback to be recognised in the system. Specifically to this project, the DSM structure ensures that vertical components show an *instigating* subsystem, and horizontal components show the *affected* sub-system. This is important because the information which is deemed most valuable is prioritised during visualisation.

1.4 Change Prediction Method

Another advantage of using DSM's is that they can have different algorithms applied to them. These can include sequencing, tearing, and clustering[27] [28] [29] however these are not within the scope of this research. The algorithm which proves most relevant is the Change Prediction Method[30] (CPM) which was first introduced by Simons[31], and built upon by Clarkson[3], it was then further developed alongside industry by Jarrett[32] [33]. The CPM is an approach to estimate how risk can be propagated by change occurring within the system. The mechanism functions by inputting product data and producing an indicative examination of the relationships between elements. The CPM requires the DSM's which describe the system, whereupon predictions can be created about the level of risk within the system. This system means that dependencies between subsystems and their relevant constructions and possible alterations can be understood[3]. The level of risk is determined by the assigned probabilistic values associated with an element. This creates the predictive matrix denoting the behaviours of each element[3] having iterated over each set by the Cambridge Advanced Modeller (CAM) software. The CPM was created as a tool to support and guide designers at points where decision making is paramount.

1.4.1 Risk

Risk is a measure of the volatility and possible issues which could arise from an artifact[34]. In this research, risk is defined as the product of how probable a change is known as *Likelihood*; and of how much a change could influence the system, known as *Impact*. In this research, to formally understand risk, it is important that the methodology of the evaluation of a network is established. For the DSM example shown in figure 2, the network describing the exact same information exists and is shown in figure 3.

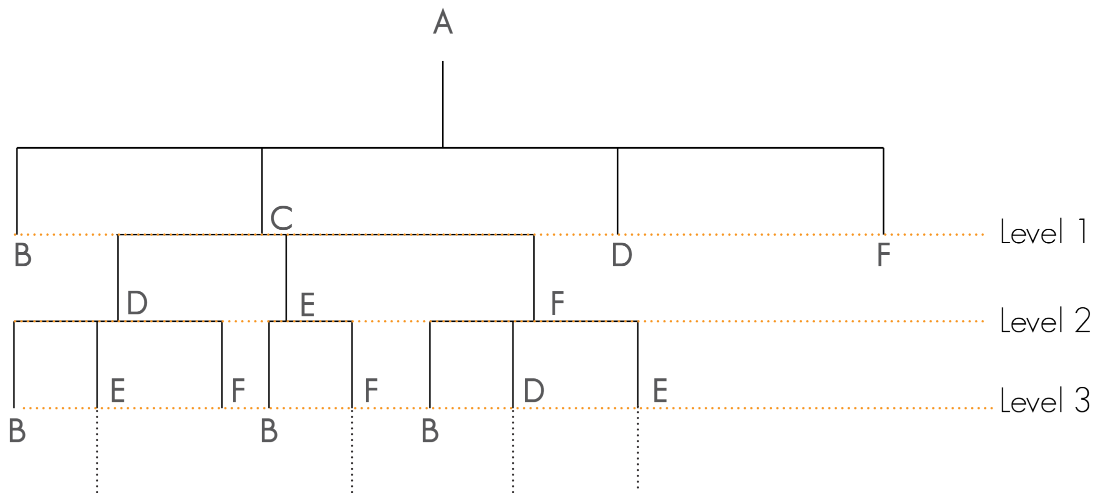


Figure 3: A connective graph illustrating the dependency shown in figure 2

The network shown in 3 should be evaluated in a certain way in order to allow algorithms to be performed upon it. The process can be carried out recursively which better suits computer analysis techniques as large systems can be evaluated quickly[35]. The definition of direct risk is shown in equation 1[36], it shows the risk, R, is equal to the multiple of the likelihood, L, and the impact, I.

$$R_{A,B} = L_{A,B} \cdot I_{A,B} \tag{1}$$

The graph in figure 3 also shows that indirect risks are possible. The direct *impact* of A affecting B is only true in one case, shown on the very left branch at level 1. However if the indirect *Likelihood* is considered, then it can be seen that not only does $A \Rightarrow B$, but following the possible paths on the second branch that $A \Rightarrow C \Rightarrow D \Rightarrow B$, while it is also true that $A \Rightarrow C \Rightarrow E \Rightarrow B$. The next branch to the right also shows that $A \Rightarrow C \Rightarrow F \Rightarrow B$. Each of these indirect relationships is shown in level 3 and the definition is shown in figure 2[36] where ρ is the partial or indirect risk from element u to b, σ the likelihood of change reaching u from a.

$$\rho_{B,U} = \sigma_{U,A} \cdot L_{B,U} \cdot I_{B,U} \tag{2}$$

The type of risk can also vary as the system can be considered in terms of how the incoming risk and outgoing risk are related. Some elements can be defined as risk absorbers, shown in figure 4 as the bottom right quadrant. While others can be considered as propagation multiplies, shown as the top left quadrant. The top right quadrant shows high risk, while the bottom left shows that the element has neutral behaviour.

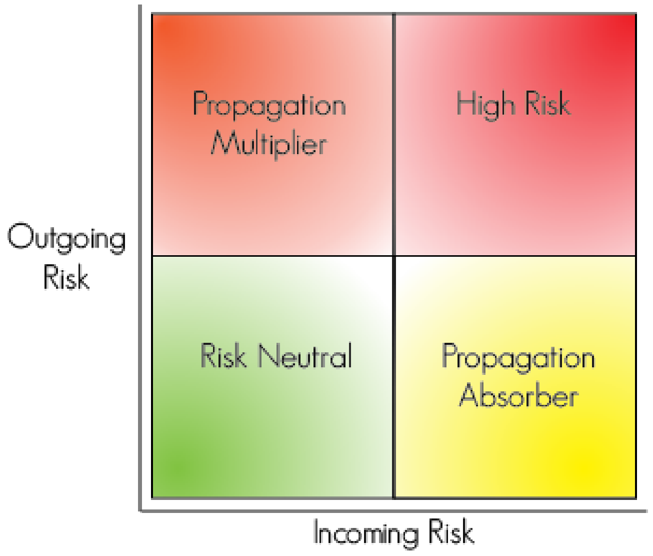


Figure 4: This graph shows a diagram for assessing a system

In order to gain data which will provide *Likelihood* and *impact* values, the various aspects of the system should be assessed. It is because of the difficulty in calibrating a consistent judgement of *Likelihood* and *impact* of an element in a system between disconnected members of an engineers team that determining the values for a unique and complex sub-systems is challenging[4]. In order to cope with this issue, values are assigned to elements by those who have greatest expertise with them.

1.4.2 Design and development: Process Efficiency

There exist two further definitions which build upon those of *impact* and risk. These are of particular interest in this research and are the predominant subject of this research. The first is *Design Process efficiency* which is defined as the reciprocal of the sum of changes and *impact*, shown in figure 3.

$$\frac{1}{Changes + Impact} \tag{3}$$

The design process efficiency shows the volume of possible connections and value in terms of the level of risk of the connections having carried out the CPM process. The second is the Development Process Efficiency which is defined as the reciprocal of the sum of the volume of risk multipliers and the risk. The formula of this definition is shown in figure 4.

$$\frac{1}{Multipliers + Risk} \tag{4}$$

These measures present more specific information about the design set up to the designers that attempting to ascertain the state of a system by other methods and is therefore an incredibly useful tool to base decisions upon.

1.5 Optimisation

Optimisation acts as a mechanism which explores a landscape controlled by different parameters, and seeks to increase the prominence or volume of a desirable feature while minimising the presence of undesirable features[37]. This is carried out by making modifications to the system and adjusting features to suit the constraints, this is useful especially when attempting to understand a mysterious landscape efficiently[38]. Decision making can be a challenging task fraught with risk, so using this mechanism to inform and improve during the process has the potential to improve knowledge of the overall project[39]. To define the optimisation problem generally, begin with the definition of a general function, a mapping between the set of all possible inputs to real numbers in the set $\mathbb{R}$:

$$f : A \to \mathbb{R}. \tag{5}$$

An optimisation algorithm attempts to find the element, x_0 in the set A that maps to the optimum output value. That is the lowest value in a minimisation problem or the greatest in a maximisation problem. For a function of some number of real parameters n, the input set A is equivalent to $\mathbb{R}^n$. This is the case in this application. In the case of a minimisation problem, the optimum solution, x_0, is one of the potentially numerous local minima contained within the function space A[40]. In optimisation algorithms it is typically possible that a system will discover a local minimum and struggle to proceed with the optimisation process. This is because the function must oppose its direction in order to 'backtrack' to a previous point where another differing route may be taken, which could potentially to lead to a possible global minimum - this is also true of finding pathways for global maximums.

These local minima are defined by this condition:

$$\exists \delta . (\forall \boldsymbol{x} . (\|\boldsymbol{x} - \boldsymbol{x}^{*}\| \leq \delta) \Rightarrow f(\boldsymbol{x}^{*}) \leq f(\boldsymbol{x})) \tag{6}$$

This condition explains that there exists a value delta, such that for all x values, the modulus of $x - x*$ is less than or equal to delta. This implies that the function f which describes the system to be optimised is less than or equal to f of $x^{\square}$. However, when multiple functions are also involved within the system, as is true of this experiment. It is because several other factors should be optimised (either to find the minima, or maxima) that this experiment is particularly complex. This is because different functions may interact with one another and thus influence other systems meaning that their relationships are entangled. However, some relationships may not be connected at all, meaning that some variables could behave independently of one another[41]. Within this research, optimisation is used for the purpose of understanding the landscape that is being produced by accordingly altering values for *Likelihood* and *impact*. These values are changed and developed while adhering to the limitations placed in this experimental model, so that the results accurately reflect the real world solutions. There are many different methods which can be used to discover the possible optimisation strategies of a system, such as a hill climbing algorithm, simulated annealing, and Genetic algorithms. Within this research, Genetic Algorithms were used.

1.5.1 Genetic Algorithms

Genetic Algorithms (GA's) are a subsection of evolutionary algorithms, which are a field of Artificial Intelligence. GA's use a defined search heuristic to carry out optimisation[42]. GA's function by mimicking the way which natural selection works[43]. The algorithm then uses a fitness function which can be considered as an objective to be achieved. The members of the population which best satisfy the fitness function are those who best adhere to the aims of the parameters which it describes[44]. There are two different ways of defining a fitness function, either as a static goal which does not change, or as a dynamic goal which develops by co-evolving or using niche differentiation[45].

The process of evolution within the algorithm is carried out by creating a population of individuals with defined characteristics - known as a chromosome which holds this data[46]. These are subject to evolutionary influences such as selection, crossover, and mutation[47]. Selection within GA's is the action of selecting members of the population because of their specific genomes. These selected members behaviour is then perpetuated as they are specifically chosen and bred due to their desirable traits to pass onto the next generation. The procedure for this action is to evaluate the population based upon the fitness function, which is then normalised. To normalise the fitness values involves dividing the fitnesses of the individuals by the sum of the final fitnesses in order to ensure that the accumulative sum is equal to 1. The population is then ordered by the size of the fitness values. A value is chosen at random between 0 and 1, the first individual is selected whose value is greater than the random value selected previously. This is then carried out again until a sufficient number of individuals are selected, this is determined by the design of the system[48].

Crossover, also known as recombination, is the process of selecting a pair of individuals from the initial or previous generation and breeding them. This has the effect of blending the genetic information together and then using it to create an offspring. This research carries this out by switching the bits of data which hold the characteristics. If the parent individuals selected have elite behaviour compared to the rest of the population, and are bred then the offspring produced should also present these successful characteristics[49].

Mutation changes the values of the genes within an individual and thus provides more genetic diversity in a different way to crossover. This change has the potential to alter the individual radically, this is done by randomly flipping bits of data[50].

1.6 Self Organising Maps

Self Organising Maps (SOM) or Kohonen map, is a system which functions using Artificial Neural Network (ANN) model algorithms. SOM's act as a powerful unsupervised learning mechanism to analyse and visually represent data which is high dimension in its nature[51]. The aim of a SOM is to create a network which responds to the patterns of input to the system. A SOM charts non-linear probabilistic dependencies between data collected by measurement. This data is then collated by using vector quantisation methods, and the relationships are displayed by geometric grids[52]. The system works by organising neurons, whose weights are either initialised with consistent values from the selected input subspace which is encompassed by the largest component eigenvectors, or with small randomised values[53]. The process is described below[54]:

1. Obtain the relevant data
2. Generate and construct the network
3. Set values for the neurone weights
4. Train the network
5. Validate the network

An n-dimensional weight vector (where n is the dimension of the input vector) defines the network values, these are mounted upon a low-dimensional network. Each of these neurons are connected within the network by a neighbourhood relation[55]. This system controls the output graph and the relative structure. The overall effect of this system, produces a grid which deforms depending upon the strength of the weights upon the data points within the system. These regular iterations of shifting weights creating an organised map of showing the discretised representation of the input data[55]. The general process of creating a SOM is shown in figure 5. Image one shows the realm which the data can exist, with the shape showing the experimental data - in reality these will be single data points which are spread around this area in varying density. The second image shows the unaffected map. The final image shows the SOM.

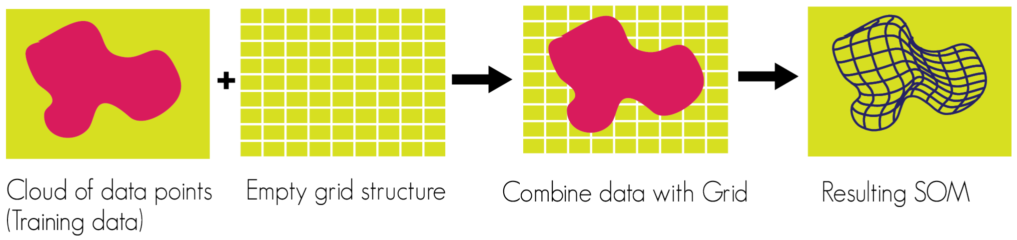


Figure 5: This image depicts the training system of a SOM[52]

However maps such as these can also take another form as shown in figure 6.

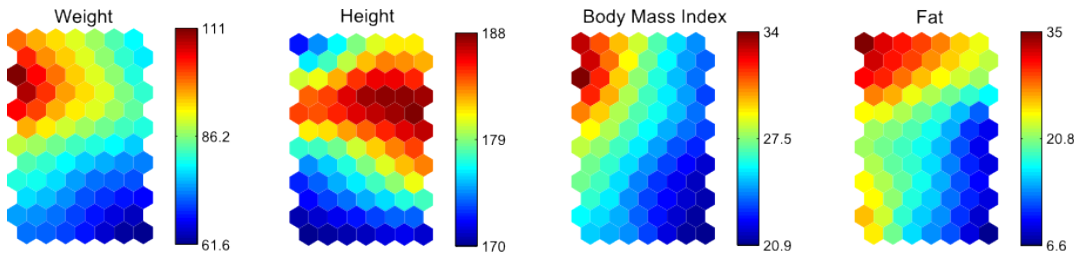


Figure 6: This image shows another way of presenting a self organising map

For more information about the SOM Toolbox used in this research, provided by Mathworks MATLAB (c) software, see the book by Kohonen and the relevant documentation[56] [57] [58].

2 Experimental Methods

2.1 Overall Design

The overall design of this experiment sits within a much larger network of information and skills, because of this, it is easier to understand the processes of the research involved as shown in figure 7.

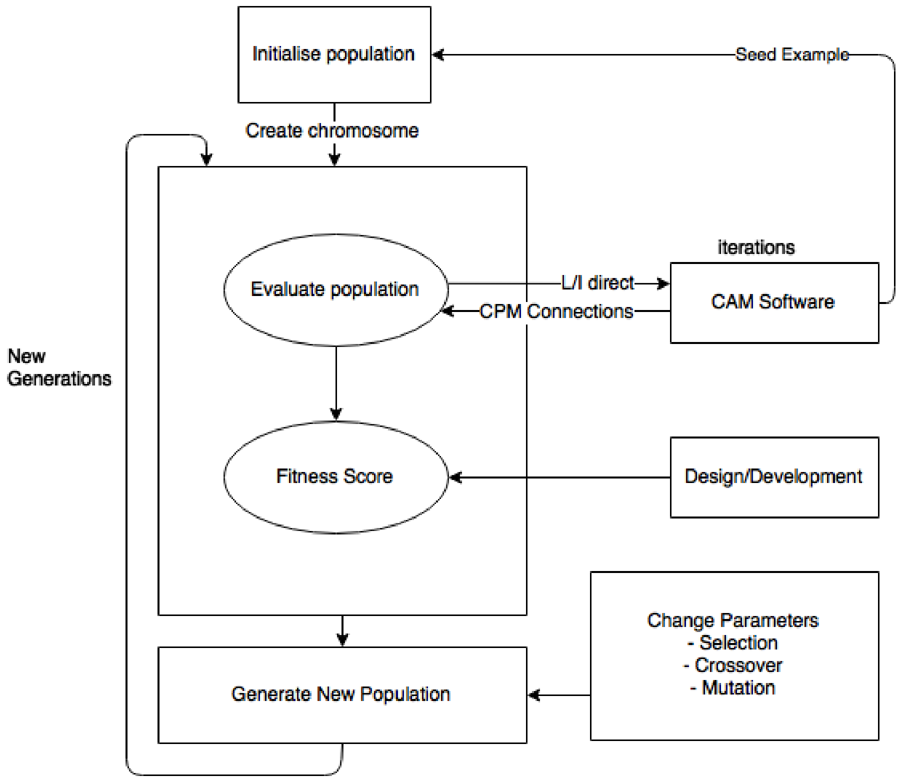


Figure 7: Diagram illustrating the research process

This diagram shows the left section which represents the process of creating a population and developing each generation. The right side shows the way that this system can be influenced to generate variation.

2.1.1 Establishing Initial Information

The standard values used for mutation was 35%, and cross over set at 1 in 12. The experiment was run on an ASUS brand computer, with a processor type Intel Core i5, a processor power of 2.5 GHz, and RAM of 6.00 GB. However due to unrelated software conflicts it was often the case that the hardware did not perform as well as is theoretically possible. Code extracts are present within the appendix in section 6.4.

2.2 Iteration One, the Generalist function

The first task of iteration one was to ensure that the optimisation system satisfied both the development parameter and also the design parameter. Each of the two aspects control 50% of the overall weighting. The experiments were initialised with a generation value of 10 and an initial population value of 10. After each experiment concluded, the population value was incremented by 10 and the next experiment was run; this pattern was followed until the population value reached 200, this was the last experiment for that generation and the run was halted. The generation was then also incremented by 10, and the population was initialised to begin at 10 again, the population was incremented by 10 once again for each experiment until the value reached 200. This pattern continued until runs had been carried out for generations 10, 20, 30, 40, and 50. In total 800 individual experiments were run for iteration one. The standard values used for mutation and cross over were used.

2.3 Iteration Two - Development & Design

Iteration two studied both development and design again, but each were considered independently. The first section ran 800 experiments in the same manner as in iteration one, the only factor which was altered was that the development was weighted at 100%. The second section ran another 800 experiments in exactly the same manner as iteration one also, but instead design was weighted at 100%. The data from iteration one and two were passed into Mathworks MatLab (R2011a) and Microsoft Excel (2013) where graphs were produced. In total 1,600 iterations were carried out for iteration two.

2.4 Iteration Three - Mutation and Crossover

The values for mutation and crossover were varied and run with population and generation values which were best represented by the previous experiments. In the first section of iteration three, the crossover rate was set to 90% and experiments were run for generation 50, population 150, 160, 170, 180, 190, 200. Each of these experiments were carried out three times each for every population as to account for the possible variation. 18 Experiments were run for each section of iteration 3. The second part of iteration three was carried out identically to the first part, except for the crossover rate which was set to 10%.

The third part of iteration three involved setting the mutation back to its initial value of 35%, and changing the mutation rate to 1 in 2. This was carried out for the same values as described in part one and two of iteration three. The fourth was carried out in exactly the same manner as the third, except the crossover rate was decreased to 1 in 24.

2.5 Iteration Four - Rescaled Generalist

Iteration four involved re-running the Generalist function which had been edited to rescale the development and design values and was weighted at 100% for the same values used in iteration three (population 150, 160, 170, 180, 190, 200 - each three times) with a mutation and crossover values reset to the original values.

2.6 SOM

The MatLab (R) package was used to create a set of Self Organising Maps (SOM), where the data from each iteration was fed into the programme and produced a graph.

3 Results

The first graph of each section shows the 2D graphical representation of the data. They show the plotted values for fitness - which can be considered the relative 'success' of the population - upon the y-axis. The generation values from 0 to 50 are plotted along the x-axis [2]. The size of the population is shows in darker or lighter colours along the graph and act as a key because it can be challenging to identify each without the colour distinction. The darker colours correspond to a larger population, the lighter colours correspond to a smaller population. The second graphs show a tilted view of the data so that all three axis can be observed at once. It is helpful to consider these diagrams as looking at the surfaces they produce. In the 3D graphs the colours are used to indicate the fitness value. Dark green indicates a low fitness, light green shows a middle range fitness, and yellow shows a high fitness. A key to understand the gradation and relations of these colours is shown on the right of each diagram in order to give easy comparison as to the value of the fitness. Fitness is shown on the y-axis, population shown on the x-axis, and generation is shown on the z-axis. Generation can be considered as how long the experiment could run through and is connected to how often the best fitness is selected. The third set of graphs within the results show the data in the form of a SOM. These graph colour areas darker or lighter depending upon the similarity between data sets. If there is a drastic difference in size between values then the graph is coloured darker, if the relationship is similar then the graph shows a section as lighter. This is used to cluster together the data and identify certain groups within it. The axis of these graphs show the position of the neurones in the network. Within the appendix there are more views of the 3D graph so that more detail can be obtained and any confusion clarified. The MATLAB code is also present for both the 2D and 3D graphs within the appendix.

3.1 Iteration One: Generalist

This experiment was the upon the Generalist set-up which weighted development and design equally.

[2]Other experiments were also carried out for generation 10, 20, 30, and 40 with the same parameters for iteration 1 and 2. These are not shown because this data set of 50 best represents entirety of data which was collected. This is because the sample is larger and therefore more representative over a larger range.

3.1.1 2D Graphical Depiction

As can be seen in 8, a two-dimensional representation of the results for iteration one are shown.

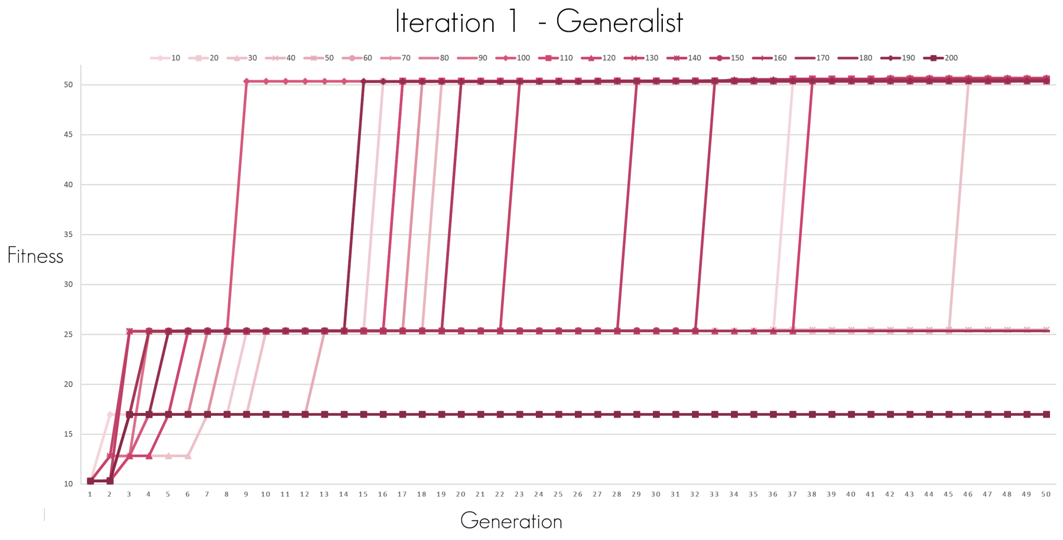


Figure 8: This figure shows the results for the generalist experiment during iteration one.

Figure 8 shows that initially the fitness values of all populations rapidly increases within the first 2 generations. Between generation 2 and 3 a large increase in fitness is apparent within darker coloured generations. It can also be seen that for some populations, more generations may be necessary to reach the same large fitness value as well-performing populations. It is shown in the graph that these jumps can be made in steps while others reach a temporary equilibrium which may last many generations before reaching a greater fitness. For population 200 the temporary equilibrium at generation 3 with fitness value between 15 and 20 for the entirety of the experiment. It can be clearly seen however, that of the populations which made the jump from this fitness between 15 and 20, the darker values jump during earlier generations as opposed to the lighter values jumping to the higher fitness value during later generations.

3.1.2 3D Graphical Depiction

The graph shown in 9 shows the same results but in terms of a three dimensional landscape.

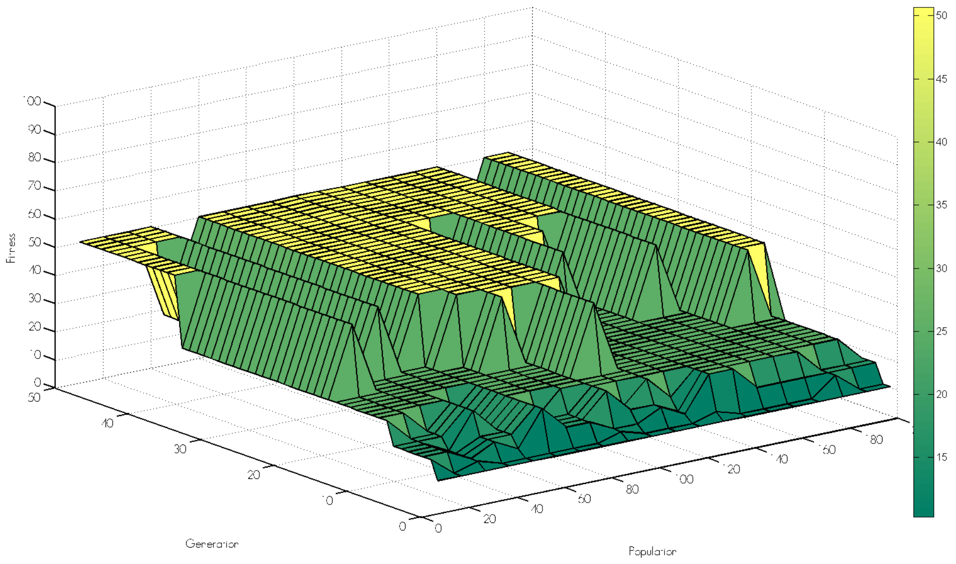


Figure 9: 3D representation of those same results

Figure 9 shows the same results as in the previous section, but for clarity these are displayed in a different format. Here, the journey between the first generation and the last generation can be seen as a surface. This surface gives an understanding about how each population develops and varies compared to those of other populations. It can be seen in the darker green areas that different populations take a different amount of time to increase fitness. This can be seen in the way that the surface initially has many peaks and troughs. The differences here are small as it takes few generations to increase fitness several times, causing a smoother surface of dips and peaks. Each population can be seen in the slightly less dark green areas and remain at this level for some generations, this can be seen because a flat surface exists which shows that minimal change exists. Past this point (a higher generation) shows sharp and infrequent peaks which are near vertical, showing that the fitnesses increase. Different populations reach this point at different times, but most peak where upon the system does not increase past the flat lighter surface shown here as yellow.

3.1.3 Neural Network Results

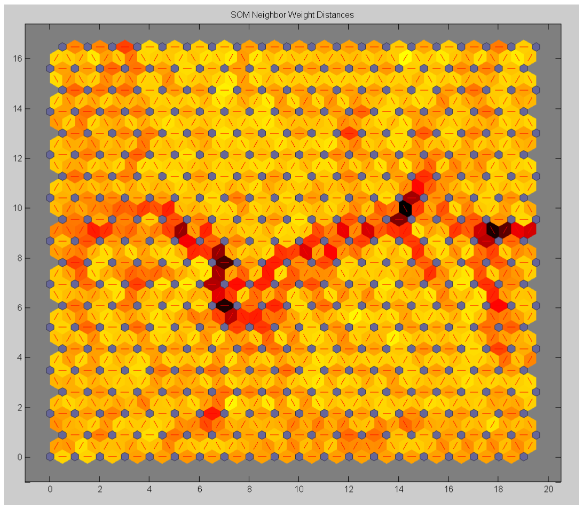


Figure 10: The same results shows as a clustered SOM.

The SOM map produced during this iteration shows a strong striation of darker colouring, indicating a similar type of clustering behaviour of the SOM.

3.2 Iteration Two: Development & Design

This experiment was designed to test both Development and Design independently by weighting each to 100% and running tests separately.

3.2.1 Development

This graphs in this section show the results when the experiments are run, having weighted only development.

3.2.1.1 2D Graphical depiction

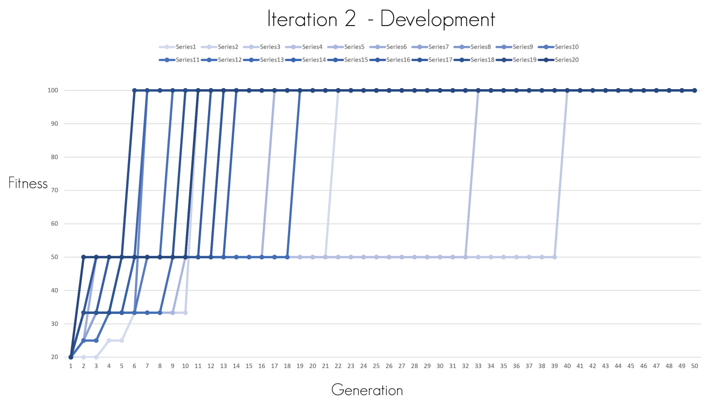


Figure 11: DEV

It can be seen that most populations increase rapidly within the first few generations in figure 11. By generation 3, darker populations increase at a greater rate in fewer generations than lighter populations. Some of these populations increase in a single bound while others produce distinct steps which indicates different fitness values in order to reach the same point. Many populations seemed to become 'stuck' by maintaining constant fitness values during 10 generations at a fitness between 30 and 35. While a darker set of populations appear to also become 'stuck' however this occurs at a a higher fitness level of 50 from generations 2 until 18, whereupon a single population maintains this fitness value for the duration of the experiment up until generation 39. Many populations eventually make the jump in fitness value. The majority of these occur between generation 4 and 39.

3.2.1.2 3D Graphical depiction

The 3D graph shows this behaviour but spread out across many populations. It can be seen in figure 12

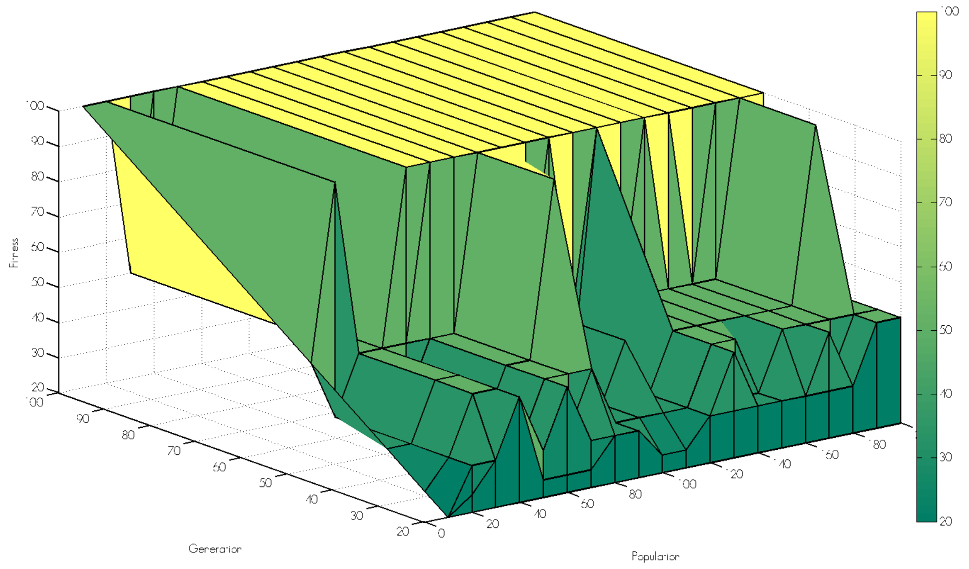


Figure 12: 3D graph showing the development results.

figure 12 shows an increase initially displaying a large gradient as the darker green area appears to show an almost vertical surface. This behaviour is universal across each of the populations during the early generations. This area has many dips and peaks, showing more variation in the behaviour of the populations. Most populations reach a greater fitness while few remain at a lower fitness value during generations between 30 and 35. This can be seen as a flat surface with a light green shade which exist between the initial peaks and the below the flat light coloured sections at the top of the graph. Few populations rapidly increase during early generations, and are illustrated as triangular shaped objects within the graph which protrude from the back of the graph. When a surface is created at the peak value then it is light coloured, shown here in yellow, and represents achievement of a high fitness value which has been established for many generations.

3.2.1.3 Neural Network

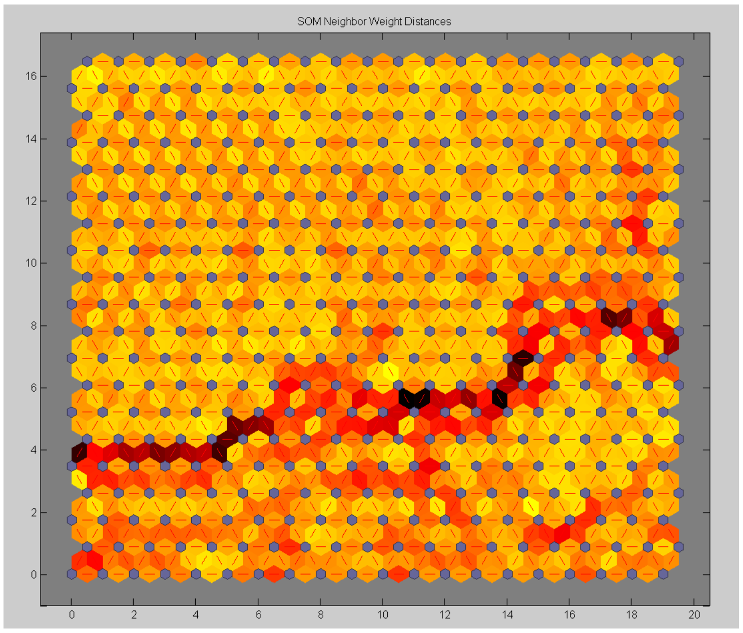


Figure 13: A graph showing the clustering behaviour of development

This graph in figure 13 shows the clustering of the system is dense due to the darkness of the colours presented in the overall space. The presence of a dominant and darker coloured striation through the landscape indicates that there exists a vast different between weights and therefore a relationship present between the variables.

3.2.2 Design

This graph in figure 11 show the results when the experiments are run, having weighted only design.

3.2.2.1 2D Graphical depiction

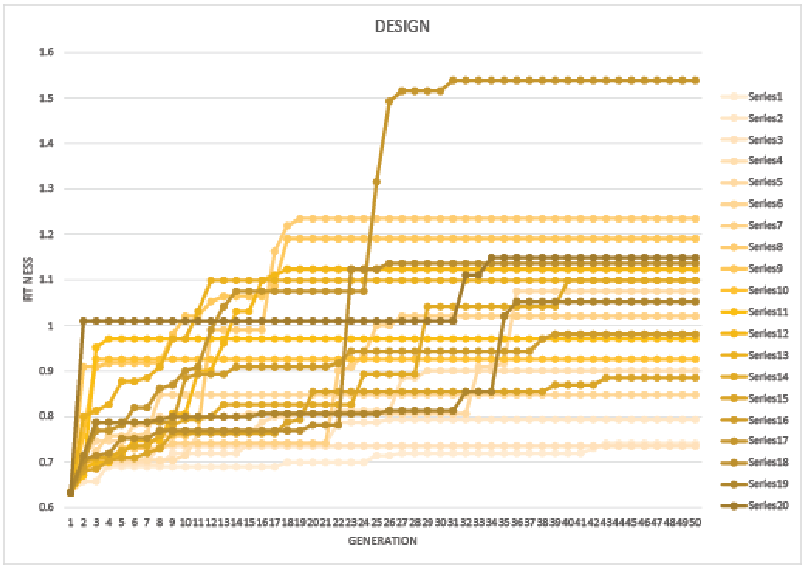


Figure 14: This graph illustrates the same results presented previous, but with a clustering algorithm applied by a neural network creating a SOM.

Figure 14 shows that notably during the design experiments the maximum fitness values range from 0.6 until 1.6. The overall behaviour however is similar to the results of iteration one despite the difference in scale. It should be noted that the populations continue to climb even upon reaching high generations. This continuous climbing behaviour differs from previous experiments as these present results that show a decrease in the frequency of increasing fitness values at larger generations. Another unique characteristic of this graph is that many different paths are taken and few populations overlap which is dissimilar to past experiments.

3.2.2.2 3D Graphical depiction

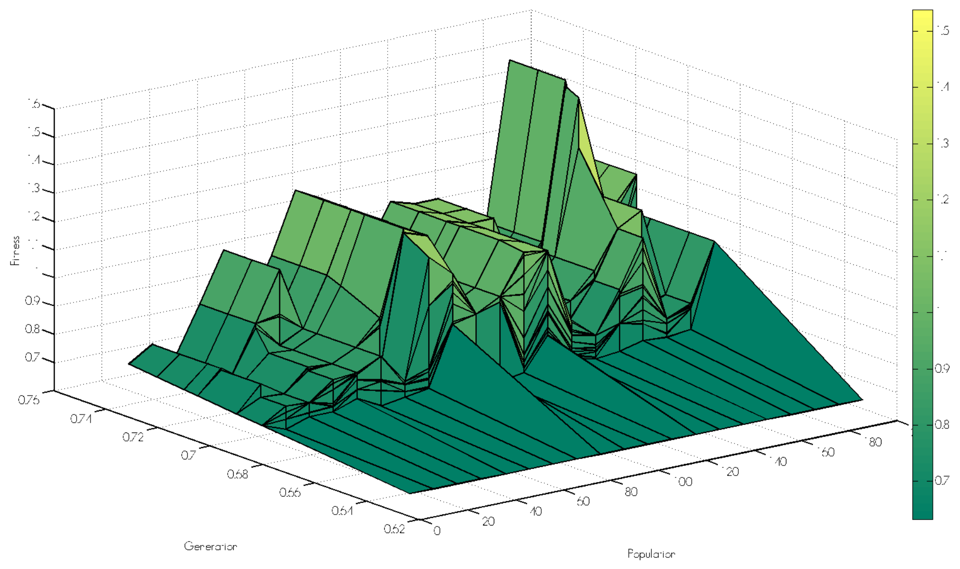


Figure 15: 3D graph of the results for design.

This graph shows a that populations require many generations to increase fitness values, and this action is done so in a single step, which is shows by the smooth surface within figure 15. After this point, the graph increases rapidly and forms a detailed surface showing small increments in fitness for all populations. This type of layering produces peaks which vary is height across the landscape, indicating a point of high activity for the optimiser. This type of behaviour is maintained and fitness increases less frequently towards later generations.

3.2.2.3 Neural Network

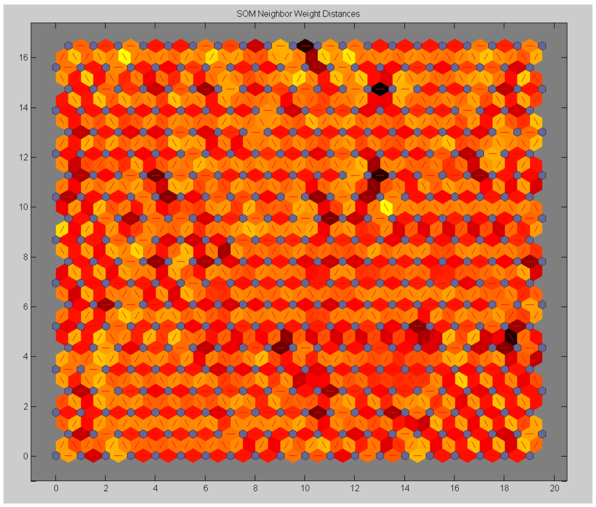


Figure 16: SOM clustering for design.

Figure 16 shows the SOM map produced by this data. The results here show a dark, landscape which maintains a consistent tone across the range of values. This means that clustering has occurred across the entire system and does not give definitive information as to the possible relationships present.

3.3 Iteration Three: Mutation and Crossover

Iteration three and four involved experiments using population values of 150, 160, 170, 180, 190, and 200 three times each. These are plotted in groups show by colour. Lighter groups show smaller populations. Darker coloured groups show larger populations.

3.3.1 Mutation

The mutation rate was changed from 1 in 12 to 1 in 2, and then to 1 in 24. The experiment was performed upon design. Each of the series within the key at the base of each diagram shows the population. These are grouped together.

3.3.1.1 2D Graphical depiction

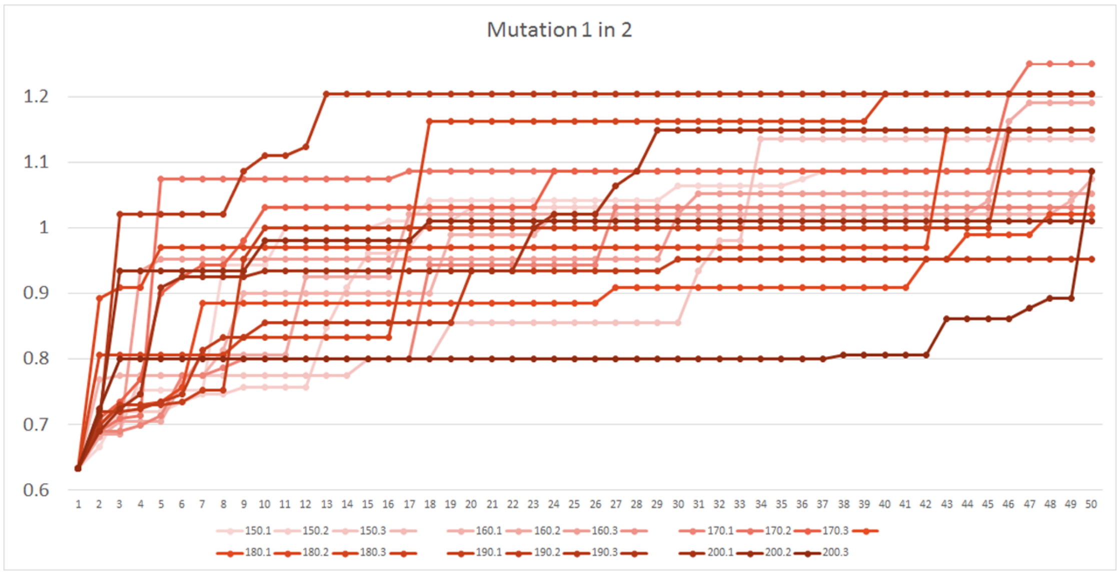


Figure 17: Mutation rate of 1 in 2, for iterations 3.

Figure 17 shows the results for a mutation rate of 1 in 2. The graph shows that during generations 1 and 10, all population increase fitness value. However some populations require a greater number of generations and steps to do this. It can be noted that the temporary plateau behaviour is also present within this graph as many populations do not always continually increase. This graph also demonstrates the behaviour present in the initial design experiment where populations continue to climb during high generations.

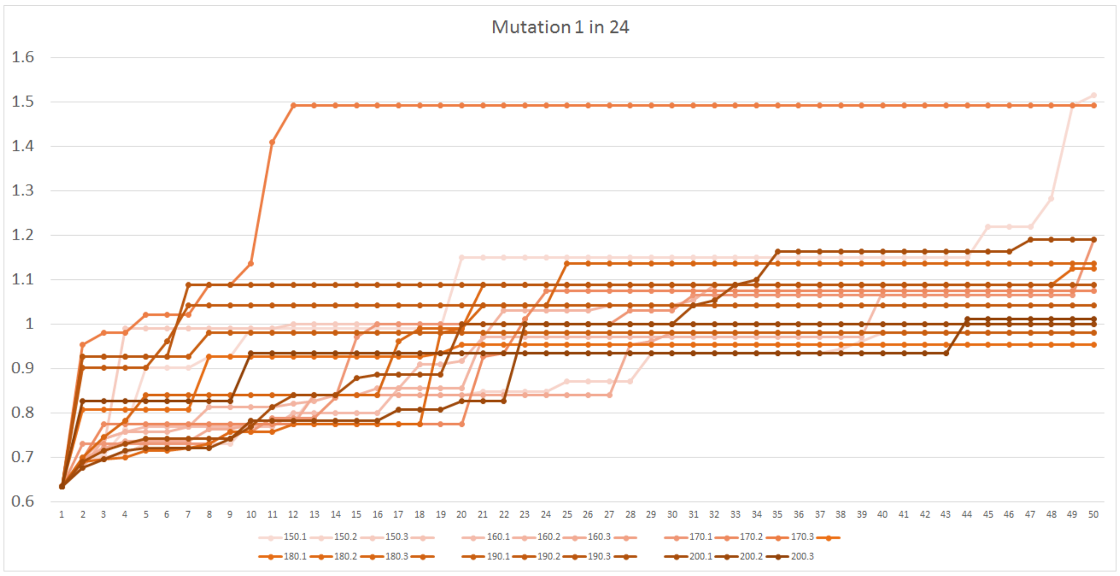


Figure 18: Mutation rate of 1 in 24, for iterations 3.

The second graph which shows the results for a less frequent mutation rate is shown in figure 18. The graph shows that each population increases during the first generations, however, the rate at which the fitness values increases is highly dispersed as some populations have high fitness values at low generations, while other do not display this as strongly. The populations then continue to develop with peaks in fitness occurring at generations 8 to 24. A temporary plateau is also evident for the majority of populations at generation 24.

3.3.2 Crossover

3.3.2.1 2D Graphical depiction

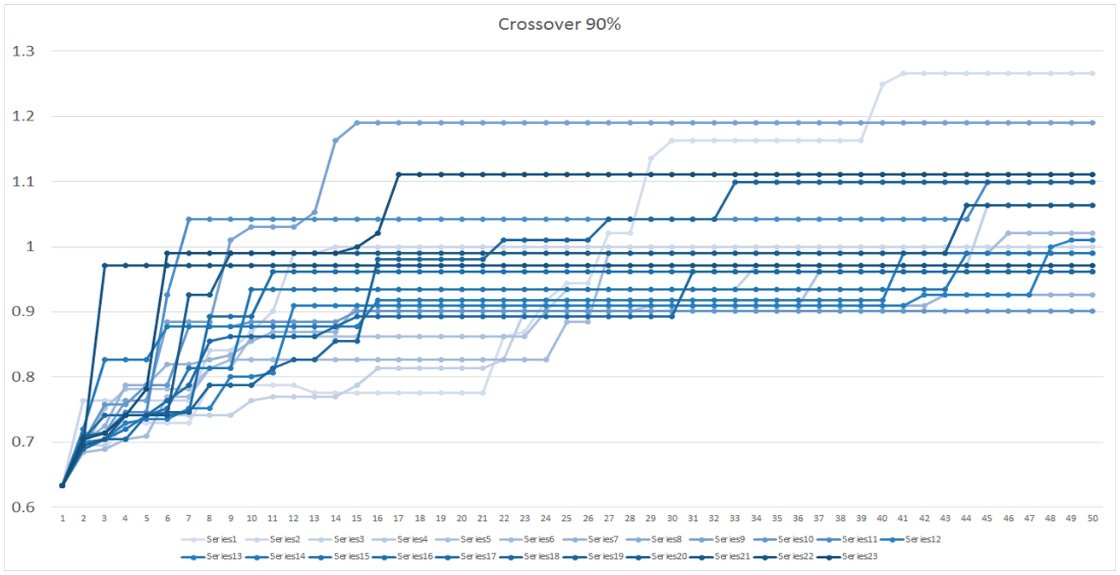


Figure 19: The results shown in a 2D graph for a crossover rate of 90%

Figure 19 shows the behaviour of the system having given a high rate of crossover. The graph shows that initially most populations increase at the same rate in the same manner. The populations then divide up and spread further and the behaviour begins to change between generations 5 to 20. The fitnesses then become more stable, where upon increases in fitness become less frequent, but still present. The smallest population, 150, initially shows a weak performance as the values are some of the lowest fitness values initially. But by generation 42, this population has the largest fitness value, demonstrating variation.

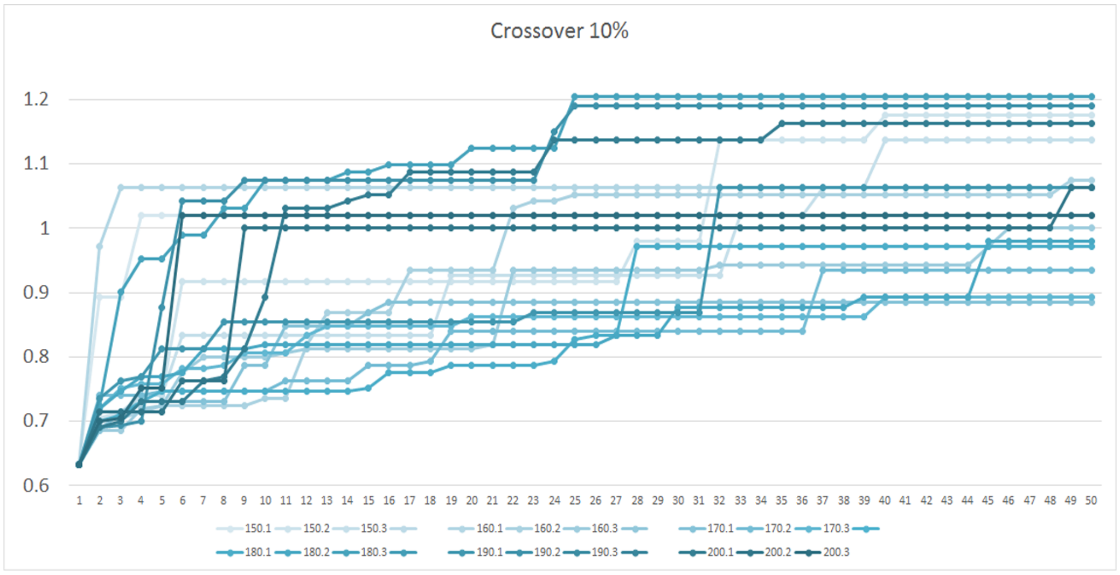


Figure 20: The results shown in a 2D graph for a crossover rate of 10%

The graph shown in figure 20 shows the behaviour of the graph for a crossover rate of 10%. Initially the fitnesses of the population are more dispersed than those of the high crossover rate shown in the previous graph. However this is true for a few populations which vary in size, as some populations are small and others are large. The fitnesses of the populations appear to be changing frequently which shows the development of the system. However the maximum fitness shown compares similarly with that of the previous experiment which tested a crossover rate of 10%.

3.4 Iteration Four: Rescaled Generalist

3.4.1 2D Graphical Depiction

This image shows the plotted mean values for each of the populations tested.

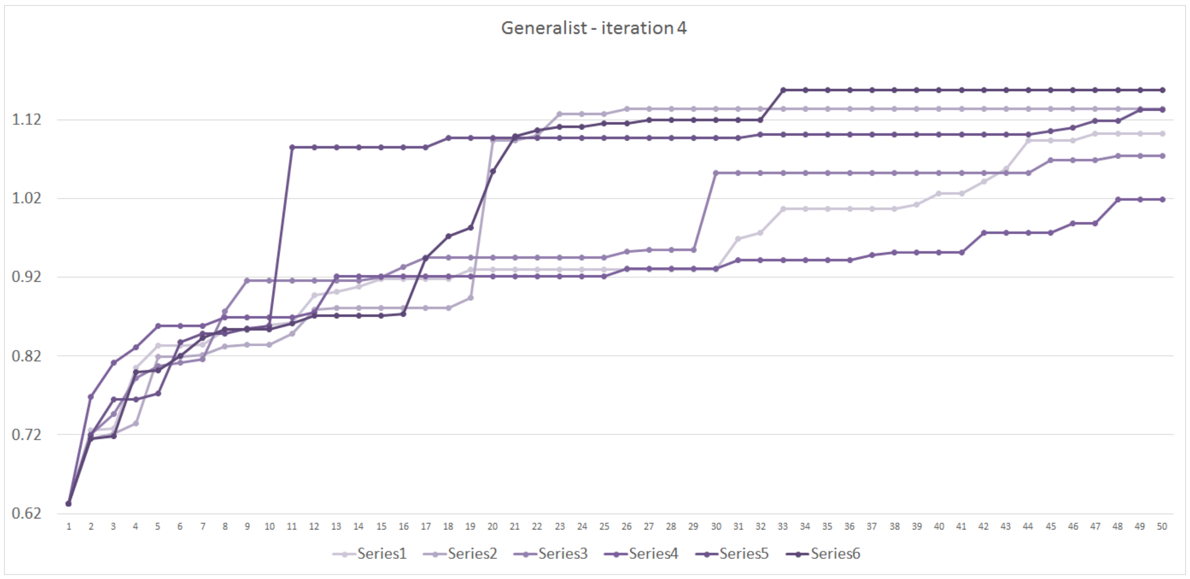


Figure 21: Results of the second generalist

This graph in figure 21 shows the behaviour of the rescaled generalist. It should be noted that the maximum fitness value is 1.2 compared to the original value of 50. The behaviour is also independent between generations as there is little overlap of values. However this could be a product of the data processing technique. It is also notable that the overall behaviour is similar in structure to the original generalist.

3.4.2 3D Graphical Depiction

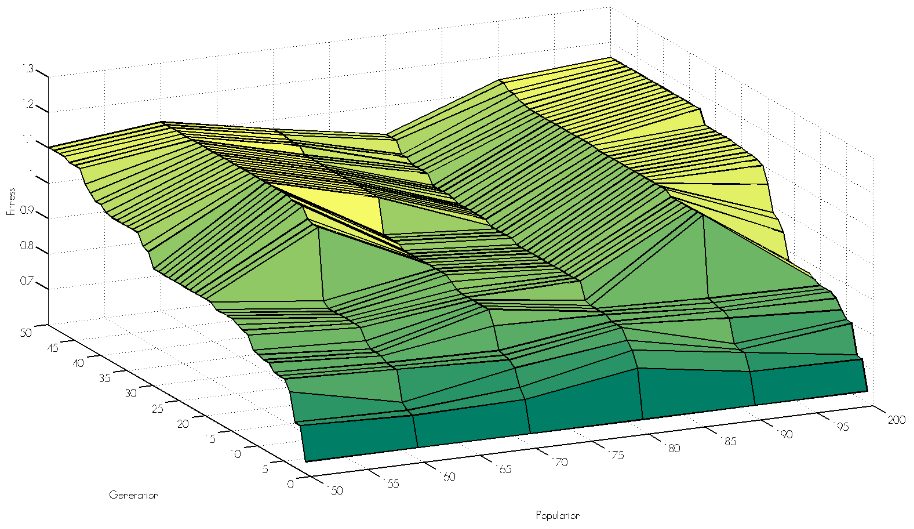


Figure 22: Results of the second generalist

The 3D representation of results show a landscape with little variation between values compared to the original generalist 3D graph shown in 9, which is also possible because of the averaging technique. However it should be noted that this experiment runs for only a few populations rather than a wide range during iteration one which may also contribute to the shape of the landscape.

4 Discussion

These experiments were challenging to present, as there are many different variables involved. It would be optimal to present a graph displaying the generations, populations, design, development, design & development, mutation, and crossover for each and every one of the possible chromosomes. These would then have to be compared in order to understand the affect each factor has upon the system. This approach would infeasible, and therefore this research has dealt with this by breaking down each of these variables separately and experimented upon them individually in order to understand the influence of each factor.

The graphs used were selected because they showed how the fitness functions changed as generations developed; and give an indication as to the paths that the individuals took and therefore the solutions related with these. The styles of graphs were selected because they allowed quick identification of behaviour and therefore aided decision making. The

2D graphs are purposely simple so that they can be understood easily, but are flawed because the data from some values will overlap and hide the other data present. This prompted the use of 3D graphs in order to show the entire landscape of the graph. It is also important to identify the presence of relationships between variables, and so the self organising map was used to create a visual method of clustering data. Each of these graphs present the values with the highest fitnesses and these high fitness values are connected to an individual which presents certain *likelihood* and *impact* values within a DSM. The construction of the system is not altered as these are fixed, but the connections are reorganised and risk values are evaluated. These successful DSM's with high fitness values could present a more modular approach, a system with a reduced number of indirect connections, or another set of advantages.

Overall, the graphs show consistent climbing behaviour, which reflects the style of the optimisation method. It should be noted that because the experiments are structured this way, there exist limitations upon the ability of the individuals to select a path which has a lower value. This is due to the fact that at the end of each generation, only the best fitness values are selected. This behaviour limits the algorithm because local maximums can be found, but global maximums could potentially be hidden. However this does have the effect that the algorithm is always improving the system with every generation, and therefore possibly performing very well for local maximums.

4.1 Iteration One: Generalist

Iteration one was created to study both development and design simultaneously in order to show the feasibility that these two desirable functions could both be optimised. It was not known if these could be optimised together, which led to different possibilities about the outcome shown in figure 23.

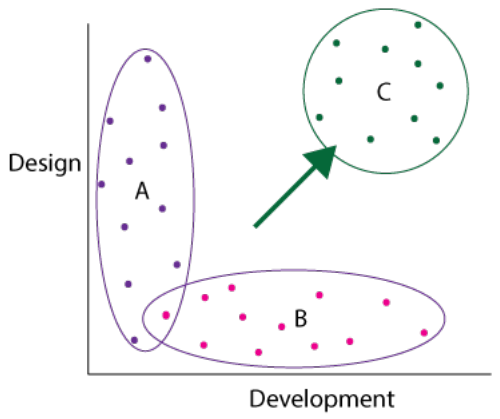


Figure 23: A graph showing the possible behaviour of the generalist function. The arrow shows the direction and distinct area which would be most desirable.

It could have been that the results show the data occurring with a large value increase in development and a small increase in design - shown as area B, and another group or sets of groups with a large increase in design, and a small increase in development - shown as area A. This would be undesirable as the 'best' outcome would be to create a population which presents large values for both design and development - shown as area C. It is typical to expect this kind of convergence because different groups of individuals tend towards different factors. However because the relationship between design and development is unknown, it means that the behaviour could be affected. Having completed the optimisation process for the generalist and creating the graphs shown within iteration one of the results section, the overall landscape was produced. But the initial question about the behaviour of the system cannot yet be answered, because the the individual effect of each is not obvious - thus creating the need for the study of iteration two, where both factors have been considered independently.

The secondary goal of this iteration of experiments was to discover if there existed a positive correlation between the size of the population, or the number of generations, and the fitness value. It was important that this was established because running experiments with large values for population and generation is computationally expensive. The most desirable result would be to find values which require the least amount of resources (time, computation) and provide the largest set of results. The theory and results shown in figure 8 suggest strongly that the more generations, the more opportunity exists for a population to reach a higher fitness value due to the trend of increasing values. The presence of a light coloured areas and a tall peak becoming more prominent at later generations supports this in figure 9.

From the results it is challenging to determine if a correlation does exist between population and high fitness because it is not obvious visually in 9. This means that mathematical methods to define the relationships prove more effective. One method to do this could be to take 'slices' across the 3D graph in figure 9 so that a 2D section is displayed which plots the population and the generation and presents values for fitness. Taking these 'slices' across different generations and plotting the linear regression will produce a value for gradient. If this value is positive then a relationship between increasing population and fitness does exist. If neutral, there exists no relationship; and if negative, then this suggests that a smaller population is linked with a higher fitness, which is the opposite to what would be expected. However the most likely outcome is that for each section taken from the data, there is great variation within the gradient values because of the optimisation method. Another way to study the relationship would be to identify the generation at which each population reached peak fitness in order to understand if a relationship does exist. However due to time constraints this was not possible, and it was deemed more valuable to study further experiments instead.

The SOM graph shown in figure 10, indicates strongly that a relationship exists between certain variables due to the strong striation of clustering across the graph. This can be treated as more evidence of increasing generation creating larger fitness values.

4.2 Iteration Two: Development & Design

Having carried out the first iteration, the next logical step was to consider both of the factors independently. Initially it was hoped that the results would present evidence of large fitness values for both development and design, and that these values were being reached during the first generations. However it is also possible that the results would demonstrate that within the previous experiment, the design factor was limiting the fitness of the development. However this could also be true in the reverse as development could be limiting design. This means that it was important to consider both independently and compare behaviours in order to deduce what influence each factor had upon the generalist system.

4.2.1 Development

As shown in section 1.4.2 the definition of development shows that maximising development is related to minimising the risk of the system and the amplifiers of that risk. This means that the populations which reaches the largest fitness values are the most desirable and provide some of the best possible solutions for the system. The results from 11, shows that the maximum fitness value was 100, which was far higher than expected based upon the maximum fitness of 50 during iteration 1. This result strongly indicates that design was limiting the achievement of development, or in the case that the design value produces a high fitness value - that by virtue of the fact of having two competing factors together, the overall level of achievement of the fitness values is limited. In order to test this, it is beneficial to run the experiments for design independently which was carried out in the next step of this iteration.

It is also evident that the fitness values climb fast and during the first 16 generations in figure 11, which was notable compared to the relatively slow generalist in iteration 1 shown in figure 8, which achieved high values of fitness for most populations at generation 46. This also potentially provides some reasoning about if the development is being limited by the design. The fact that such high fitnesses can be reached is a good indication, especially when considering the prediction that the optimisation method would cause populations to become stuck in local minima. It is also interesting that all of the populations reached the maximum fitness, as shown in figure 12, suggesting that the landscape could be more simple than expected. It indicates that maxima are common, or that there is one maxima and the rest of the landscape has few other obstacles. This can be seen in figure 12 which shows sections with flat planes, indicating the possibility that populations briefly encountered local minima or maxima.

The graphs in figures 11 and 12 also add confidence to the conclusion of iteration one, stating that increasing generation is related to increasing fitness values. It appears upon figure 11 that the darker coloured generations increase during earlier generations than smaller populations. This suggests that using larger populations is more beneficial to the system, as theoretically, there exist a greater number of members to explore the space.

The SOM graph shown in figure 13 shows a relationship is present due to the dark striation across the graph, however this pattern is more broken than the results from the generalist shown in figure 10. This indicates that the same relationship is present, and that it is weaker during iteration two.

4.2.2 Design

The object of studying design is to understand the impact upon the system, and to measure the severity of any consequences caused by making changes to the system. In order to ensure that any changes with the system produce a minimal affect upon the overall system, it is important to maximise design. It was expected that the fitness values produced from this experiment would be large due to the results produced in the previous iteration displaying a maximum fitness of 50. However it was more likely that the fitness values would be small, and that these were limiting the development feature during the generalist experiments during iteration one. The results for design in figure 14 show that the maximum fitnesses achieved were 1.6. Notably this is a small value compared to the maximum fitness of 100 produced in the development experiments, shown in figure 11. The small value does endorse the suggested theory that the generalists fitness was lower than the fitness for development, because of the influence that this small design value had upon the entire system.

It can be seen that during the design experiments, more generations were required to reach a high fitness values than during the development experiment. This could be because it takes the design populations longer to begin reaching peaks, this can be seen in figure 15 by the small gradient of the surface at early generations. However the scaling difference between fitness values should be taken into account. The behaviour also shows that the populations continue to climb and make progress compared to the results of iteration 1 and 2, where climbing behaviour ceased for most populations between generations 30 to 50 - shown in figures 8 and 11. The effect is that any progress in the future is limited by the already slow start, possibly indicating that the surface is detailed, presenting many local minima or maxima.

It can also be concluded that design shows a relationship between increasing population, and reaching a maximum fitness in fewer generations. However from the results, it is visually unclear whether there is a strong direct relationship that exists. For this reason it would be beneficial to take the measures proposed during the discussion of iteration one.

The SOM graph shown in figure 16 shows clustering behaviour, but this is not distinct as no defined areas demonstrate a notable colour difference. This indicates that any relationships here are particularly weak.

4.3 Iteration Three: Mutation and Crossover

Mutation and crossover create unique effects upon the system as they maintain and encourage genetic diversity. Mutation helps avoid convergence early on and therefore opens up the number of possible options to take in future. Overall the effect is that discovering and overcoming local minimums or maximums within the landscape becomes more common. Implementing mutation is important because it allows diversity to become re-established at different stages of the experiments. Crossover creates a different effect to the divergence of mutation as it encourages convergence. This causes each member of the population to become closer to specific minimums and maximums. Crossover is advantageous because it could aid the individuals in finding the global maximum within the search area. This could also be problematic because if the global maximum is 'hidden' behind local maximums, then the global maximum will become unobtainable.

Is was expected that the results of the system would vary greatly under the influence of mutation and crossover. However it was not clear how this would occur. This is because the frequency and probability in which these altered variables were applied did not have a clear previously established relationship. Thus meaning that the expected results would depend strongly upon the values chosen for crossover and mutation. For this reason it was important that variation in the values of for both crossover and mutation were considered.

It was because of the possible variation of fitness values established during previous experiments that iteration 4 no longer studied the same set of populations. During iteration 3, and 4, each population was run 3 times each in order to account for the variation, and gain an understanding as to its effect upon the results. It would also support or quash the idea that certain populations may have 'good luck' and 'bad luck' which could explain the relationship between population size and fitness. This smaller set of experiments was chosen because the most data could be gained from the least computational cost.

4.3.1 Mutation

The first values used to explore the effect of changing only mutation during iteration 3's design experiment were 1 in 2. This means that for every generation, the rate at which the population mutates is half of the time. The results of this are shown in figure 17. This value was purposely set at a high value so that the effect upon the system would be drastic, and initial conclusions could be made. It was however possible that this was such a large mutation rate that any relationship would be hidden. The influence of a regular mutation rate could potentially inhibit the progress of individuals because as each reaches a 'good' point and becomes closer to a maxima, then the individual could suffer a change. Thus causing issues for individuals who were already following an effective path, and aiding those individuals who weren't with a change to their chromosome. This latter theory of inhibiting behaviour seems to be supported by the fact that the highest fitness values reached were 1.2 - shown in figure 17, instead of the 1.6 - shown in figure 14 of the standard design result. It is because this value is comparably much lower, that this experiment was not as successful as the standard mutation rate of 1 in 12.

The next mutation experiment used a value of 1 in 24 which is a much less frequent rate compared to the standard 1 in 12, the results for this are shown in figure 18. The expected effect of such an uncommon rate of change to the system, could be that the individuals becomes stuck at local maximums and remain so for the duration of the experiment, or conversely that this behaviour aids the individuals and allows them to continue the task of finding maximums. From the results it can be seen that the overall maximum fitness produced by this second experiment is 1.5. The continued climbing behaviour seen in the previous experiment of iteration 3 is present within this experiment also. Overall the fitness is also much lower than in the original design experiment, meaning that values close to 1 in 12 should be considered.

4.3.2 Crossover

The rates for crossover were also chosen to be extreme, as to ensure that the influence upon the system could be noted more clearly. It was expected that the effect of a high crossover rate would limit the system, and disallow a large amount of continuous climbing, possibly resulting in a low maximum fitness value. The first rate which was used was 90% which was higher than the standard 35%. The results for this are shown in figure 19. this high level of convergence was expected to speed up the system when finding local maximums and then remain stable. However the results show that this is not the case. The maximum fitness values is shown to be between 1.2 and 1.3 which again is a lower fitness than using a standard crossover value of 35% shown in figure 14.

The second crossover rate was 10%, this decrease means that it may be possible for convergence to exist but in a less frequent way which could allow the system to explore more while still becoming focused. It can be seen that the largest fitness value was 1.2 which is lower than the standard 1.6 of the normal design experiment. The data shows that the populations still climb towards the last generation, indicating that decreasing the crossover rate ultimately slows the system down further. In future it is a case of fine tuning the rate of crossover until the highest value can be found. This will involve trying many different iterations of values for crossover in order to select the value which will perform best.

4.4 Iteration Four: Rescaled Generalist

The results from iteration four, shown in figure 21 show similar behaviour compared to the original generalist shown in figure 8. The notable difference after scaling this is that the maximum fitness is around 1.2 which is lower than the original design value, shown in figure 14. This indicates the one of the initial theories that although design and development work efficiently independently, together they may limit each other.

The smoothness of the 3D graph shown in figure 22 indicates that averaging the variation gives a more consistent approximation of the behaviour of the system.

5 Conclusion

These experiments were created to optimise the system presented by industry. This was done in order to create a system which will support and guide designers during the complex design process, while limiting the number of errors and changes which would be necessary. This was done by using genetic algorithms to seek out the best values and organisations of the system and to understand how these can be created. Overall the initial research questions have developed and can be considered partially satisfied. The first questioned making the design process automated, but in reality the optimal values will change for each value. This means that the research question should be developed in a way which aims to investigate a method of finding the best values possible for each project. This could be carried out by using smart artificial intelligence approaches which respond to the optimisation landscape in order to create the best solutions. The second research question was concerned with ensuring that this can happen using software, and upon reflection this research has noted that the software is the correct route to providing these solutions. In terms of testing the effectiveness of the software, the data should be passed back to the overall research team in order for it to be processes further, and assessed alongside the industry partners who have to power to asses. In future this field of research should continue to develop and grow, especially because the potential benefits are so useful.

6 Critical reflection

6.1 Hardware

The specific factor which would have improved the research would have been to have more computational power. The hardware used limited the speed at which these experiments could be executed. It also proved unreliable as the software was prone to crashing, and other processes struggled to run alongside the experiments. Had the system been improved then more experiments could be executed quickly, the data could then have been collated more quickly ensuring that judgements could be made more quickly; thus improving the reach of this project. To improve this in the future, it would be beneficial to set up an inexpensive computer network so that the processes could be spread and therefore carried out with greater speed. However this could possibly involve editing the code which poses a greater risk of errors occurring.

6.2 Errors

It is possible that human error existed during this experiment as the values were inputted each time by the experimenter. This means that the experiments potentially suffered the effects of mistakes during typing, for instance selecting the wrong numeric value. However, due to the edit to the code which ensured that the folder naming occurred automatically, these mistakes would have been evident and therefore filtered out. This means that there exists a small chance that results for an experiment are inaccurate.

6.2.1 Automation

It is because of errors such as these that automating the input values would be beneficial, as human error could be avoided. This would save time also, as once one experiment had finished, the next would begin immediately. While during experimentation, estimations had to be made about when the next experiment was likely to conclude, this would change as the size of the parameters inputted did. Automating the process of producing data, and then collating it into a program like Microsoft Excel, or Mathworks MATLAB would also save time and ensure that results could be produced more quickly, but this would involve editing the code with each change.

6.3 Reliability

Elements of this research was provided by other scientists and industrial partners, as is expected in an interdisciplinary research area. However it is because others were responsible for providing the information which the research is based upon, that it may not be completely reliable. This is the case with values provided for *impact* and *Likelihood.* Due to the fact that different designers may have a varied understanding as to the quantity of risk of different elements; which could stem from their distinct opinions and experience. Overall means that data the results are being compared to is not entirely accurate. However, because this research aims to give a statistical understanding of how to guide designers, accuracy is not necessarily that important because probability is involved. It is also possible that a bug exists within the code, but during experimentation the only evidence of this is the fact that the code very occasionally became unresponsive and would not progress during experimentation. Yet this was a rare action which occurred only a few times during the 2000+ experiments.

6.4 Future of Research

This research area is flourishing as industry approaches the future. New materials which will unlock many new possibilities and release the ability to involve older ones in a way which had previously been difficult. Or to use pre-existing materials in a completely different way. While new requirements will begin to exist in a way they never have done before - such as the *impact* upon the ecosystem or restrictions on safety. Factors like these will push the engineering industry to warp and adapt at a fast pace. All of this means that the design process must also adapt in order to cope with in influx of new technology such as software which will ultimately mean engineers will become closer to ensuring that the design process is as simple, easy, and effective as possible. However there is still work to do until this point, but research like this will aid the progress to reaching that point.

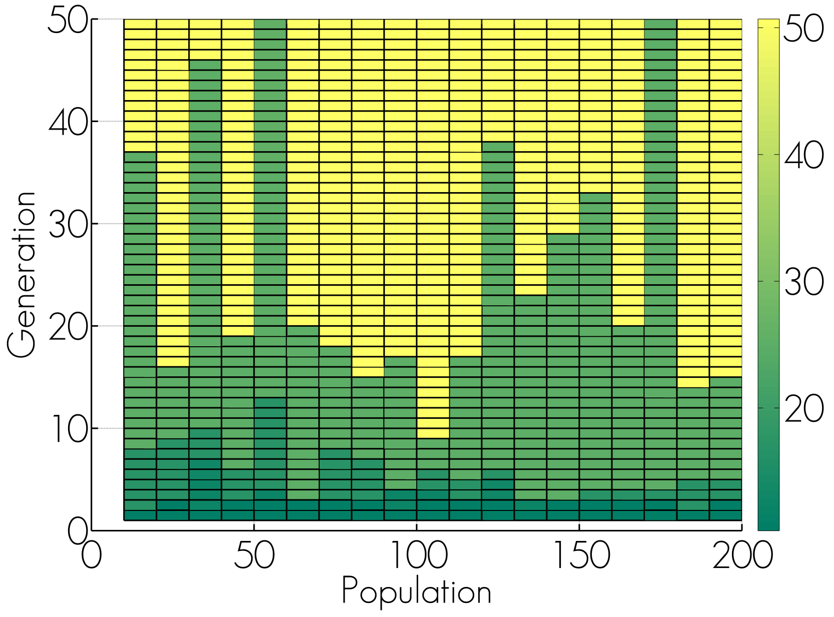


Figure 24: This graph shows the 3D graph in iteration one, from above

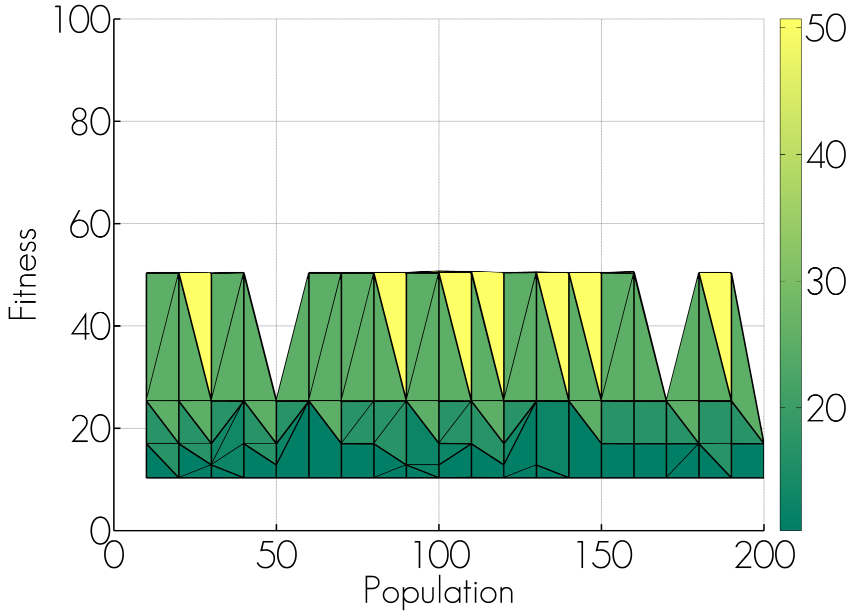


Figure 25: This graph also shows the 3D graph from iteration one from the X-Y view.

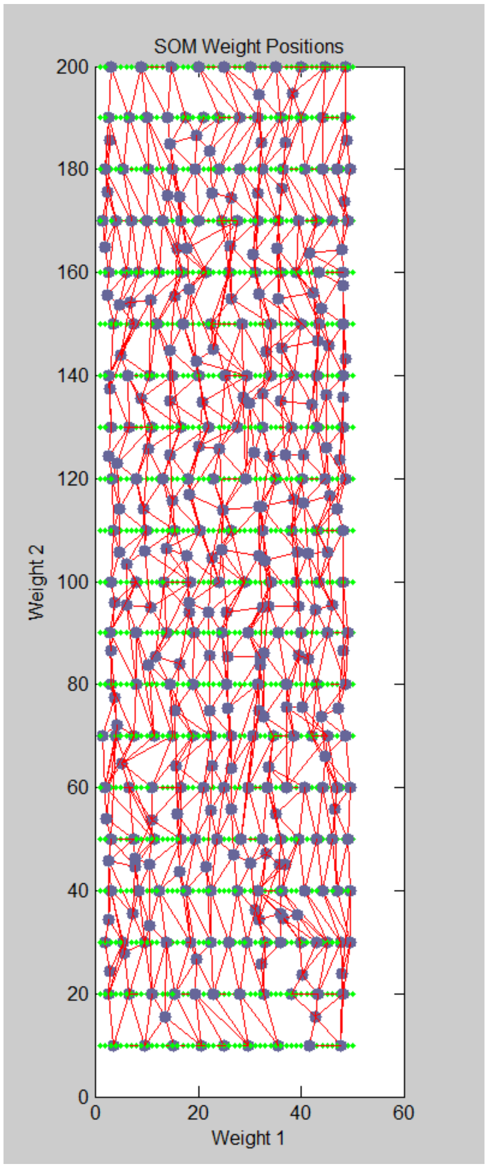


Figure 26: This graph shows a SOM, created with the values from iteration one. Each point correlates to a data point, and the deformations show the distribution of data.

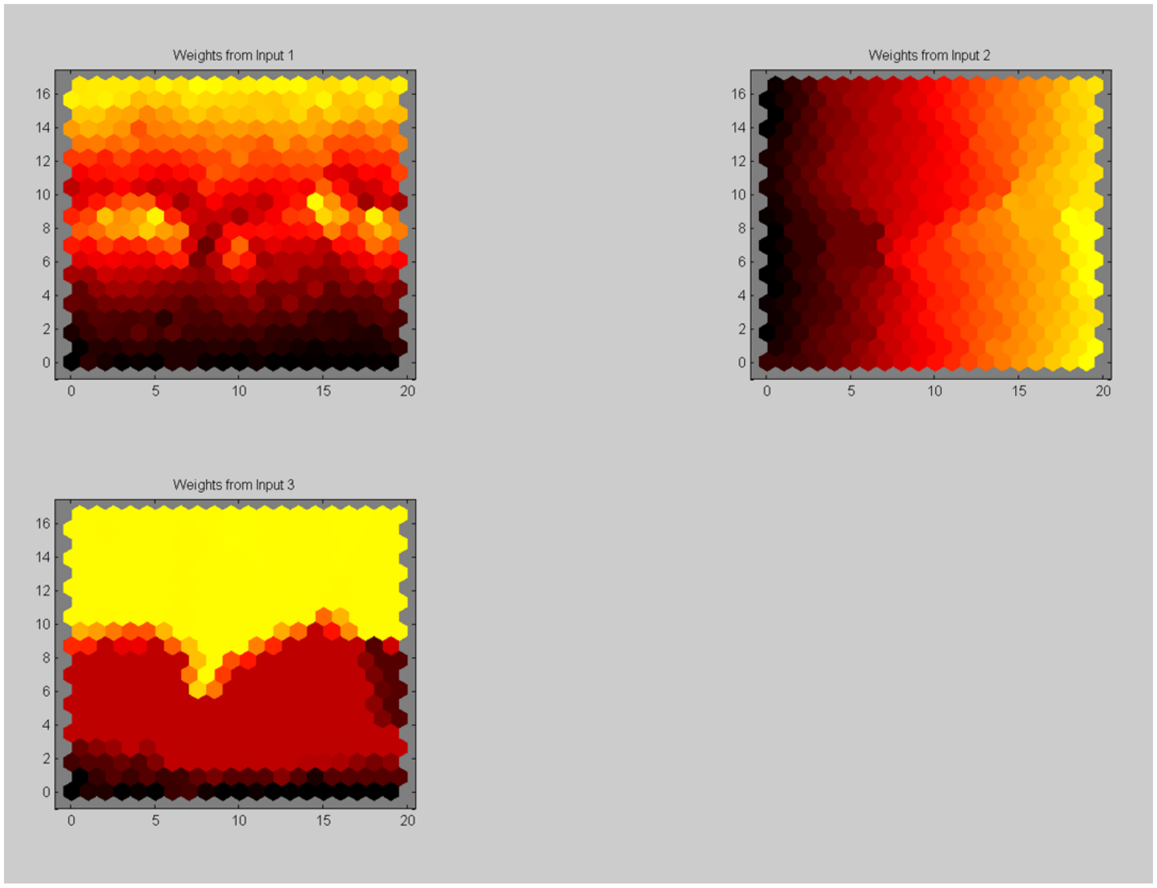


Figure 27: This set of graphs show the weights from each input vector. Smooth graduations is colour indicate a consistent change in values. Input one is the generation, input two is the population size, and input three are the fitness values.

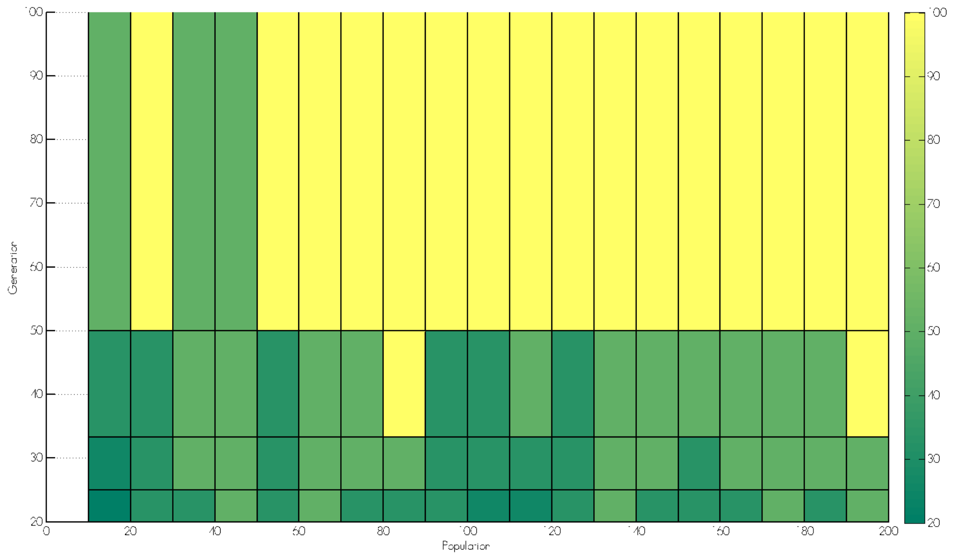


Figure 28: This image shows the data from iteration two - development, initially shows as a 3D graph, this shows a 2D view from above

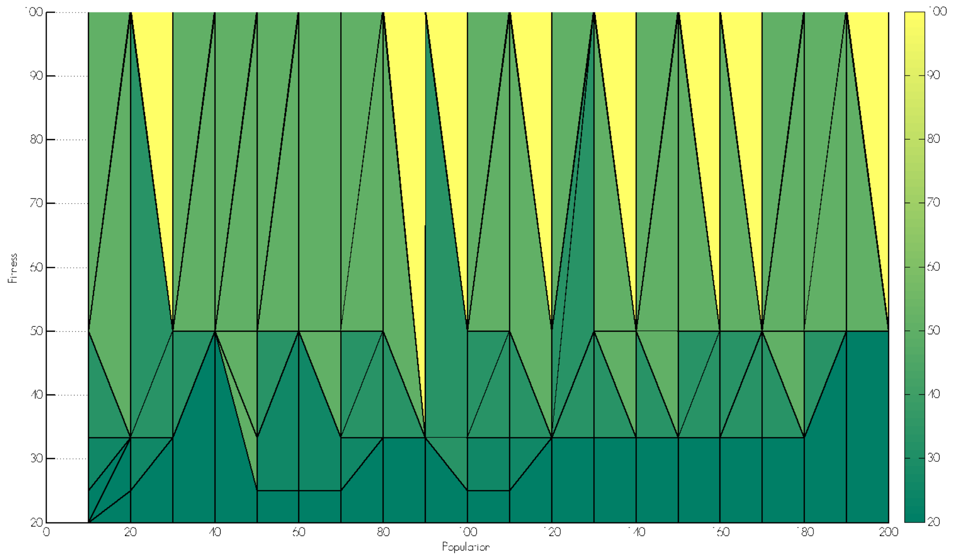


Figure 29: This image shows the data from iteration two - development, initially shows as a 3D graph, this shows an X-Y view.

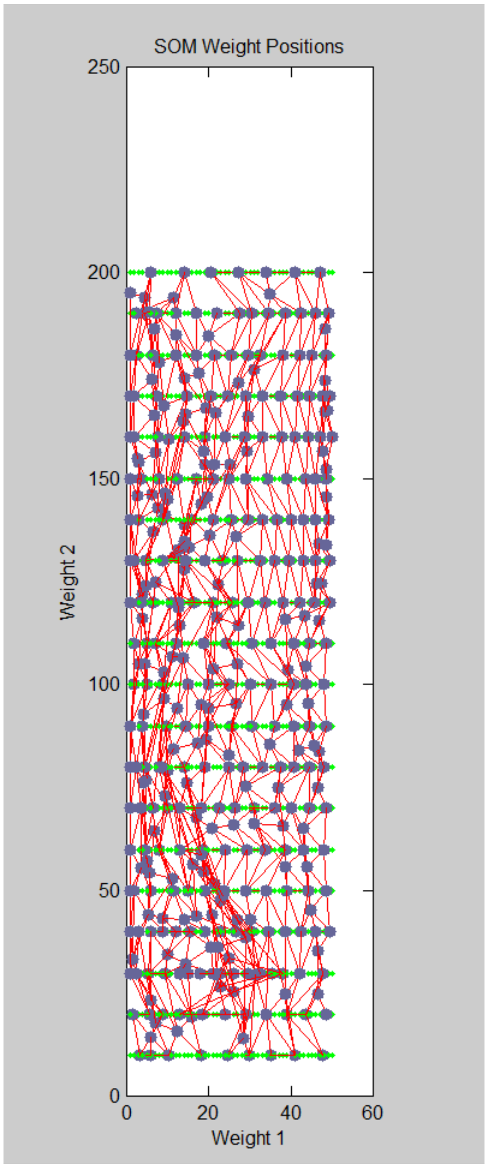


Figure 30: This graph shows a SOM, created with the values from iteration two - development. Each point correlates to a data point, and the deformations show the distribution of data.

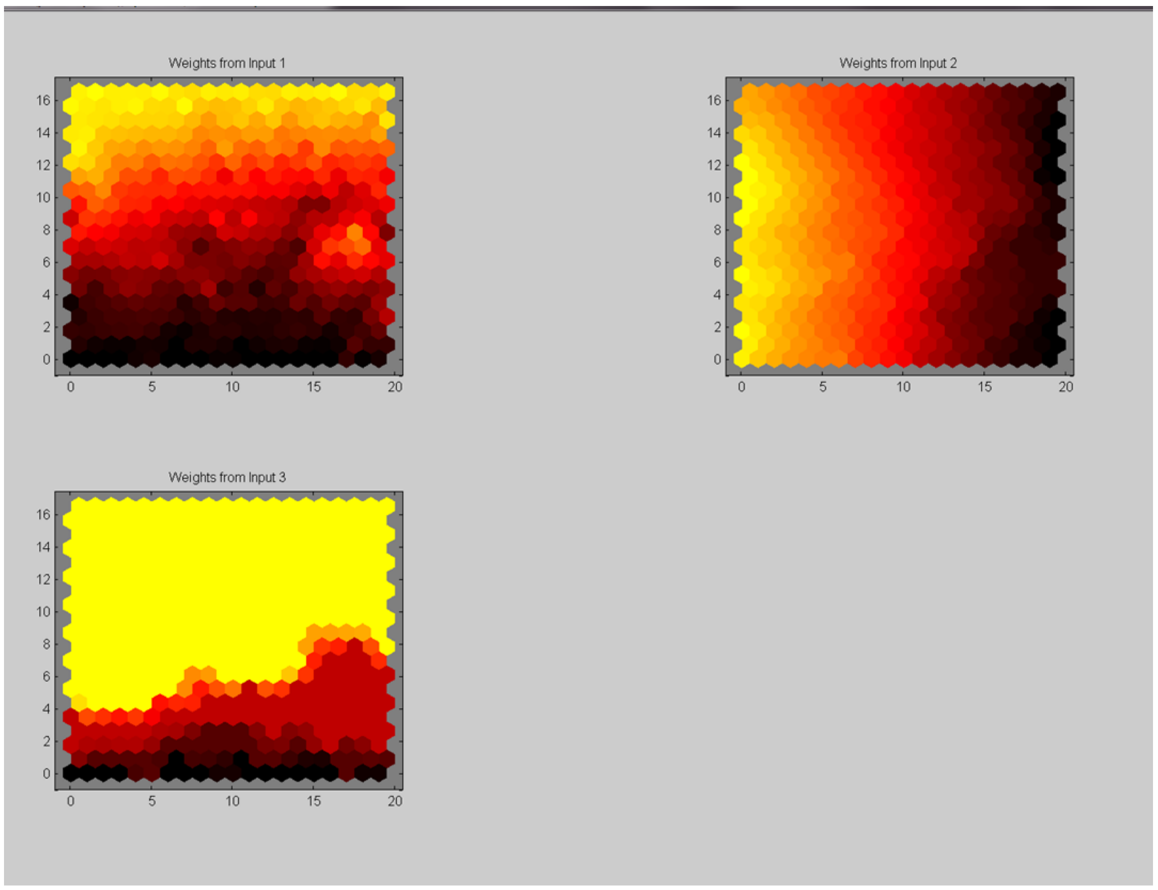


Figure 31: This set of graphs show the weights from each input vector. Smooth graduations is colour indicate a consistent change in values. Input one is the generation, input two is the population size, and input three are the fitness values.

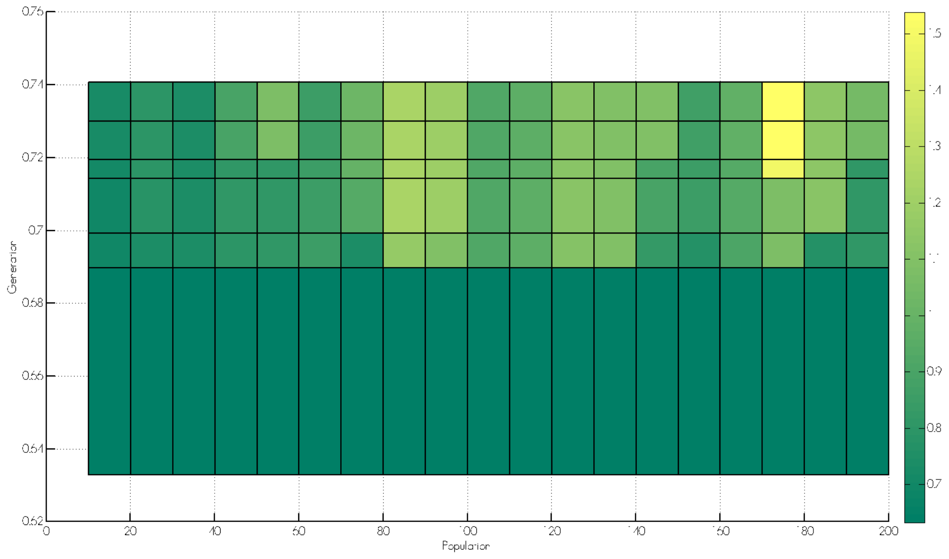


Figure 32: This image shows the data from iteration two -design, initially shows as a 3D graph, this shows a 2D view from above

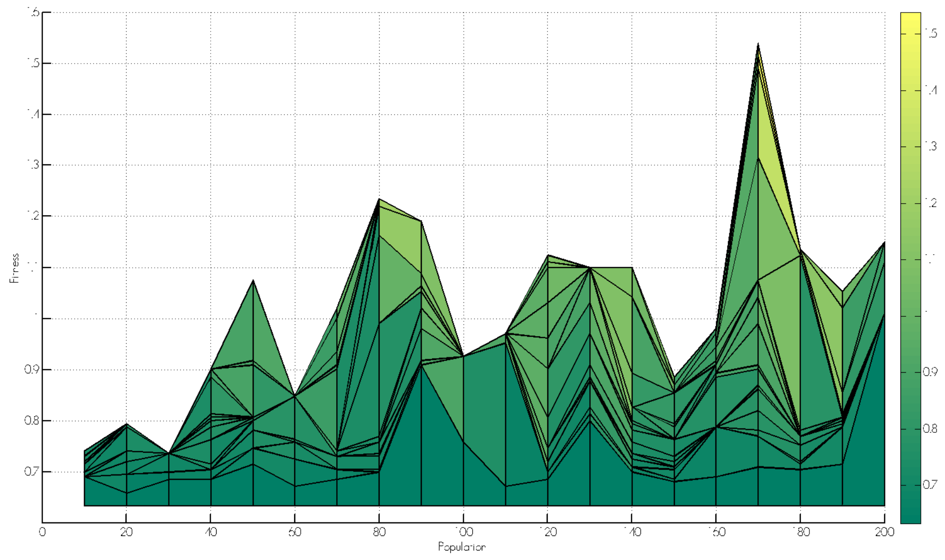


Figure 33: This image shows the data from iteration two - design, initially shows as a 3d graph, this shows an X-Y view.

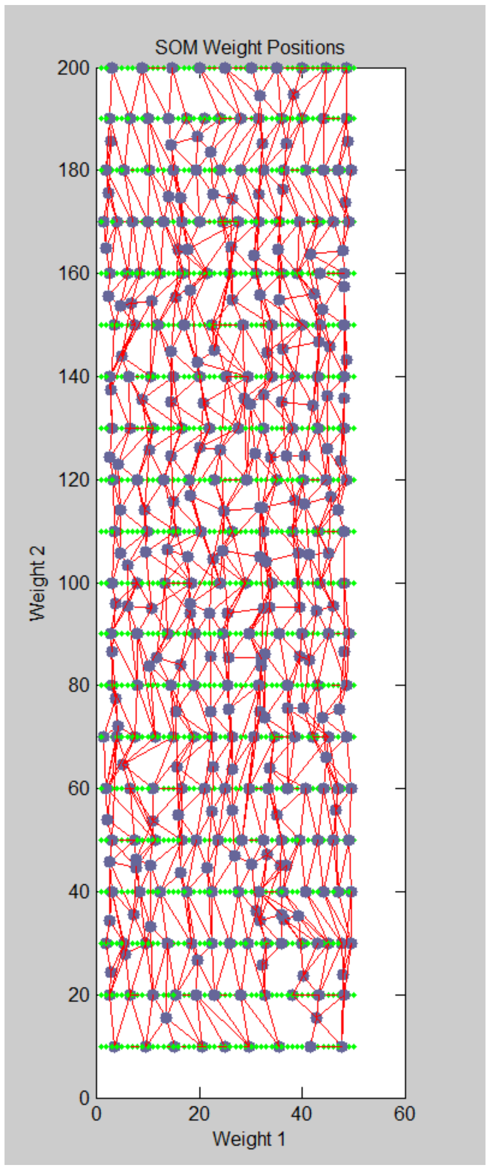


Figure 34: This graph shows a SOM, created with the values from iteration two - design. Each point correlates to a data point, and the deformations show the distribution of data.

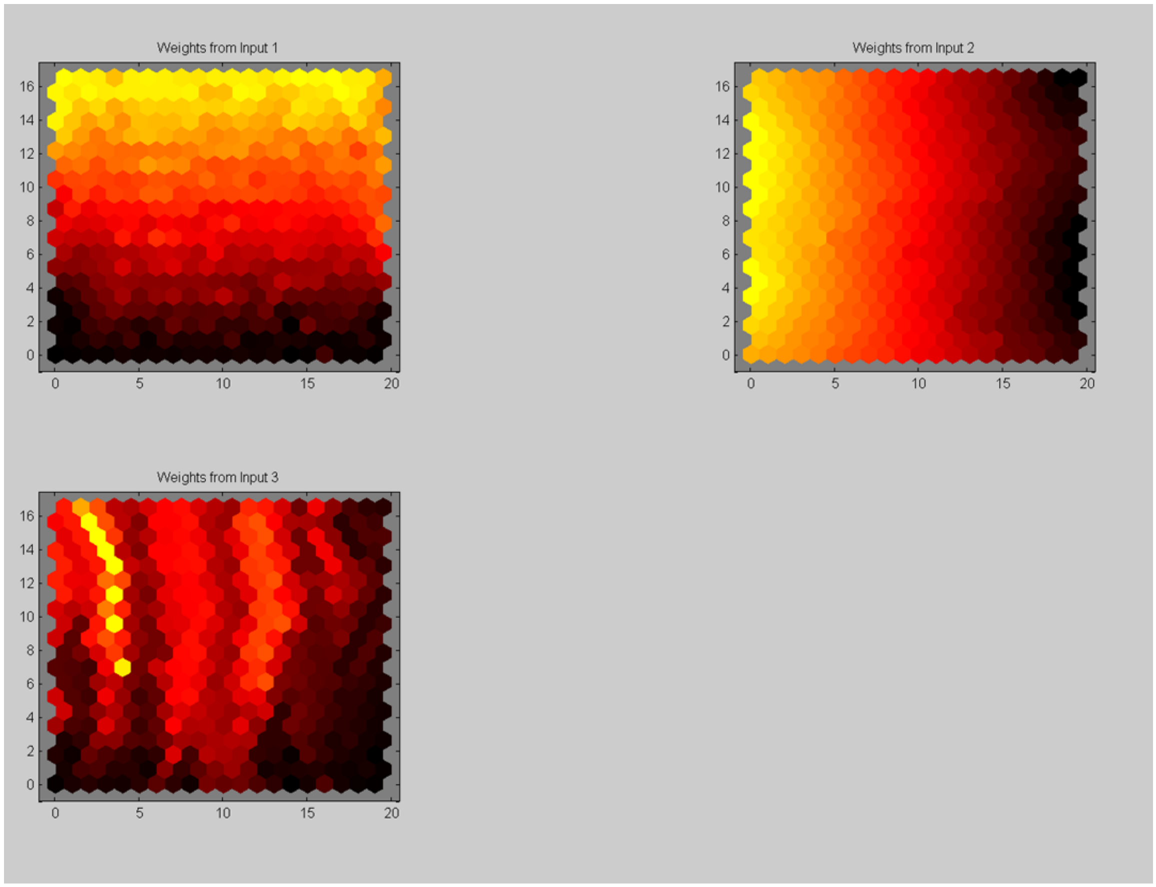


Figure 35: This set of graphs show the weights from each input vector. Smooth graduations is colour indicate a consistent change in values. Input one is the generation, input two is the population size, and input three are the fitness values.

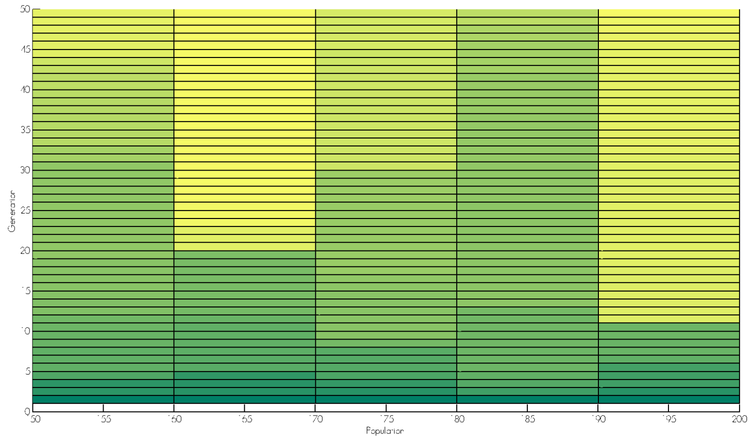


Figure 36: 3D view of iteration four results from above.

```
data = importdata('Filename.csv');

% data stores all the data from the CSV file so need to extract popuation
% population and generation

pop = data(1,2:end);
%row 1 is population, col 2 till the end
gen = data(2:end,1);
%col 1 is generation, row 2 till the end
fitness = data(2:end,2:end);
%the rest is fitness

%FOR PLOTTING 3D
surf(pop, gen, fitness);

colormap summer;
%alpha .4;

xlabel('Population');
ylabel('Generation');
zlabel('Fitness');

function [newData1] = importfile(fileToRead1)
%IMPORTFILE(FILETOREAD1)
%  Imports data from the specified file
%  FILETOREAD1:  file to read

%  Auto-generated by MATLAB on 10-Aug-2015 09:39:57

% Import the file
sheetName='Sheet1';
[numbers, strings] = xlsread(fileToRead1, sheetName);
if ~isempty(numbers)
    newData1.data =  numbers;
end
if ~isempty(strings)
    newData1.textdata =  strings;
end
```

References

[1] Tom Gilb. Competitive engineering. *Competitive Engineering. Elsevier*, 2005.

[2] Nigel Cross. *Engineering design methods: strategies for product design.* John Wiley & Sons, 2008.

[3] P John Clarkson, Caroline Simons, and Claudia Eckert. Predicting change propagation in complex design. *Journal of Mechanical Design*, 126(5):788–797, 2004.

[4] René Keller et al. *Predicting change propagation: algorithms, representations, software tools.* PhD thesis, Cambridge University Engineering Department, 2007.

[5] Roger Maull, David Hughes, and Jan Bennett. The role of the bill-of-materials as a cad/capm interface and the key importance of engineering change control. *Computing & Control Engineering Journal*, 3(2):63–70, 1992.

[6] Ernst Fricke, Bernd Gebhard, Herbert Negele, and Eduard Igenbergs. Coping with changes: causes, findings, and strategies. *Systems Engineering*, 3(4):169–179, 2000.

[7] Matthias Kreimeyer, Frank Deubzer, Ulrich Herfeld, and Udo Lindemann. A survey on efficient collaboration of design and simulation in product development. In *23rd CADFEM Users' Meeting 2005, International Congress on FEM Technology with ANSYS CFX & ICEM CFD Conference, International Congress Center Bundeshaus Bonn, Germany*, 2005.

[8] Michael D Devine, Thomas E James Jr, and Mr Timothy I Adams. Government supported industry-university research centers: issues for successful technology transfer. *The Journal of Technology Transfer*, 12(1):27–37, 1987.

[9] Louis L Bucciarelli. *Designing engineers.* MIT press, 1994.

[10] Gerhard Pahl and Wolfgang Beitz. *Engineering design: a systematic approach.* Springer Science & Business Media, 2013.

[11] Tom Gilb and Susannah Finzi. *Principles of software engineering management*, volume 4. Addison-Wesley Reading, MA, 1988.

[12] Sharon K Parker, Carolyn M Axtell, and Nick Turner. Designing a safer workplace: Importance of job autonomy, communication quality, and supportive supervisors. *Journal of Occupational Health Psychology*, 6(3):211, 2001.

[13] Chris Argyris. Good communication that blocks learning. *Harvard Business Review*, 72(4):77–85, 1994.

[14] Offer Shai. Design through common graph representations. In *ASME Design Engineering Technical Conferences, Chicago*, pages 2–6, 2003.

[15] Yong-Zheng Zhang, Bin-Xing Fang, Yue Chi, and Xiao-Chun Yun. Risk propagation model for assessing network informationsystems. *Ruan Jian Xue Bao(Journal of Software)*, 18(1):137–145, 2007.

[16] Joseph P Forgas and Jennifer M George. Affective influences on judgments and behavior in organizations: An information processing perspective. *Organizational behavior and human decision processes*, 86(1):3–34, 2001.

[17] Manfred Broy. Challenges in automotive software engineering. In *Proceedings of the 28th international conference on Software engineering*, pages 33–42. ACM, 2006.

[18] R Keller, T Eger, CM Eckert, PJ Clarkson, et al. Visualising change propagation. In *DS 35: Proceedings ICED 05, the 15th International Conference on Engineering Design, Melbourne, Australia, 15.-18.08. 2005*, 2005.

[19] Steven D Eppinger and Tyson R Browning. *Design structure matrix methods and applications*. MIT press, 2012.

[20] Tyson R Browning. Applying the design structure matrix to system decomposition and integration problems: a review and new directions. *Engineering Management, IEEE Transactions on*, 48(3):292–306, 2001.

[21] Ali Yassine and Dan Braha. Complex concurrent engineering and the design structure matrix method. *Concurrent Engineering*, 11(3):165–176, 2003.

[22] Mike Danilovic and Tyson R Browning. Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management*, 25(3):300–314, 2007.

[23] Amira Sharon, Dov Dori, and Olivier De Weck. Model-based design structure matrix: deriving a dsm from an object-process model. In *Second International Symposium on Engineering Systems*, pages 1–12, 2009.

[24] Bharath Shekar, R Venkataram, and BM Satish. Managing complexity in aircraft design using design structure matrix. *Concurrent Engineering*, page 1063293X11426461, 2011.

[25] Mike Danilovic and Bengt Sandkull. The use of dependence structure matrix and domain mapping matrix in managing uncertainty in multiple project situations. *International journal of project management*, 23(3):193–203, 2005.

[26] Steven D Eppinger, Daniel E Whitney, and David A Gebala. Organizing the tasks in complex design projects: Development of tools to represent design procedures. In *NSF Design and Manufacturing Systems Conference, Atlanta, Georgia*, 1992.

[27] A Yassine. An introduction to modeling and analyzing complex product development processes using the design structure matrix (dsm) method. *Urbana*, 51(9):1–17, 2004.

[28] Roberto Almeida Bittencourt and Dalton Dario Serey Guerrero. Comparison of graph clustering algorithms for recovering software architecture module views. In *Software Maintenance and Reengineering, 2009. CSMR'09. 13th European Conference on*, pages 251–254. IEEE, 2009.

[29] Amr A Oloufa, Yasser A Hosni, Mohamed Fayez, and Pär Axelsson. Using dsm for modeling information flow in construction design projects. *Civil Engineering and Environmental Systems*, 21(2):105–125, 2004.

[30] Timothy Jarratt, Claudia Eckert, and P John Clarkson. Development of a product model to support engineering change management. *Proceedings of the TCME*, pages 331–344, 2004.

[31] CS Simons. *Change propagation in product design.* PhD thesis, 2000.

[32] TAW Jarratt, Claudia M Eckert, NHM Caldwell, and P John Clarkson. Engineering change: an overview and perspective on the literature. *Research in engineering design*, 22(2):103–124, 2011.

[33] Tim Jarratt, P John Clarkson, Geoff Parks, and Claudia Eckert. Use of monte carlo methods in the prediction of change propagation. In *Engineering Design Conference*, pages 487–498, 2002.

[34] Bilal M Ayyub. *Risk analysis in engineering and economics.* CRC Press, 2014.

[35] Piergiorgio Odifreddi. *Classical recursion theory: The theory of functions and sets of natural numbers.* Elsevier, 1992.

[36] PJ Clarkson, Caroline Simons, and Claudia Eckert. Change prediction for product redesign. In *International Conference on Engineering Design*, 2001.

[37] GB Dantzig. The nature of mathematical programming. *Mathematical Programming Glossary*, 2010.

[38] Evin J Cramer, JE Dennis, Jr, Paul D Frank, Robert Michael Lewis, and Gregory R Shubin. Problem formulation for multidisciplinary optimization. *SIAM Journal on Optimization*, 4(4):754–776, 1994.

[39] George A Hazelrigg. A framework for decision-based engineering design. *Journal of mechanical design*, 120(4):653–658, 1998.

[40] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.

[41] Mark W Krentel. The complexity of optimization problems. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 69–76. ACM, 1986.

[42] Thomas Bäck. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[43] Charles Darwin. *The origin of species*. GoodBook Classics, 1951.

[44] André Baresel, Harmen Sthamer, and Michael Schmidt. Fitness function design to improve evolutionary structural testing. In *GECCO*, volume 2, pages 1329–1336, 2002.

[45] Ji-Pyng Chiou and Feng-Sheng Wang. Hybrid method of evolutionary algorithms for static and dynamic optimization problems with application to a fed-batch fermentation process. *Computers & Chemical Engineering*, 23(9):1277–1291, 1999.

[46] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[47] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[48] David E Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms*, 1:69–93, 1991.

[49] Gilbert Syswerda. Uniform crossover in genetic algorithms. 1989.

[50] Mandavilli Srinivas and Lalit M Patnaik. Adaptive probabilities of crossover and mutation in genetic algorithms. *Systems, Man and Cybernetics, IEEE Transactions on*, 24(4):656–667, 1994.

[51] Esa Alhoniemi, Jaakko Hollmén, Olli Simula, and Juha Vesanto. Process monitoring and modeling using the self-organizing map. *Integr Comput Aided Eng*, 6(1):3–14, 1999.

[52] Teuvo Kohonen, Erkki Oja, Olli Simula, Ari Visa, and Jari Kangas. Engineering applications of the self-organizing map. *Proceedings of the IEEE*, 84(10):1358–1384, 1996.

[53] Teuvo Kohonen and Timo Honkela. Kohonen network. *Scholarpedia*, 2(1):1568, 2007.

[54] Mathworks Matlab. Neural Network Design Steps documentation, 2015.

[55] Juha Vesanto, Johan Himberg, Esa Alhoniemi, and Juha Parhankangas. Self-organizing map in matlab: the som toolbox. In *Proceedings of the Matlab DSP conference*, volume 99, pages 16–17, 1999.

[56] Matlab Documentation. The mathworks inc, 2005.

[57] MATLAB Documentation. Fuzzy toolbox user's guide of matlab. the mathworks, 2002.

[58] Teuvo Kohonen. The self-organizing map. *Neurocomputing*, 21(1):1–6, 1998.