DESIGN SPACE VISUALIZATION

A Dissertation submitted to the University of Hyderabad in partial fulfillment of the degree of

MASTER OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE

by

ASHOK SAYAM

DEPARTMENT OF COMPUTER & INFORMATION SCIENCES

SCHOOL OF MATHEMATICS & COMPUTER / INFORMATION SCIENCES

University of Hyderabad
(P.O.) Central University, Gachibowli
Hyderabad – 500 046
Andhra Pradesh
India

CERTIFICATE

This is to certify that the dissertation entitled "**Design Space Visualization"** submitted by **ASHOK SAYAM** bearing Reg. No **09MCMI13** in partial fulfillment of the requirements for the award of **Master of Technology** in **Artificial Intelligence** is a bonafide work carried out by him under my supervision and guidance.

The dissertation has not been submitted previously in part or in full to this or any other University or Institution for the award of any degree or diploma.

Prof. C. R Rao
Project Supervisor,
Department of CIS,
University of Hyderabad.

Head of the Department,
Department of CIS,
University of Hyderabad.

Dean,
School of MCIS,
University of Hyderabad.

DECLARATION

I, **ASHOK SAYAM** hereby declare that this Dissertation entitled "**Design Space Visualization**" submitted by me under the guidance and supervision of **Prof. C.R Rao** is a bonafide work. I also declare that it has not been submitted previously in part or in full to this University or other University or Institution for the award of any degree or diploma.

Date: **ASHOK SAYAM**

09MCMI13

Acknowledgement

It is a great pleasure to acknowledge the support and encouragement received in the successful completion of this project. It is a privilege to express my profound gratitude and indebtedness to my supervisor **Prof. C.R Rao** for his valuable, inspiring and constant support throughout the progress of this project.

I am very much thankful to **Mr. P.S.V.S Sai Prasad** for the continuous guidance provided throughout the project.

I am grateful to the Head of Department (DCIS), **Prof**. **C.R Rao** for providing us both the freedom and an excellent lab facility where we could test our work effectively.

I am extremely thankful to **A.I Lab** staff for their kind cooperation. A special thanks to all my friends, especially my lab mates for their valuable suggestions and encouragement.

I express my gratitude to all my family members, friends for their love and affection, without which this work would not have been possible. Finally, I would like to thank God, who made all the things possible.

ASHOK SAYAM

Abstract

Design Space Visualization (DSV) helps the analyst to gain insights into the design space under consideration with the generation of plots, figures and understanding the key relationships between inputs and outputs of interest. This project helps in finding *interesting regions* in the design space of a particular system.

We used Principal Component Analysis (PCA) and Self Organizing Maps (SOM) as the visualization techniques to investigate the design space under consideration. PCA, which is mainly used for dimensionality reduction, is used in DSV by performing clustering operations (k-means) on the generated principal component scores. It shows us how our data is spread in the form of clusters. It gives an idea of mapping interested object (dimension) of the system on the generated Voronoi Tessellation diagram, so that we can select the components or area into which this falls in.

SOM is one type of artificial neural network which is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map. SOM has been successfully applied in data mining and visualization. It is a model for clustering and visualizing high-dimensional data, which groups the data with similar characteristics.

Using the SOM's U-matrix technique, we can visualize the cluster characteristics of the data. The bar plane map gives an idea of which component value is dominating in a particular region. The Component Plane map gives the visualization between design variables and objective function. A SOM GUI is developed, with the help of which we can select the region of interest in the design space. This gives the range of the variables and enables us to concentrate only on the required design space, leaving the uninteresting regions, which decreases the range of the variables thus decreasing the total search space and helps us to choose values for the variables which optimize the objective function. This helps in reducing the design search space, saving the product design time and promoting the design efficiency.

Table of Contents

Abstract v

Table of Contents vi

List of Figures viii

List of Tables ix

1. Introduction to Design Space Visualization (DSV) 1

1.1 What is DSV? 1

1.1.1 Types of Design. 1

1.2 Why DSV? 2

1.3 Objective. 2

1.4 Organization of Thesis. 3

2. Multi Disciplinary Optimization (MDO) 4

2.1 Definition of MDO. 4

2.2 Why MDO? 5

2.3 Problem Formulation in MDO. 6

2.4 Sample problems in MDO. 11

2.5 Steps in MDO. 13

2.6 Challenges in MDO 13

2.7 Advantages & Disadvantages 14

3. Design of Experiments (DoE) 15

3.1 Brief Introduction. 15

3.2 Techniques useful in DoE. 15

4. Principal Component Analysis and Self-Organizing Maps 17

4.1 Principal Component Analysis (PCA) 17

4.2 Self-Organizing Maps (SOM) 18

4.2.1 Working of SOM 19

4.2.2 SOM Algorithm 20

4.2.3 Differences between SOM and K-means 21
5. Visualizations using Principal Component Analysis and Self-Organizing Maı 22
5.1 Principal Component Analysis (PCA) 22
5.1.1 Clustering 22
5.1.2 Voronoi Tessellation 22
5.2 Self-Organizing Maps (SOM) 22
5.2.1 Structure and Shape. 23
5.2.2 Properties of the prototype vector set. 25
5.2.3 Examining new data with the map. 26
5.3 Graphical User Interface (GUI) for selection of cells in SOM 28
5.4 Limitations of SOM 29
6. Results and Discussions 30
6.1 visualizations by PCA 31
6.2 visualizations by SOM 39
7. Conclusion and Future Work 55
References 56
Appendix

List of Figures **Page No**

2.1 MDO aspects of design. 5
2.2 Two-variable design space of a rectangular wing. 8
2.3 Eight Member Hub Frame Assembly with a Single Load Case. 12
4.1 Figure showing PC components. 17
4.2 Figure showing SOM. 19
5.1 U-matrix representation showing the spread of data. 24
5.2 Figure showing U-matrix and component planes. 25
5.3 Figure showing the number of hits by data in each cell. 26
5.4 Figure showing GUI for SOM map showing selection of a region on the map 28
5.5 Figure showing selection of classes on the GUI. 29
6.1 Figure showing the plots between attributes of the data. 31
6.2 Figure showing clusters in the data. 32
6.3 Figure showing Voronoi Tessellations. 33
6.4 Figure showing the plots between attributes of the data. 34
6.5 Figure showing clusters in the data. 35
6.6 Figure showing Voronoi Tessellations. 36
6.7 Figure showing U-matrix, Component planes and Labels of the data. 37
6.8 Figure showing all Components of the data in the bar plane. 38
6.9 Figure showing U-matrix, Component planes and Labels of the data. 39
6.10 Figure showing all Components of the data in the bar plane. 40
6.11 Figure showing U-matrix, Component planes and Labels of the data. 42
6.12 Figure showing all Components of the data in the bar plane. 44
6.13 Figure showing U-matrix, Component planes and Objective function of the data. 45
6.14 Figure showing the number of hits by data in each cell. 46
6.15 Figure showing U-matrix, Component planes and Objective function of the data. 48
6.16 Figure showing the number of hits by data in each cell. 50
6.17 Figure showing U-matrix, Component planes and objective function of the data. 52
6.18 Figure showing the number of hits by data in each cell. 54

List of tables **Page No**

6.1 Showing results obtained, estimated reduction in search space of individual Components of De Jong's function. 49

6.2 Showing results obtained estimated reduction in search space of individual Components of Axis Hyper Ellipsoid function. 53

CHAPTER 1

Introduction

1.1 What is Design Space Visualization?

To understand a complex engineering system of real world, first it needs to be modelled, simulated and then it is manufactured. Complex engineering systems like aerospace wing design, Geostationary Platform design etc., include design variables from multiple disciplines. In the design process the effect of variables in one discipline may unknowingly affect the design variables in other disciplines resulting in depriving overall design and efficiency of the system. Especially for these kinds of problems a method called Multi-Disciplinary Optimization (MDO) is used.

In design space visualization an MDO problem is taken and it's design variables are identified and with the help of visualization techniques by generating plots, graphs, figures etc., relations between design variables are visualized and interesting regions in the design space are identified.

1.1.1 Types of Design

Forward design:

Forward design starts from design space (design variables and their values) to performance space (evaluation of the products' performance) of complex products. In traditional optimization, a sampled point is randomly selected to start the searching process, and then performance of this solution is evaluated. This type of design is known as forward design.

Backward design:

To focus on smaller interesting design regions, a new method which maps from performance space (evaluation of the products' performance) to design space (design variables and their values) has emerged, called as backward design. Use of an

expert system with the past design knowledge helps engineers to focus on interesting regions rapidly in the whole design space at the beginning of product optimization [1].

1.2 Why Design Space Visualization?

Design Space Visualization helps the analyst to gain insights into the design space under consideration with the generation of plots, figures and understanding the key relationships between inputs and outputs of interest.

In the modern practical engineering field, complex products/systems such as automobiles, aircrafts and ships are usually quite difficult to be configured and designed, the reason for this is a large amount of multidisciplinary knowledge and expertise must be mastered by designers, and huge design space must be exploited entirely to get appropriate design solutions.

- The Backward design explained above helps the engineers to use the successful constructed products' data and design experience.
- Helps engineers to focus on interesting regions rapidly in the whole design space at the beginning of product design, which makes the succeeding strict optimization more quickly and efficiently.
- It reduces the searching scope of design space by concentrating only on the interesting regions.
- It saves product design time.
- It promotes design efficiency.

1.3 Objective

By applying visualization techniques like PCA, SOM and by building SOM visualization tools on multi-dimensional datasets and by generating plots, figures we should find out the key relationships between design variables and objective functions by finding, how the design variables are responding to the objective functions.

1.4 Organization of Thesis

Chapter 1 deals with design space visualization and what is the need for it. Chapter 2 gives a brief description of Multi-disciplinary Optimization (MDO). Chapter3 deals with Design of Experiments, a set of statistical techniques to deal with MDO problems. Chapter 4 gives a brief idea about Principal Component Analysis (PCA) and Self-Organizing Maps (SOM). Chapter 5 deals with how visualizations could be done using PCA and SOM. Chapter 6 gives the in-depth analysis carried out on different datasets and shows how design search space can be reduced by using the above visualization techniques. Chapter 7 gives the conclusion to this dissertation and the future work that can be carried out.

CHAPTER 2

MULTI-DISCIPLINARY DESIGN OPTIMIZATION (MDO)

2.1 Brief Description

Most modern engineering systems are multidisciplinary and their analysis is often very complex, involving hundreds of computer programs, many people in different locations. Modern design projects are more complex and problem has to be decomposed and each part tackled by a different team.

MDO is a methodology for the design of complex engineering systems and subsystems that coherently exploits the synergism of mutually interacting phenomena. It is also known as multidisciplinary optimization and multidisciplinary system design optimization (MSDO) [4].

MDO is a field of engineering that uses optimization methods to solve design problems incorporating a number of disciplines. MDO techniques have been used in a number of fields, including automobile design, naval architecture, electronics, computers, and electricity distribution. The largest number of applications has been in the field of aerospace engineering, such as aircraft and spacecraft design.

For example, the Boeing blended wing body (BWB) aircraft concept has used MDO extensively in the conceptual and preliminary design stages. The disciplines considered in the BWB design are aerodynamics, structural analysis, propulsion, control theory, and economics.

- **Multidisciplinary** - comprised of more than one traditional disciplinary area described by governing equations from various physical, economic, social fields.
- **System** - A system is a physical or virtual object that exhibits some behavior or performs some function as a consequence of interactions between the constituent elements.
- **Design** - The process of conceiving and planning an object or process with a specific goal in mind.
- **Optimization** - To find a system design that will minimize some objective function.

2.2 Why MDO

Current design processes of a complex coupled system generally rely on sequential decision making through the traditional route starting with conceptual design, going on to the preliminary design phase and then to the detailed design, wherein each subsequent phase sees a significantly reduced design freedom as a consequence of decisions frozen in the earlier phase. In a complex coupled system, when interdisciplinary conflicts arise, current design practice relies on non-automated means to resolve these, which might lead to sub-optimal designs in a global sense.

Whereas, MDO combines analysis and optimizations in several individual disciplines with those of the entire system concurrently, through formal mathematical processes. It puts into place a formal integrated system design process for better product quality by effectively exploiting the synergism of interdisciplinary couplings. MDO, as a discipline, itself comprises of many areas of research such as the formal MDO problem formulation, architectures and methodologies for decomposition; approximation concepts and design oriented disciplinary analyses; software frameworks for implementation of integrated analysis, optimization, data management and communications; interfaces for human expertise and heuristic knowledge; etc.

a) Multi Disciplinary Aspects of Design:

Emphasis is on the multidisciplinary nature of the complex engineering systems design process. Aerospace vehicles are a particular class of such systems.

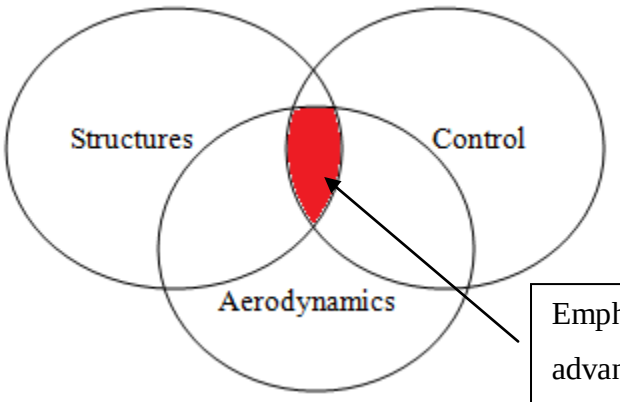


Fig: 2.1 MDO aspects of design.

b) Why system-level, multidisciplinary optimization?

Disciplinary specialists tend to strive towards improvement of objectives and satisfaction of constraints in terms of the variables of their own discipline.

In doing so they generate side effects - often unknowingly that other disciplines have to absorb, leading to the detriment of the overall system performance. Example: High wing aspect ratio.

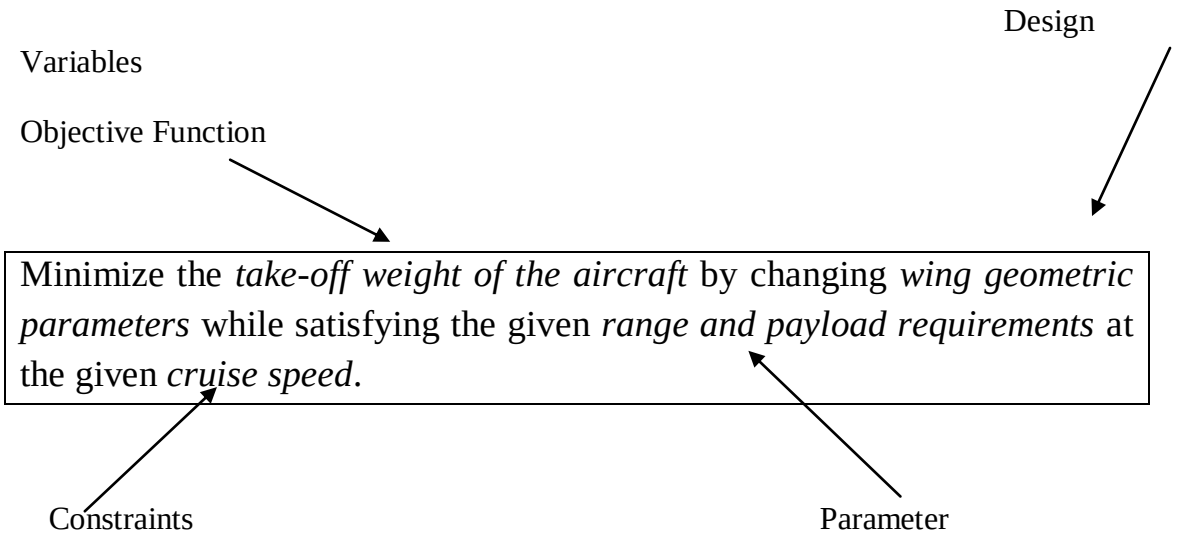
c) Supporting Disciplines include:

Multidisciplinary design optimization of complex systems cannot take place without substantial contributions from supporting disciplines like:

- ✓ Human Interface Aspects of Design
- ✓ Intelligent and Knowledge-Based Systems
- ✓ Computing Aspects of Design
- ✓ Information Integration and Management

2.3 MDO Problem Formulation:

Example problem



Design Variables:

- A Design vector ***x*** contains ***n*** variables that form the design space.
- Design Variables are numbers whose values can be freely varied by the designer to define a designed object.
- Design variables are often bounded, that is, they often have maximum and minimum values. Depending on the solution method, these bounds can be treated as constraints or separately.

For instance, the thickness of a structural member can be considered a design variable. Another might be the choice of material. Design variables can be continuous (such as a wing span), discrete (such as the number of ribs in a wing), or boolean (such as whether to build a monoplane or a biplane).

Example:

$$\mathbf{X} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_i \\ . \\ . \\ x_n \end{pmatrix} = \begin{pmatrix} \text{aspect ratio [-]} \\ \text{transmit power [W]} \\ \text{\# of apertures [-]} \\ \text{orbital altitude [km]} \\ . \\ . \\ \text{control gain [V/V]} \end{pmatrix}$$

X_i can be …

Real : $x_i \in R$

Integer: $x_i \in Z$

Boolean: $x_i \in \{T,F\}$

Binary: $x_i \in \{0,1\}$

Design Space example:

As a very simple example, consider a rectangular wing with a pre-defined airfoil. It can be defined by deciding on the values of the following two design variables:

{b, c}

Placing these variables along orthogonal axes, it define a design space, or set of all possible design options. Each point in the design space corresponds to a chosen design, as illustrated in the Fig 2.2.

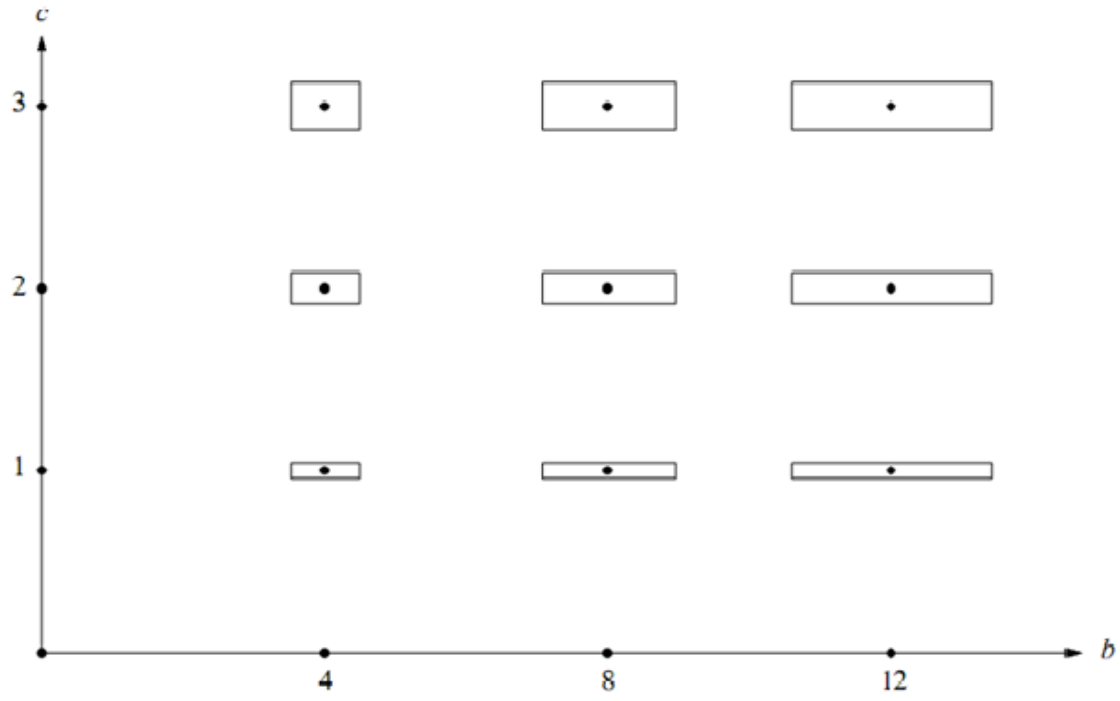


Fig: 2.2 Two-variable design space of a rectangular wing.

Objectives:

The objective can be a vector J of z system responses or characteristics we are trying to maximize or minimize.

$$J = \begin{pmatrix} J_1 \\ J_2 \\ J_3 \\ J_i \\ . \\ . \\ J_z \end{pmatrix} = \begin{pmatrix} \text{Cost [\$]} \\ \text{Range [km]} \\ \text{Weight [kgs]} \\ \text{Height [mt]} \\ . \\ . \\ \text{Data rate [bps]} \end{pmatrix}$$

Often the objective is a Scalar function, but for real systems often attempt Multi-objective optimization:

$X \longmapsto J(X)$

Some objectives can be Conflicting.

An objective is a numerical value that is to be maximized or minimized. For example, a designer may wish to maximize profit or minimize weight. Many solution methods work only with single objectives. When using these methods, the designer normally weights the various objectives and sums them to form a single objective. Other methods allow multiobjective optimization, such as the calculation of a Pareto front.

Objective function is a "measure of goodness" that enables us to compare two designs quantitatively.

Response Variables:

These are the variables other than the objective functions, which we will be observing in the design space.

Parameters:

Parameters p are quantities that affect the objective J, but are considered fixed, i.e. they cannot be changed by the designers.

Sometimes parameters p

(1) can be turned into design variables x to enlarge the design space.
(2) are former design variables that were fixed at some value because they were found not to affect any of the objectives J or because their optimal level was predetermined.

Constraints:

Constraints act as boundaries of the design space x and typically occur due to finiteness of resources or technological limitations of some design variables. Often, but not always, optimal designs lie at the intersection of several active constraints.

A constraint is a condition that must be satisfied in order for the design to be feasible. An example of a constraint in aircraft design is that the lift generated by a wing must be equal to the weight of the aircraft. In addition to physical laws, constraints can reflect resource limitations, user requirements, or bounds on the validity of the analysis models.

Inequality Constraints: $g_j(x) <= 0$ $j=1, 2,\ldots., m1$

Equality Constraints: $h_k(x) = 0$ $k=1, 2,\ldots., m2$

Bounds: $x_{i.LB} <= x_i <= x_{i.UB}$ $i=1,2,\ldots,n$

Constraints versus Objectives

It can be difficult to choose whether a condition is a constraint or an objective.

For example: should we try to minimize cost, or should we set a constraint stating that cost should not exceed a given level. The two approaches can lead to different designs. Sometimes, the initial formulation will need to be revised in order to fully understand the design space. In some formulations, all constraints are treated as objectives (physical programming**).**

2.4 Sample problems in MDO

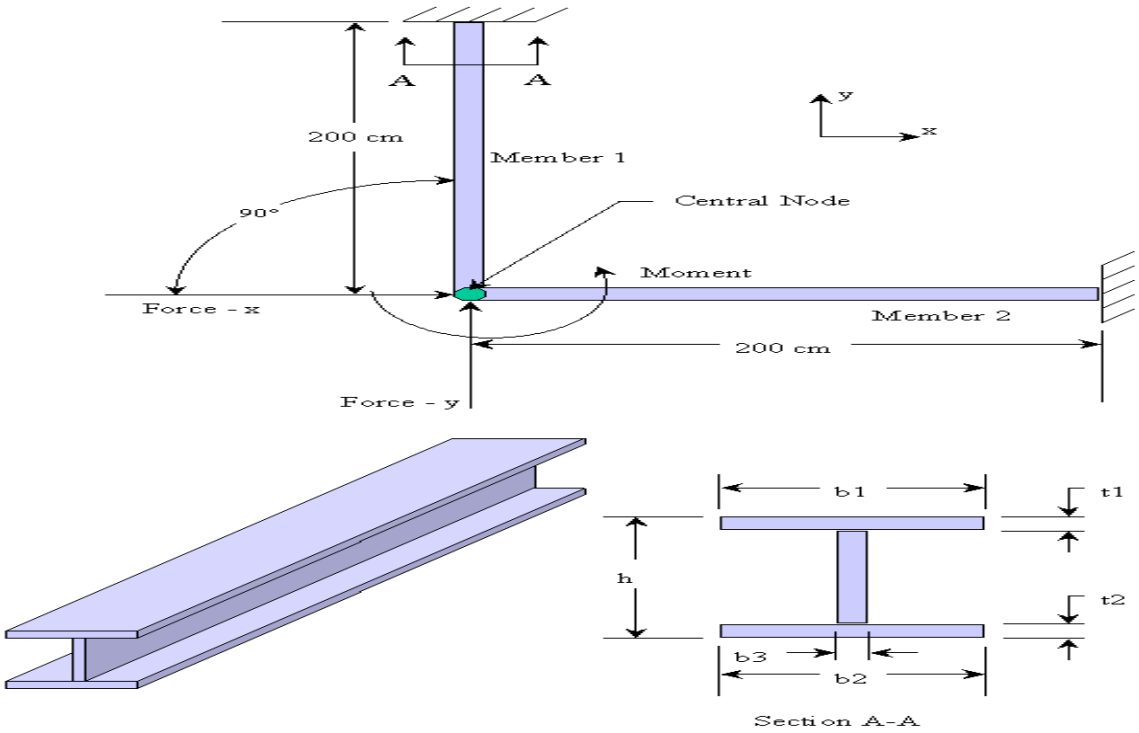
(1) Aerospace wing design

Aerospace wing design problems include design variables from multiple disciplines like controls, structures, aerodynamics etc., An example problem may look like minimizing the take-off weight of the aircraft by changing wing geometric parameters while satisfying the given range and payload requirements at the given cruise speed.

(2) Hub Frame Problem

It deals with the structural sizing of a hub frame. The members in the frame structure are I-beams. As can be seen by the figure below, the system has a central node where the members attach. Each member is defined by six design variables that give the cross-sectional dimensions for the I-beam. Different load cases can be applied at the central node of the two-dimensional frame and are each defined by a vertical force, a horizontal force, and a moment.

The objective for this problem is to find the lightest structure possible that is able to support all of the different loading cases. In minimizing the mass, or equivalently the volume, the stresses and buckling in each member must not be greater than the respective maximums defined for the material. Additionally, the central node of the structure cannot rotate or translate more than a specified amount.



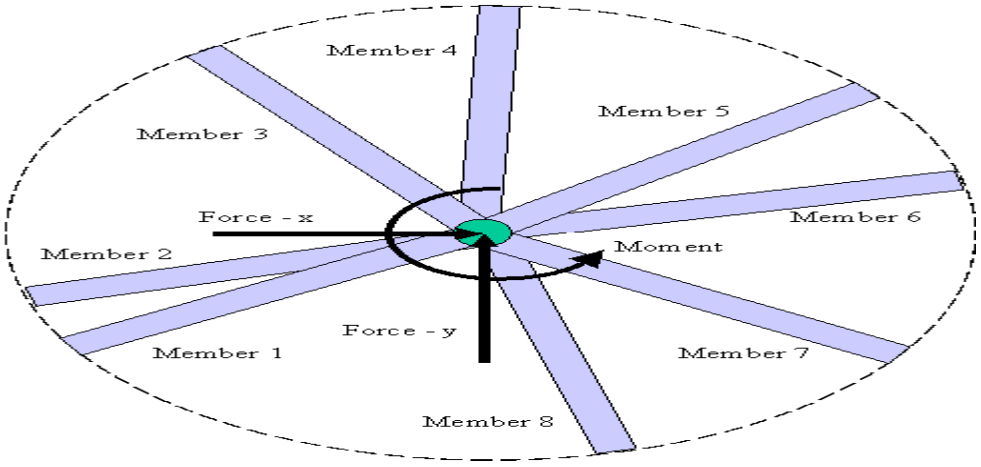


Fig: 2.3 Eight Member Hub Frame Assembly with a Single Load Case

2.5 MDO in design process

1. Define overall system requirements.
2. Define design vector **x**, objective **J** and constraints.
3. System decomposition into modules.
4. Modeling of physics via governing equations at the module level – module execution in isolation.
5. Model integration into an overall system simulation.
6. Benchmarking of model with respect to a known system from past experience, if available.
7. Design space exploration (DoE) to find sensitive and important design variables *xi*
8. Formal optimization to find min J(x).
9. Post-optimality analysis to explore sensitivity and tradeoffs: sensitivity analysis, approximation methods, isoperformance includes uncertainty.

In Practice...

1. Step through 1-8
2. The optimizer will use an error in the problem setup to determine a mathematically valid but physically unreasonable solution

   or

   The optimizer will be unable to find a feasible solution (satisfies all constraints).
3. Add, remove or modify constraints and/or design Variables.
4. Iterate until an appropriate model is obtained.

2.6 Challenges of MDO

- Fidelity/expense of disciplinary models
  Fidelity is often sacrificed to obtain models with short computation times.
- Complexity
  Design variables, constraints and model interfaces must be managed carefully.
- Communication
  The user interface is often very unfriendly and it can be difficult to change problem Parameters.

- Flexibility

  It is easy for an MDO tool to become very specialized and only valid for one particular problem.

2.7 Advantages and Disadvantages of MDO

Advantages

- reduction in design time
- systematic, logical design procedure
- handles wide variety of design variables & constraints
- not biased by intuition or experience

Disadvantages

- Computational time grows rapidly with number of design variables.
- Numerical problems increase with number of design variables.
- limited to range of applicability of analysis programs
- will take advantage of analysis errors to provide mathematical design improvements
- difficult to deal with discontinuous functions.

CHAPTER 3

Design of Experiments (DoE)

3.1 Introduction

Design of experiments (DoE) is a collection of statistical techniques providing a systematic way to sample the design space. It is useful when tackling a new problem for which we know very little about the design space. It studies the effects of multiple input variables on one or more output parameters and often used before setting up a formal optimization problem. Design variables are called as factors, values of design variables are called levels and outputs are the observations.

- Identify key drivers among potential design variables
- Identify appropriate design variable ranges
- Identify achievable objective function values

3.2 Techniques Useful in DoE

Full factorial Design, Orthogonal arrays, Latin Hypercubes, Response Surface Methodology (RSM) etc., are the techniques that will be used in Design of Experiments [12].

Matlab's model based calibration toolbox offers a full range of proven experimental designs, including [14]:

- Classical: Box-Behnken, Central-Composite, and Full Factorial.
- Space-filling: Latin Hypercube and lattice.

Response Surface Methodology

The above mentioned techniques are useful in identifying a few important variables from a large set of candidate variables, i.e., a screening experiment and ascertain how a few variables impact the response.

Where as, RSM is used to know what specific levels of the important variables produce an optimum response.

RSM procedure:

1. Plan and run a factorial (or fractional factorial) design near/at the starting point.
2. Fit a linear model (no interaction or quadratic terms) to the data.
3. Determine path of steepest ascent (PSA) - quick way to move to the optimum - gradient based.
4. Run tests on the PSA until response no longer improves.
5. If curvature of surface is large go to step 6, else go to step 1
6. Neighborhood of optimum - design, run, and fit (using least squares) a 2nd order model.
7. Based on 2nd order model - pick optimal settings of independent variables.

CHAPTER 4

PCA and SOM

4.1 Introduction to PCA

PCA is a statistical technique that is mainly used for the compression and classification of data. The purpose is to reduce the dimensionality of a data set (sample) by finding a new set of variables, smaller than the original set of variables, which retains most of the sample's information. Information means the variation present in the samples given by the correlation between the original variables.

PCA uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components and these PC's are ordered by the fraction of the total information each retains. This transformation is defined in such a way that the first principal component has as high a variance as possible (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to (uncorrelated with) the preceding components. It has applications in face recognition, image compression, finding patterns in high dimensional data, etc., PCs are a series of linear least squares fits to a sample, each orthogonal to all the previous ones [11].

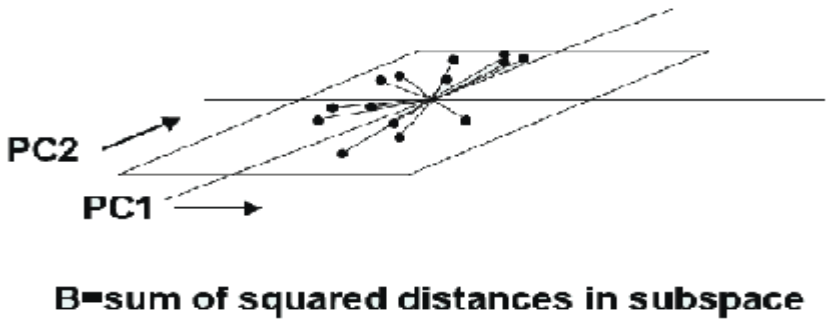


Fig 4.1 Figure showing PC components.

In the Figure 4.1 PC1 and PC2 represent the orthogonal axis to the data and give uncorrelated variables.

Calculation of Principal Components:

1. Take a dataset
2. Calculate the co-variance matrix of the dataset.
3. Calculate the Eigen vectors and Eigen values of that co-variance matrix.
4. Eigen Vectors with the highest Eigen values are selected as the Principal Components.
5. Generally, the first two PC's are necessary to represent the dataset.

In Matlab the in-built function *princomp* is used to generate principal component coefficients and scores.

4.2 Introduction to SOM

A self-organizing map (SOM) or self-organizing feature map (SOFM) is a type of artificial neural network which was developed by professor TEUVO KOHONEN is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map. SOM has been successfully applied in data mining and visualization. It is a model for clustering and visualizing high-dimensional data, which groups data with similar characteristics. The purpose of visualization is to project data onto a graphical representation to provide a qualitative idea of its properties [5].

Self-organizing maps are different from other artificial neural networks in the sense that they use a neighborhood function to preserve the topological properties of the input space. Usually, in SOM, the multi-dimensional data is mapping to the two dimension space with hexagonal grids. Thus, it makes the multi-dimensional design variables visualization come true in complex products design.

It belongs to the category of competitive learning networks. SOM cannot provide any geographical features, coordinates, distance and so on, but it can describe closeness or distribution of the input design variables. It contains an input layer and a output layer with no hidden layer.

A self-organizing map consists of components called nodes or neurons. Associated with each node is a weight vector of the same dimension as the input data vectors and a position in the map space. The usual arrangement of nodes is a regular spacing in a hexagonal or rectangular grid. The procedure for placing a vector from data space onto the map is to find the node with the closest weight vector to the vector taken from data space and to assign the map coordinates of this node to our vector.

4.2.1 Working of SOM

Diagrammatic sketch of SOM is depicted in the Fig 4.2. n-dimensional design variables and m-objective functions vectors are input to the input layer, where n and m are positive integer, at the same time, assigned to them to each (n + m) neuron. In the output layer, n + m dimensional weight vectors V = {v1, v2, v3, . . . , vn+m} are randomly assign to neurons. In SOM algorithm, the input vectors are fully connected with neurons in the output layer. The unsupervised learning in SOM is to cluster together similar patterns while preserving the topology of input space. During this learning process, two main goals are pursuing to be achieved. The first is, output layer searches for the winning unit, which is the neuron with a closer weight vector to each input vector in the input layer. The second is, in order to be much closer to the input design variables and objective function vectors, weight vectors of the winning unit and its neighboring neurons will be updated. As a result, the n + m dimensional input vectors are projected onto a sequence of neighboring neurons in the two-dimensional hexagonal grid.

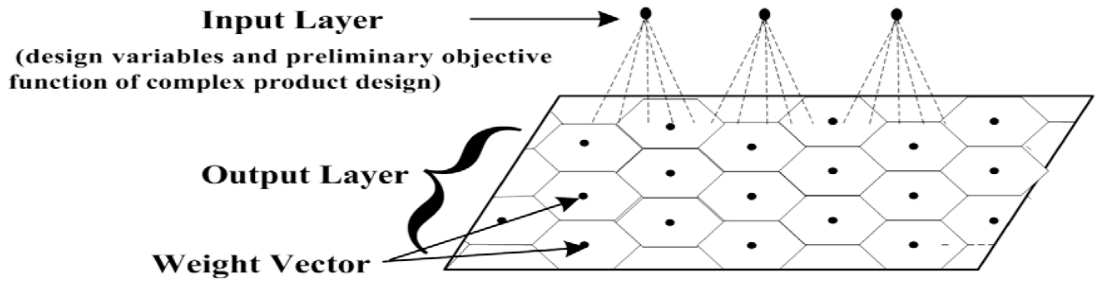


Fig: 4.2 Figure showing the Self-Organizing map.

4.2.2 SOM Algorithm

The SOM training algorithm consists of four steps. They are: Initialization, Competition, Cooperation, and Synaptic Adaptation.

1. Initialization

- Initialize the weights of the nodes and set the learning rate and topological neighborhood parameters.
- Select a vector x from the input pattern distribution for input to the network.

2. Competition

- Determine the node K with weight vector closest to the input vector X, by calculating the difference in weights of the input vector and nodes.

$$\text{Winner neuron} = \text{argmin}_i \left(\| \vec{X} - \vec{W_j} \| \right)$$

3. Cooperation

- The winning node locates the center of a topological neighborhood of Co-operating nodes.

$$h_{j,i} = \exp\left(-d^2_{j,i} / 2\sigma^2 \right) \text{ and}$$

The topological neighborhood shrinks with the time.

$$\sigma(t) = \exp\left(-t / \tau_1 \right) \qquad h_{j,i}(t) = \exp\left(- d^2_{j,i} / 2\sigma^2(t) \right)$$

4. Synaptic Adaptation

- Update the weight vectors in the (t+1)st iteration according to the below equation.

$$\mathbf{\vec{W_j}(t+1) = \vec{W_j}(t) + \eta(t)}\, h_{j,i}(t)\, (\vec{x} - \mathbf{\vec{w}_j(t)})$$

$$\mathbf{\eta(t) = \eta_0 \exp}\left(-t / \tau_2 \right)$$

where **η** implies the learning rate at time t and $h_{j,i}(t)$ implies the neighborhood kernel around the winner unit j at time t.

- Reduce the neighborhood and learning parameters and repeat steps 2 through 5 until the network weights stabilize.

4.2.3 Difference between SOM and K-means clustering

K-means first finds the cluster centroids and then takes a sample form the dataset, calculates the difference between weight vectors of the sample to all the cluster centroids and assigns the sample to that particular centroid where the difference value is minimum. This procedure continues until the learning rate becomes zero, which is initialized in the beginning to a random value between [0, 1].

In SOM, when the input neuron finds the matching unit on the map, not only matching unit's weight but also the weights of it's neighborhood units is also updated. This makes the map preserve the topological structure of the input space.

The neighborhood function *h* is usually a function that decreases with the distance (in the output space) to the winning unit, and is responsible for the interactions between different units. During training, the radius of this function will usually decrease, so that each unit will become more isolated from the effects of its neighbors. is important to note that many implementations of SOM decrease this radius to 1, meaning that even in the final stages of training each unit will have an effect on its nearest neighbors, while other implementations allow this parameter to decrease to zero.

CHAPTER 5

Visualizations using PCA and SOM

5.1 Visualizations by PCA

Performing PCA on the dataset gives the principal component scores. These scores give the representation of the dataset in the principal component space. Now by performing a clustering operation such as the k-means on these scores, we will get cluster centroid locations and cluster indices of each point.

5.1.1 Cluster diagram

By taking principal component scores of the data and applying k-means clustering on the data, we get a figure which shows the clusters present in the dataset. With this we can know how the data is distributed into different classes.

5.1.2 Voronoi tessellation

By finding out cluster centroids in the principal component scores of the given data, lines joining one cluster centroid to other cluster centroid are drawn. Now, the perpendicular bisectors of these lines give the Voronoi tessellation diagram, which represent discrete, set of objects in the space.

One approach to find the required design space is to map the interesting object on to this figure and select the components or areas into which this falls [18].

5.2 Visualizations by SOM

The Matlab comes with a default toolbox designed for neural networks. On typing the command *nntool* it opens a GUI of neural network toolbox. The command *nctool* opens a GUI of neural network clustering tool, with which we can solve a clustering problem with the help of a self-organizing map (SOM) network. The nctool GUI is useful in showing how the data is spread on the map, weight positions etc., but advanced visualizations are not possible with it.

Another toolbox designed by Juha Vesanto and team from Laboratory of Computer and Information Science, Helsinki University of Technology, Finland is very useful in SOM based data visualization methods [3].

SOM-Based Data Visualization Methods

These methods help in achieving the aim of visualization in understanding the mapping area and enabling the investigation of new data samples with respect to it.

It describes three main visualization methods. They are:

1. Structure and Shape.
2. Properties of the prototype vector set.
3. Examining new data with the map.

5.2.1 Structure and Shape

It gives an idea of the overall shape and possible cluster structure of the dataset. Using the projection methods it tells what and where are the clusters and outliers? and what is the shape of the data cloud?

Sammon's projection and Curvillinear component analysis (CCA) projection give a rough idea of clusters in the data, where as U-matrix technique shows the clusters on SOM.

U-matrix (unified distance matrix)

U-matrix holds distances between neighboring map units, as well as the median distance from each map unit to its neighbors and, are visualized using gray shade.

Distance matrix methods show borders between clusters and visualization can be done through the similarity of the map units, which can be done by giving a color to each map unit so that similar map units get similar hues. The advantage of the distance methods over projections such as Sammon's is that they do not use position to indicate clusters. Thus the position can be used for linking the figures to other visualizations, and some other visual dimension can be used to show the cluster information.

White dots indicate locations of map units, hexagons between them show actual values of the U-matrix. The darker the shade, the bigger the distance between neighboring map units. The light areas with borders are the *clusters.*

Calculation of U-matrix

U-matrix representation requires huge number of calculations. After the SOM map is trained the, difference in distance between the adjacent cells are calculated for whole map and are stored. Now median of the obtained distances are calculated for a cell by taking the distances from all its adjacent cells and that is used in representing U-matrix.

As the size of the input data increases the map size increases and U-matrix calculations becomes cumbersome.

The SOM Matlab toolbox function *som_umat* calculates u-matrix values and generates the u-matrix figure.

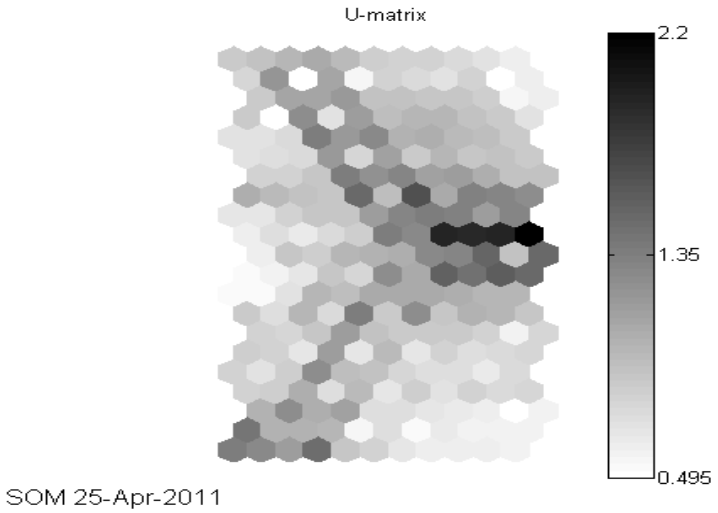


Fig: 5.1 U-matrix representation showing the spread of data.

The above figure shows the spread of the given input data into three cluster regions, which are separated by the dark border.

- the lower right region
- the upper right region
- the left region.

5.2.2 Properties of the prototype vector set

The U-matrix visualizations give an idea of the overall cluster characteristics of the data. The component planes play a key role in visualizing the properties.

Each component plane consists of the values of a single vector component in all map units. Simple inspection of the component plane provides an idea of the spread of values of that component. Component planes can also be used for correlation hunting. Correlations between component planes are revealed as similar patterns in identical positions of the component plane.

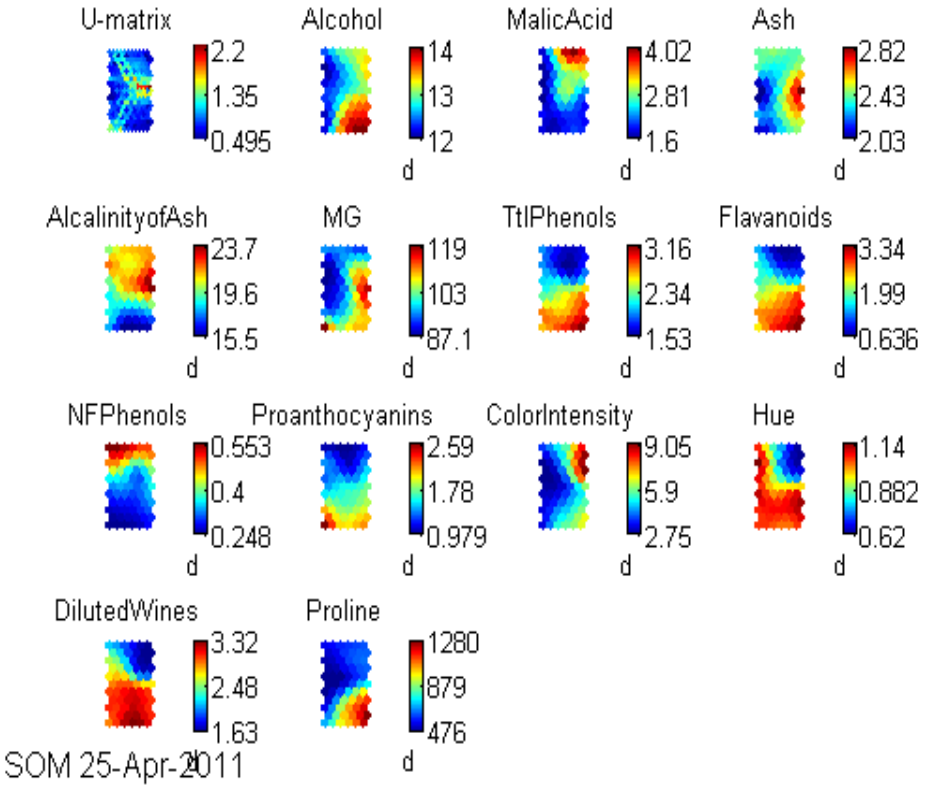


Fig: 5.2 Figure showing U-matrix and component planes.

From the component planes map by, choosing the interesting regions in the objective function, we can know the range of the values of the components in that particular interested region and thus, reduces the design search space.

The Fig 5.2 is generated by using the *som_show* function in the som Matlab toolbox.

som_show

It shows basic SOM visualizations such as component planes, u-matrix etc.,
The syntax of the function is
h = som_show(sMap, ['argID', value, ...])
the square braces indicate optional arguments.

5.2.3 Examining new data with the map.

- It gives the part of the mapped distribution which best corresponds to the given data.

The response between data sample and each prototype vector are calculated and the nearest prototype vector is the Best Matching Unit (BMU) of that sample. The response is simply shown by showing the BMU on the map. When the BMUs of multiple data samples are aggregated, a hit histogram results. It gives the response of the given data on the map.

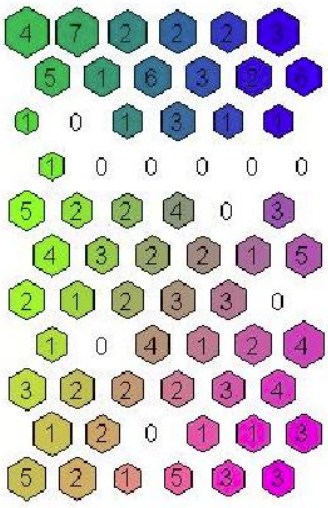


Fig: 5.3 Hit Histogram showing the number of hits by data in each cell.

- These tell how accurate the correspondence/localization is.

The accuracy can be measured as the average quantization error.

Quantization error:

It is the average distance between each data vector and its BMU.

Topographic error:

The proportion of all data vectors for which first and second BMUs are not adjacent units.

- Can compare how similar are two data vectors in terms of the map.

By projecting the new data sample on to the map and comparing it with the nodes on the map, we can visualize in which area the new test sample falls.

5.3 Graphical User Interface (GUI) for selection of cells in SOM

The SOM Matlab toolbox [6] contains a function with name *som_select* , which is used for the manual selection of map units from a visualization. The code for this function is written with some errors; we rectified the errors and made the function to work. For this function to work, we should generate the figure either by using *som_show* or *som_show_add* functions as the handle of the generated figure is to be passed onto the *som_select* function.

som_select, generates a GUI which enables in selection/removal of the cells from our region of interest. It displays the string present in the map labels field. With the help of this, we can concentrate only on our interested regions/cells, eliminating unwanted regions/cells.

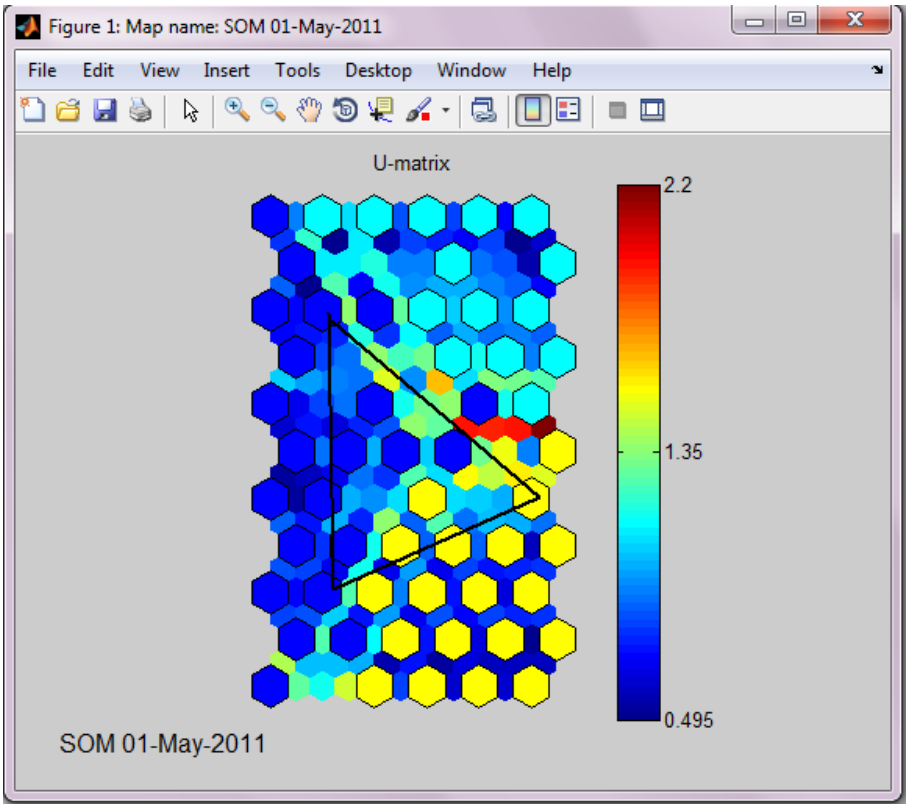


Fig 5.4 Figure showing GUI for SOM map showing selection of a region on the map.

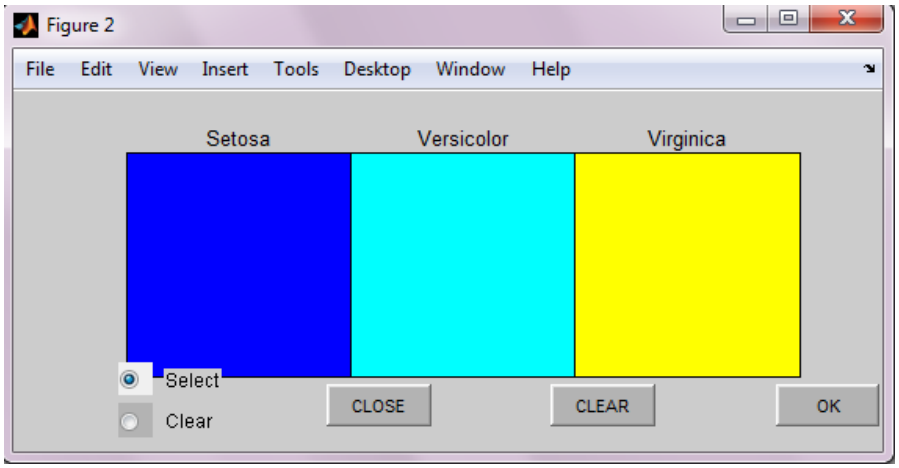


Fig 5.5 Figure showing selection of classes on the GUI.

The left mouse button allows us to draw a polygon, right mouse button selects/clears the cells in that region and middle mouse button selects/clears the cells individually. If we select a colour on the GUI in figure 2 and select cells on GUI in figure 2, those cells will be turned into the selected colour.

In this way, if we know from the past design knowledge the interested region/cells, we can select those cells with the help of this GUI and can concentrate on the search space only on those region/cells, which decreases lot of design search space.

5.4 Limitations of SOM

- Missing Data

Need a value for each dimension of each member of samples in order to generate a map. In the toolbox, if data is missing then, it is represented by the value Nan (Not-a-number) in the data matrix. Missing components are handled by excluding them from the distance calculations.

- Computationally very Expensive

As the dimensions of the data increases, the map size increases and the more number of calculations are involved in finding distance matrix (U-matrix)and the more neighbors we used to calculate the better similarity map we will get, but the number of distances the algorithm needs to compute increases exponentially.

- No Objective Function

The formulation of the algorithm has no objective function that could be optimized.

↳ No comparison between classes

A distance between two points in the map space signifies either similarity or difference between the corresponding two vectors in the data space, but it will not say to what degree two class distributions are similar or different.

SOM of SOMs called SOM^2 [9]

As SOM is a map of data vectors but not a map of class distributions, it does not deal with the degree of comparison between classes. Whereas SOM^2 is an extension of SOMs called "SOM of SOMs," in which the mapped objects are self-organizing maps themselves. SOM^2 provides a method for the topological mapping of classes by comparing their distributions instead of comparing individual vectors. For class visualization and analysis this method will be very much useful.

↳ It is possible to fail finding of the design knowledge due to a large number of objective functions and design variables. For example if one objective function includes design space of one variable and the same is opposed by another objective resulting in a conflict leading in failure of the design knowledge.

CHAPTER 6

Results

We have taken some of the sample data sets from the UCI Machine Learning Repository [16] and objective functions from the Genetic and Evolutionary Algorithm Toolbox for Matlab [17]. We applied the visualization techniques described in chapter 5 on these datasets and observed some correlations among the data. The observations are listed along with the visualization figures on each dataset.

6.1 Visualizations by PCA

6.1.1 Classification datasets

Test No 1

Name of the Dataset	Number of Components	Decision Variable
Iris [16]	5	Species

Iris dataset contains 4 components, sepal length, sepal width, petal length and petal width. Fifth component is the species component, which describes about the species of the sample.

1. Scatter Plot

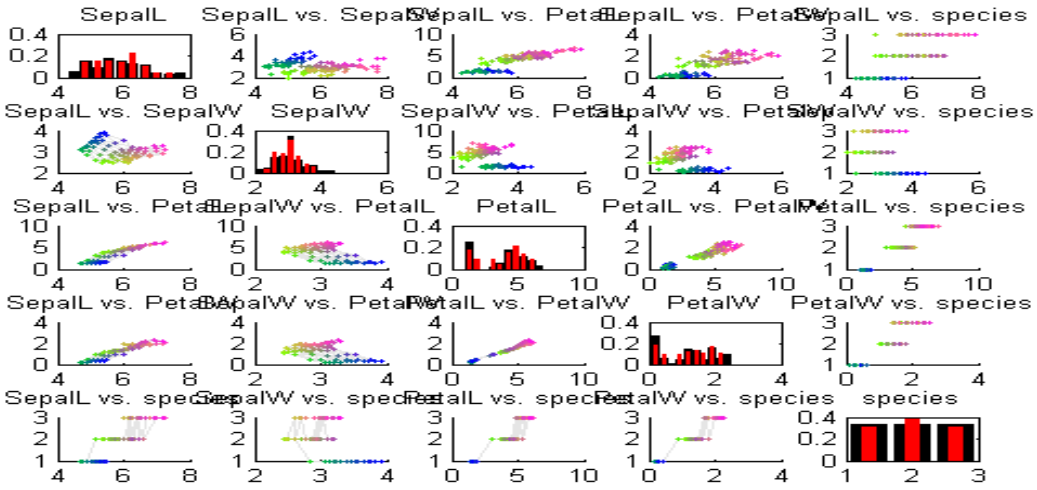


Fig: 6.1 Figure showing the plots between attributes of the data.

- The subplots (4,3) and (3,4) show some linear correlation between the components Petal length and Petal width.
- From subplots (1,3) and (1,4) it is clear that Sepal length, Petal length and Petal width are related.

2. From Cluster Map

- By taking principal component scores and applying k-means clustering on the data gives the following figure which shows the clusters in the dataset. Here data is classified into 3 clusters.
- One cluster which is on the top is completely separated by the clusters lying below in the map. By this way we can visualize the information of the dataset in the form of clusters.

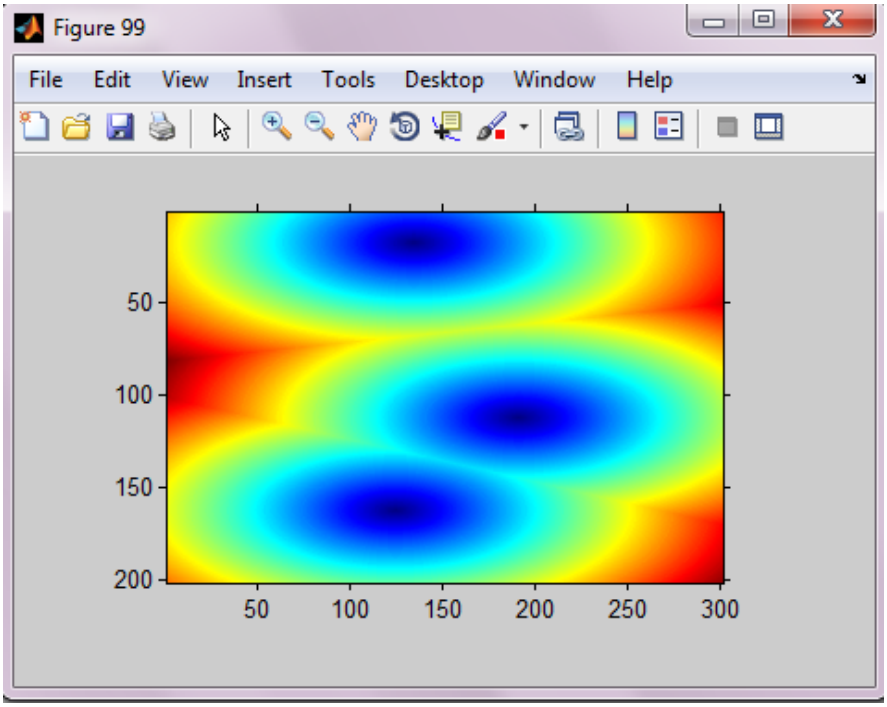


Fig: 6.2 Figure showing clusters in the data.

3. Voronoi Tessellation Figure

- By finding the cluster centroid locations in the principal component scores, straight lines are drawn between the cluster centroids. The perpendicular bisectors of these straight lines give discrete set of objects in the space. The below figure takes three cluster centroids and draws the lines to from centroid of one cluster to other.

- One approach to find the required design space is to map the interesting object on to this figure and select the components or areas into which this falls.

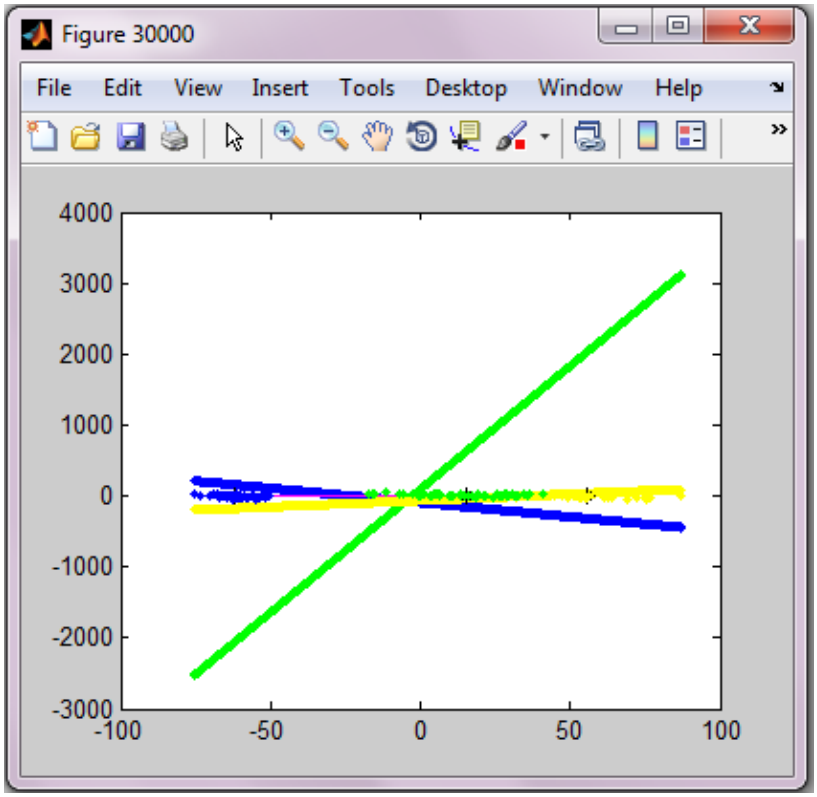


Fig: 6.3 Figure showing Voronoi Tessellations.

Test No 2

Name of the Dataset	Number of Components	Decision Variable
Wine [16]	13	Class

Wine dataset consists of 13 components and the data is classified into 3 classes.

1. Scatter plot

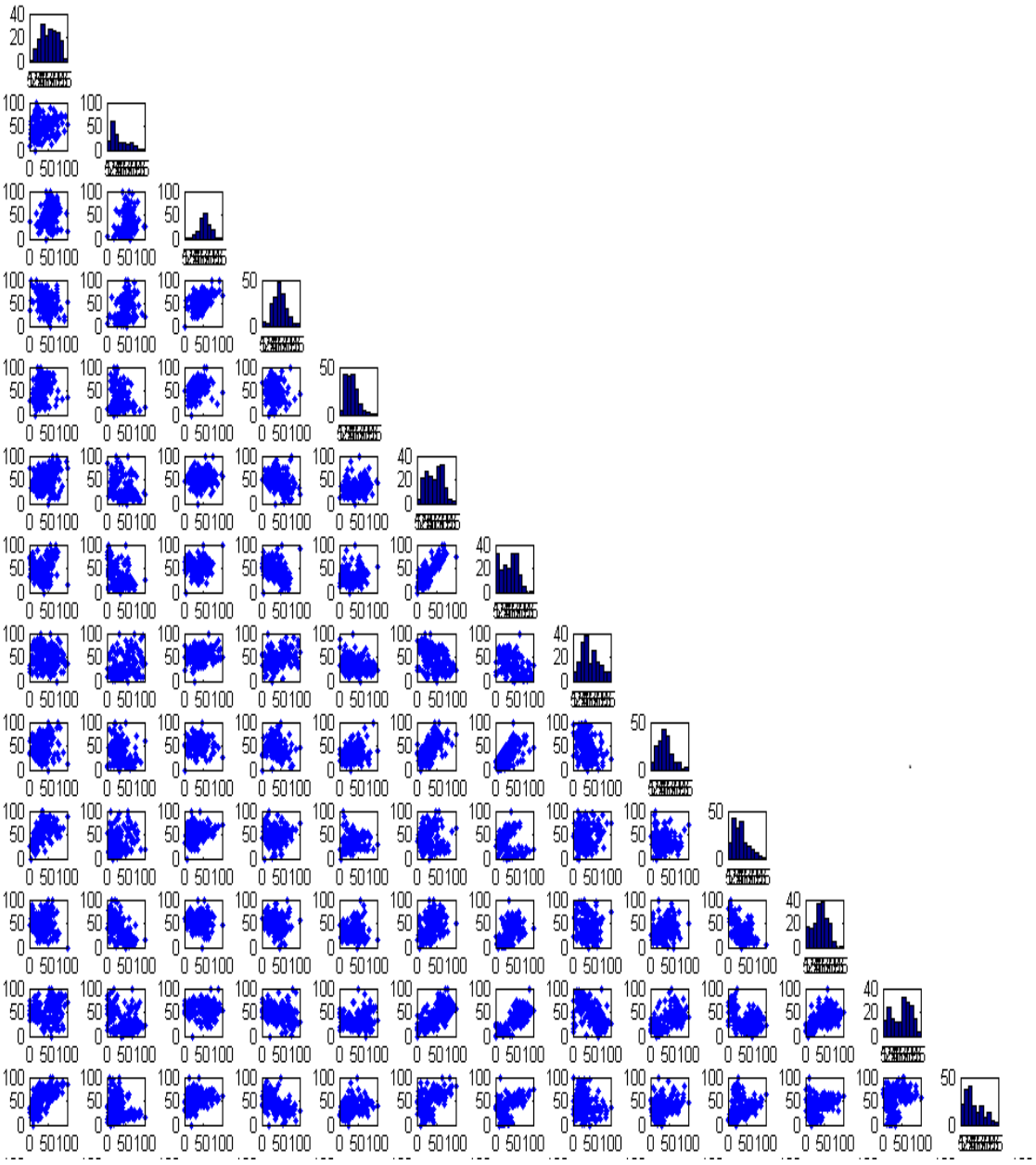


Fig: 6.4 Figure showing the plots between attributes of the data.

On x-axis the components are from 1-13 and on y-axis from 13-1.

As the dimensions increase, it becomes difficult to visualize the correlations between the variables. Comparisons of plots between individual components show correlations between them. The subplots (7, 6), (7, 9) and (7, 12) shows positive correlation.

2. <u>From cluster map</u>

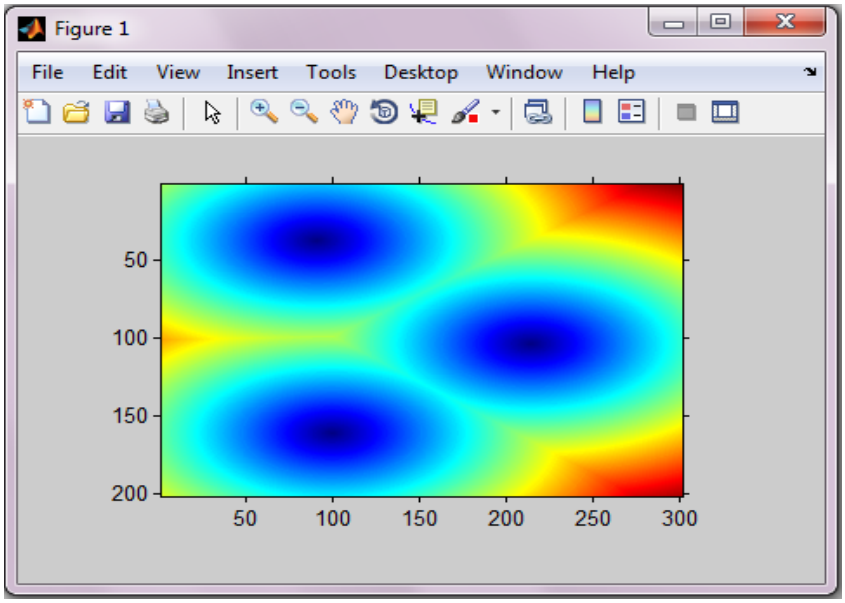


Fig: 6.5 Figure showing clusters in the data.

By applying K-means on principal component scores, we get the above map. The above map shows three clusters. The wine dataset is classified into three clusters.

3. From Voronoi Tessellation Diagram

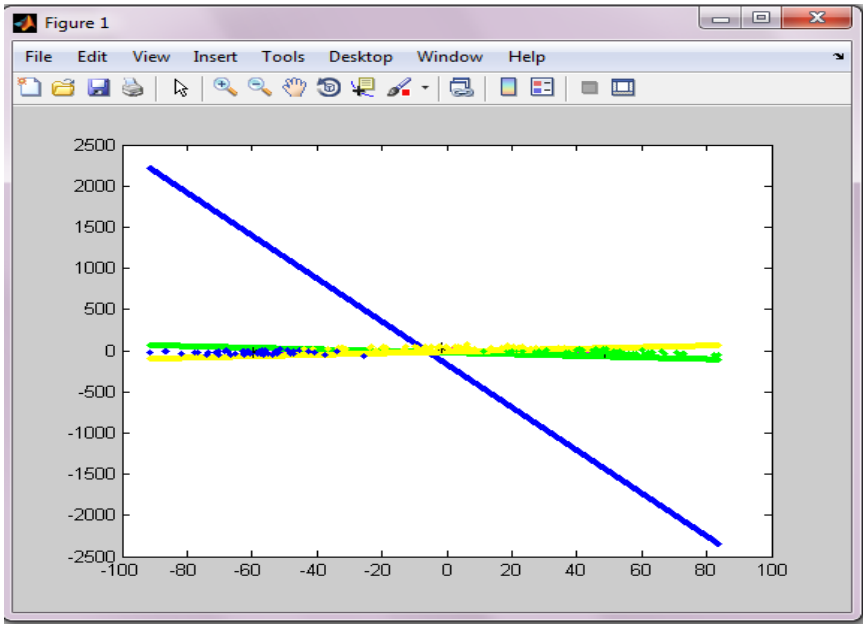


Fig: 6.6 Figure showing Voronoi Tessellations.

The small dots represent the cluster centroids in the three classes and the lines represent the perpendicular bisectors. This is useful in generating the discrete set Voronoi tessellation figure.

6.1.2 Multi-dimensional Optimization Functions

Test No 3

Name of the Dataset	Number of Components	Decision Variable
De jong's function [17]	10	Objective

We have taken the De jong's function and generated ten dimensional samples and tested these samples with PCA and SOM visualization techniques.

1. Scatter Plot

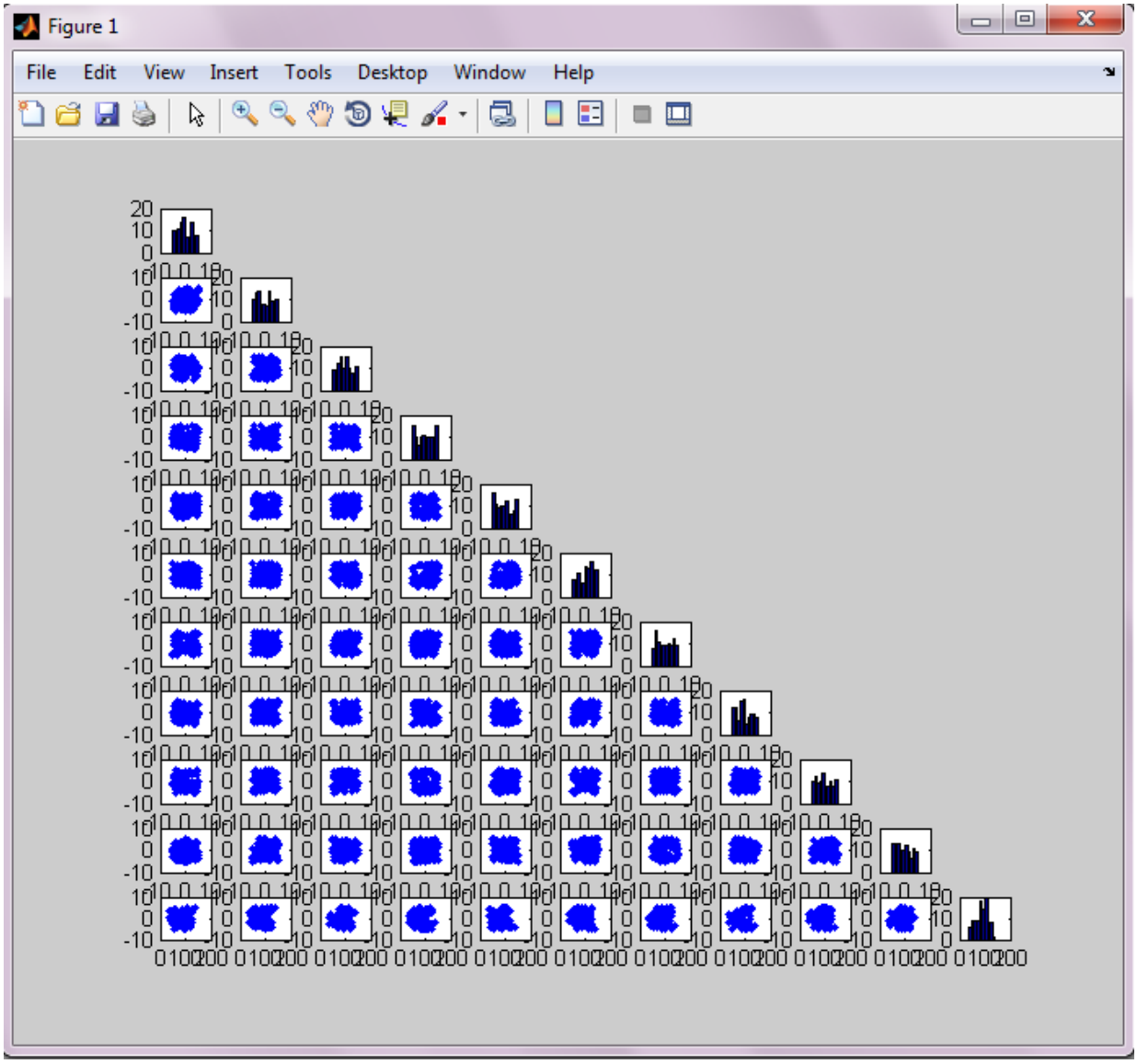


Fig 6.7 Figure showing the plots between attributes of the data.

On x-axis the components are from 1-11 and on y-axis from 11-1.

As the dimensions increase, it becomes difficult to visualize the correlations between the variables. Comparisons of subplots between individual components show correlations between them. The subplots (1,7), (4,8) and (6,9) show similar maps.

2. From cluster map

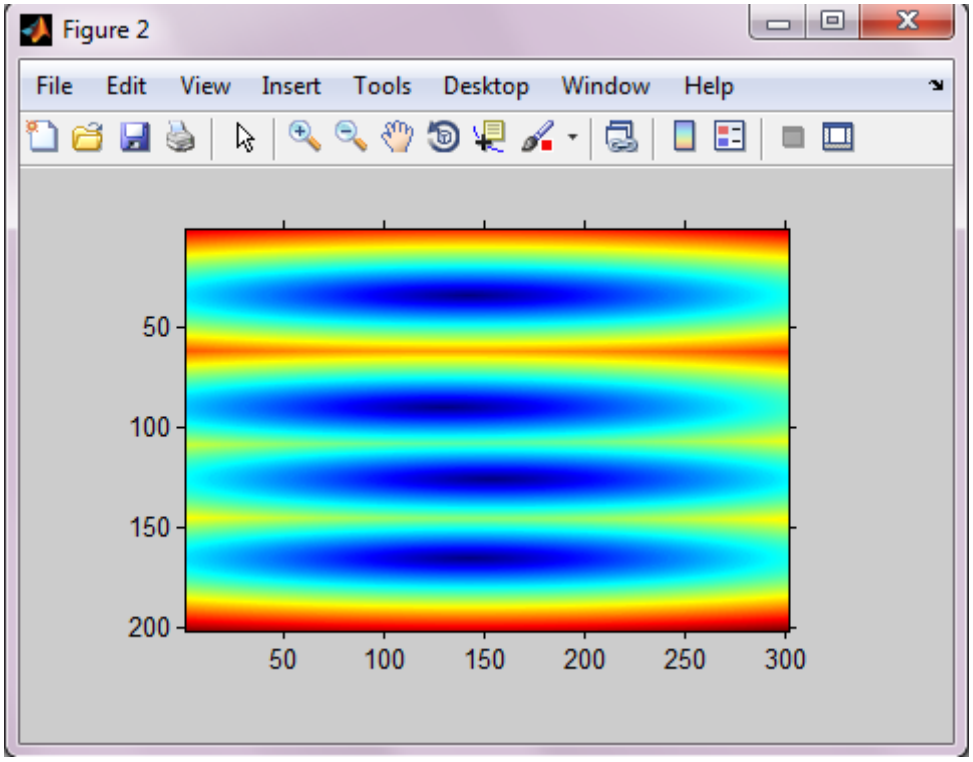


Fig 6.8 Figure showing clusters in the data.

By applying the K-means clustering technique on PC scores, we can visualize the clusters. The top cluster is well distinguished from the below clusters and we can observe this in the SOM U-matrix representation.

6.2. Visualizations by SOM

6.2.1 Classification datasets

Test No 1

Name of the Dataset	Number of Components	Decision Variable
Iris [16]	5	Species

The first map shows the U-matrix and next 4 maps show the different components in the data and the last one shows the decision variable component.

1. From Component planes Map

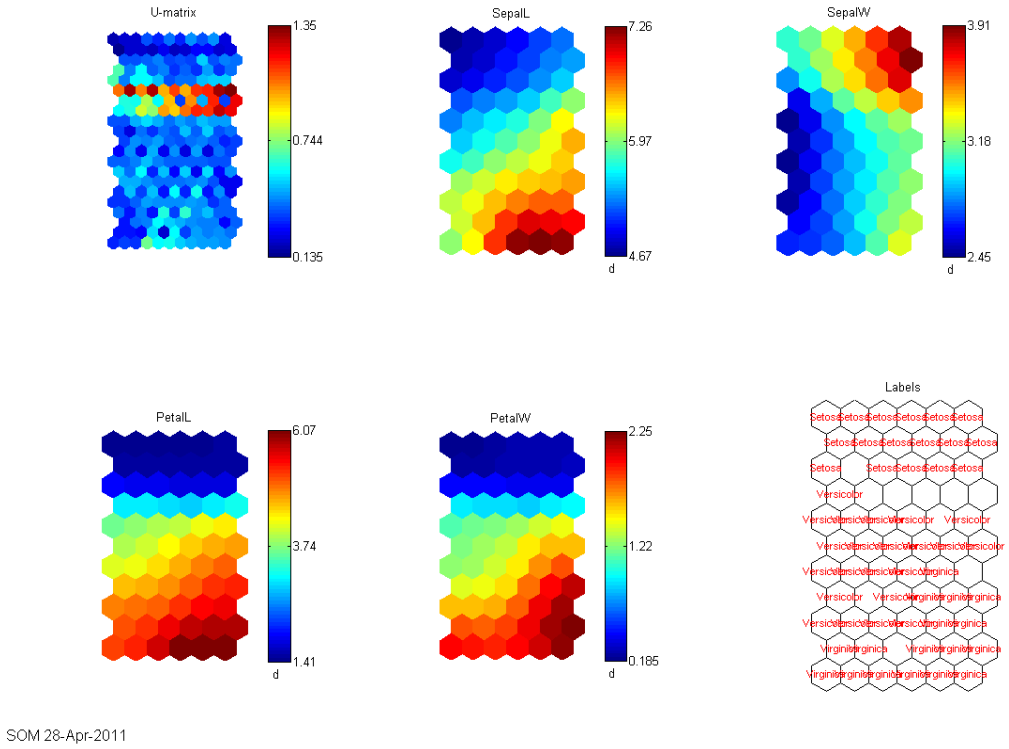


Fig: 6.9 Figure showing U-matrix, component planes and Labels of the data.

- PetalL (petal length) and petalW (petal width) have high linear correlation and show similar characteristics with respect to the class variables.
- SepalL is correlated (at least in the bigger cluster) with PetalL and PetalW.
- SepalL and SepalW have a clear linear correlation, but it is slightly different for the two main clusters.

From U-matrix and labels map

It clearly shows and gives labels to clusters. Top side of the map is one cluster (setosa) and bottom area is combination of two clusters (versicolor and virginica).

2. From bar plane map

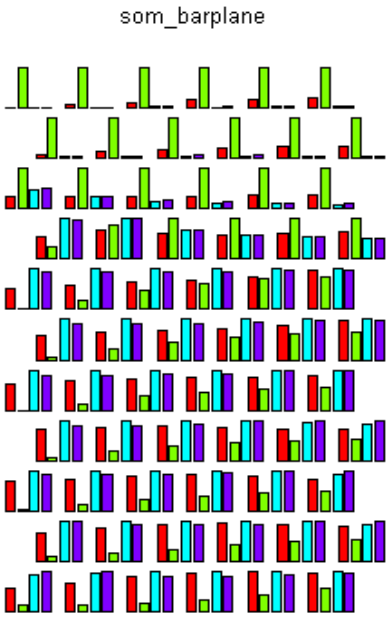


Fig: 6.10 Figure showing all components of the data in the bar plane.

The second component, SepalL (sepal width) is playing main role in the top cluster (setosa) of the map and in the left cluster, components 1,3 and 4 are playing major role. Whereas in, the right most cluster all components are distributed.

3. Errors Obtained

- Quantization error value obtained: 0.393
- Topographic error obtained: 0.013

4. Observations between PCA and SOM

PCA gives basic idea about spread of the data by showing clusters It shows correlations between components in the scatter plot.

SOM gives much more visualization than PCA. It not only shows the clusters, it also give labels to these clusters and shows the component that is playing major role in deciding the cluster. The component planes give the correlations between the components. It gives the value bar for each figure so that we can know the range of the component.

For this dataset,

- PCA shows that Petal length, Petal width and Sepal length are correlated. It also shows that he data is classified into three clusters.
- SOM also show that Petal length, Petal width and Sepal length component maps are similar. It also says that Sepal width component is the factor in distinguishing the setosa cluster from other clusters.

Test No 2

Name of the Dataset	Number of Components	Decision Variable
Wine [16]	13	Class

1. F**rom Component planes Map**

The first map shows the U-matrix and next 13 shows the different components in the data and the last one shows the decision variable component.

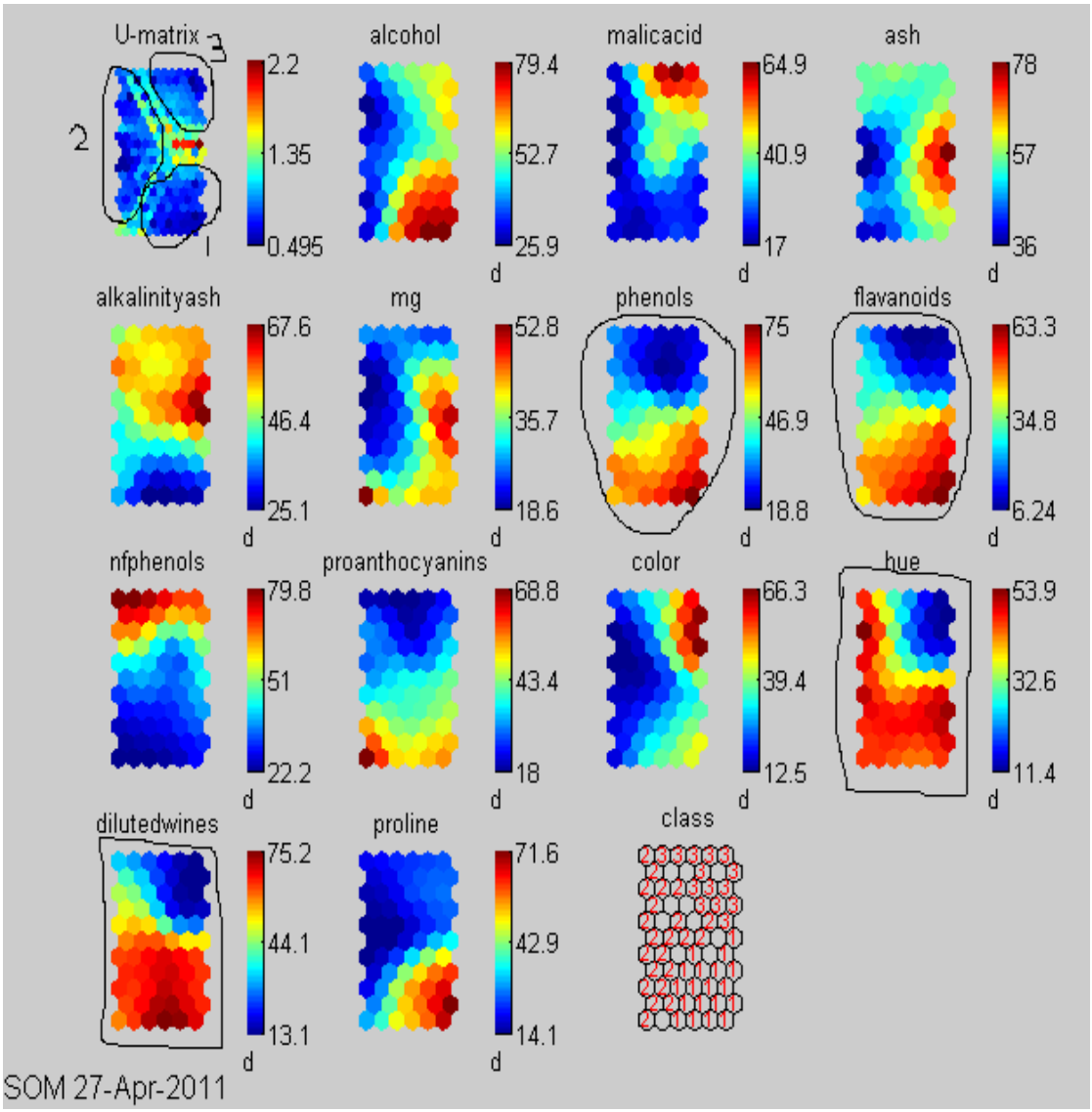


Fig: 6.11 Figure showing U-matrix, component planes and class Labels of the data.

- Flavanoids, phenols show similar response to the change in objective function.
- Diluted wines, hue show similar response to the change in objective function. (So, dimensionality reduction can be done.)

From U-matrix and labels map

- U-matrix representation shows the classified data and labels map gives the following information.

- Left side of the map is class two
- Upper right is class three and
- Lower right is class one.

- The value in the component

(1) Alcohol

is high in class1 ranging from 61.6 to 79.4

is low in class2 ranging from 25.9 to 43.7

is average in class3 ranging from 57 to 43

(2) colour

is 12.5 to 27 in class 2

is 48.4 to 30.5 in class 1

is 66.3 to 30.5 in class 3

similarly we can visualize the ranges of other components.

2. From Bar Plane map

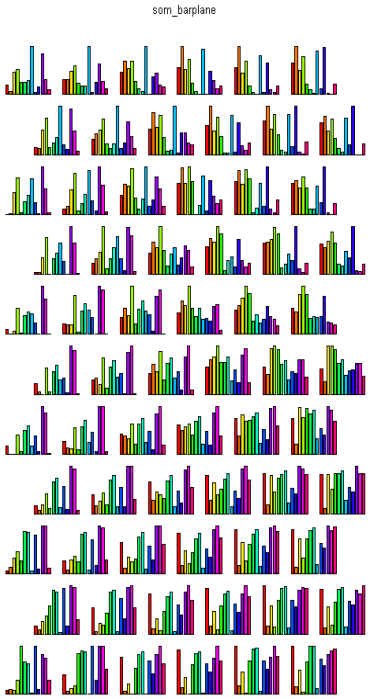


Fig: 6.12 Figure showing all components of the data in the form of bar plane.

- In class2 cluster (left side of the map) the components which are dominating are 6,7,8,11,12.
- In class1 cluster (right down) the components which are dominating are 1,6,7,11,12,13
- In class3 cluster (right top) the components which are dominating are 1,2,8,10.

  If we know what type of class we want to design then we can concentrate only on the regions on which it depends upon mostly.

3. Errors obtained

1. Quantization error value obtained: 1.883

2. Topographic error obtained: 0.017

4. Observations between PCA and SOM

PCA and SOM give the same basic visualizations, they say components 7, 8, 12 are correlated and classifies data into 3 clusters.

SOM shows that the components alcohol and color are the deciding factors in classifying. It also shows the range of these components in various clusters.

Test No 6

Name of the Dataset	Number of Components	Decision Variable
Breast Cancer [16]	9	Class (good or malign)

The first map shows the U-matrix and next 9 maps show the different components in the data and the last one shows the decision variable component.

1. From Component Planes Map

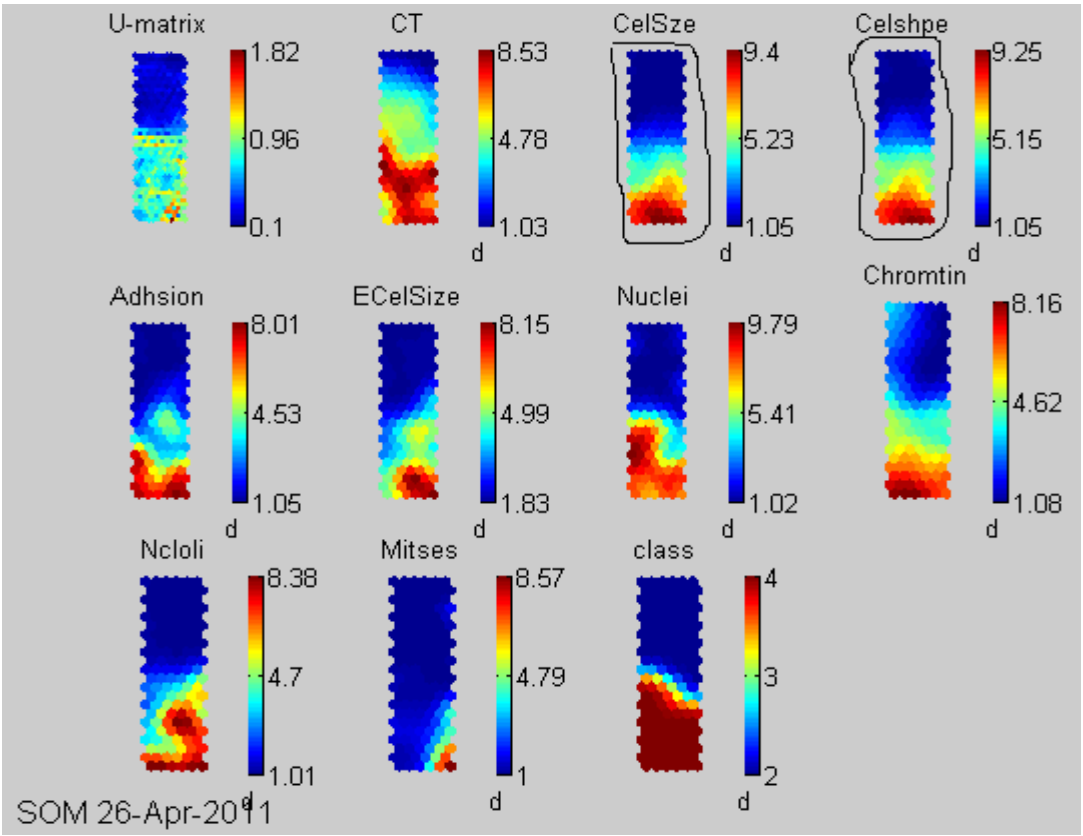


Fig:6.13 Figure showing U-matrix, component planes and class Labels of the data.

- Component3 (cell size) and Component4 (cell shape) are similar.

- The class component shows that the upper region is good and lower is malign It is clear from all the components that in malign class, sizes of all the components are increasing.

2. From bar planes map

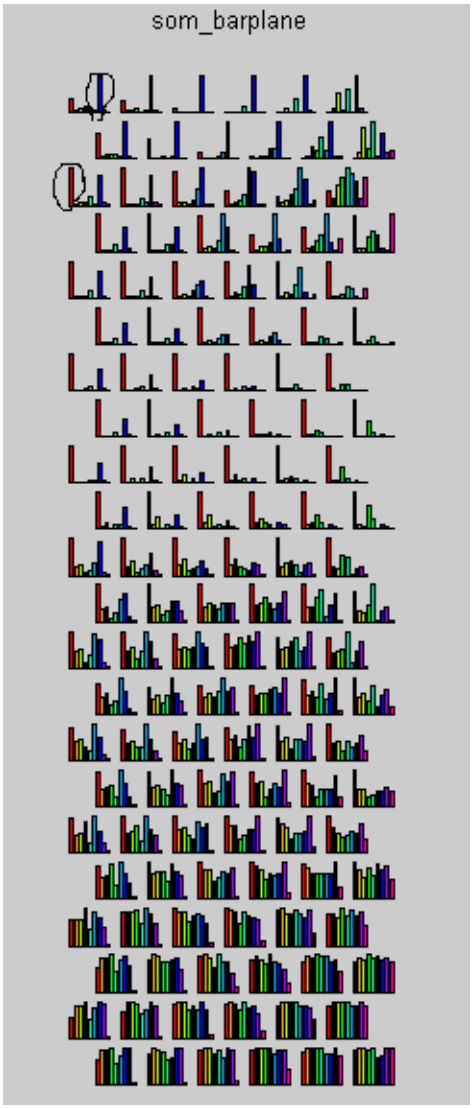


Fig: 6.14 Figure showing all components of the data in the form of bar plane.

From the above figure it is clear that in the good class, component 1 (CT) and component 7 (Chromatin) are playing major role in deciding the class. In the malign class all the components are very high indicating that larger values of the components make the sample fall in this region.

3. Errors obtained

- Quantization error value : 0.946
- Topographic error obtained: 0.029

6.2.2 Multi-dimensional Optimization Functions

Test No: 7

Name of the Dataset	Number of Components	Decision Variable
De jong's function 1 [17]	10	Objective

The first map shows the U-matrix and next 10 maps show the different components in the data and the last one shows the decision variable component, the objective function.

1) **From Component Planes Map**

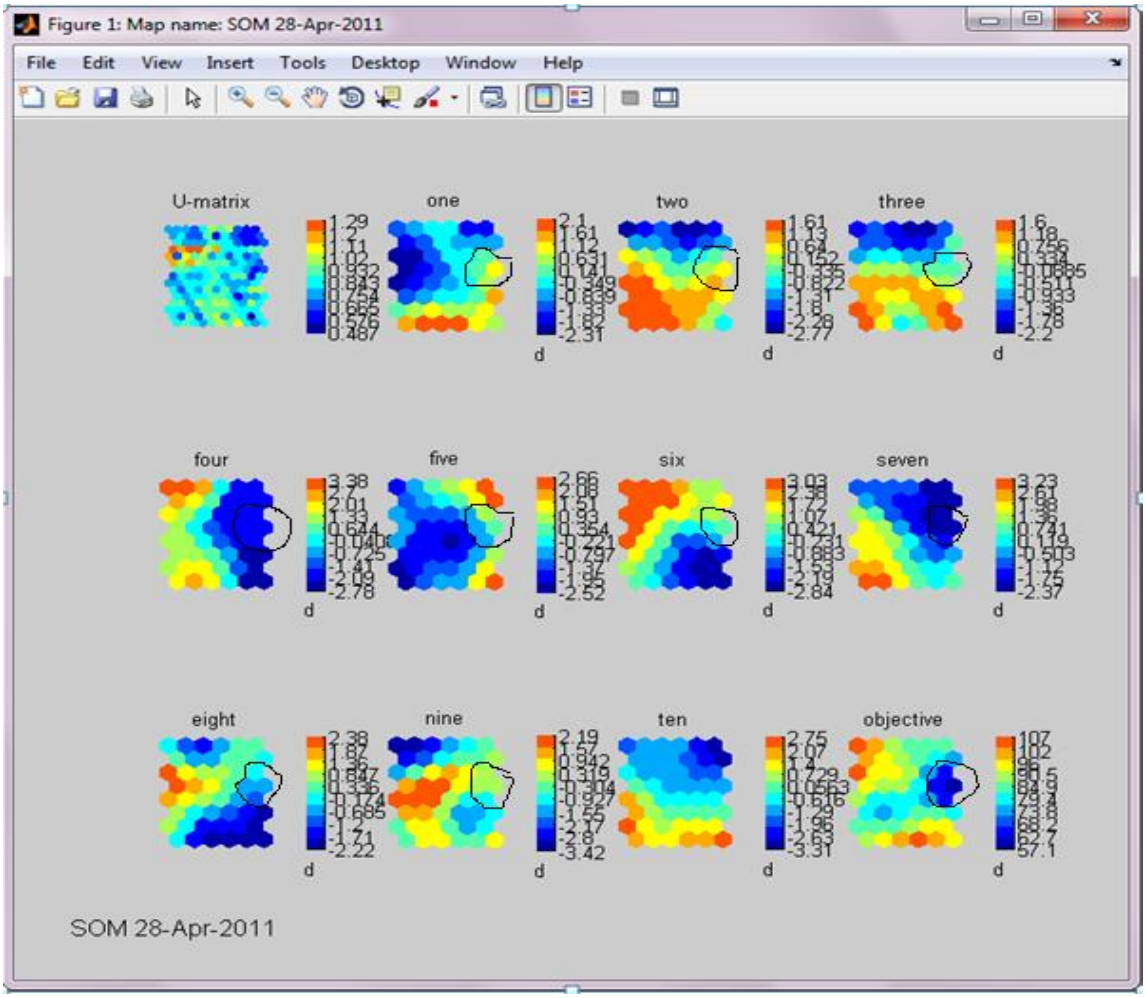


Fig: 6.15 Figure showing U-matrix, component planes and objective function of the data.

we have to minimize the last component, the objective function. . Let our interesting region lies between 57.1 to 68.2, which is indicated by a mark on the objective function map. Then we select the same region from all the components where the objective function is minimum in the objective component of the map.

- The U-matrix gives an idea of the spread of the data. The dark blue cluster on the top right says that more data is located there and the rest is distributed throughout the map. The red colour on the top says less data is present there.
- The components two and three, show some similar responses.

so, on observing the maps we get the following ranges of the components.

Component number	Search space	% Estimated Reduction in search space.
One	0.141 to 1.12	80
Two	0.152 to 1.13	80
Three	-0.511 to -0.885	90
Four	-2.78 to -2.09	90
Five	0.93 to -0.221	80
Six	1.07 to -0.883	60
Seven	-1.12 to -2.37	80
Eight	-0.174 to -1.2	80
Nine	-0.319 to -0.927	80
Ten	0.616 to 1.96	80

6.1: showing results obtained, estimated reduction in search space of individual components.

So, approximately there is **75** to **80** percent reduction in the design search space. This reduction helps the designer in saving much time and enables him concentrate only on the obtained design space.

2) <u>Hits figure</u>

- It shows the projection of the data set. A principle component projection is made for the data, and applied to the map. The color map is done by spreading a color map on the projection.

- Distance matrix information is extracted from the U-matrix, and it is modified by the knowledge of zero-hits (interpolative) units. The above figure shows the color code, with clustering information and the number of hits in each unit.

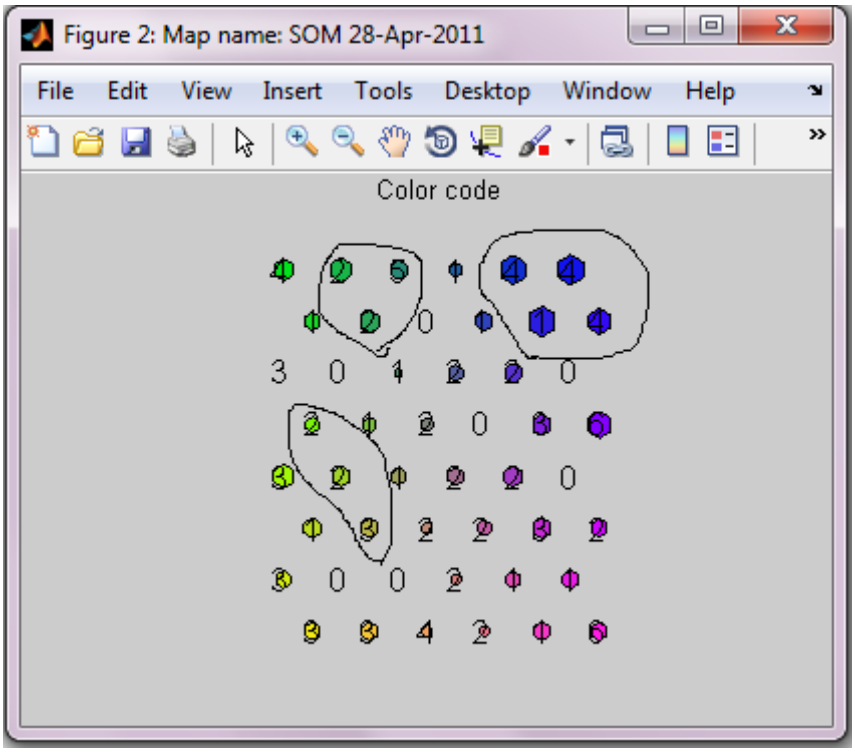


Fig: 6.16 Figure showing the number of hits by data in each cell.

- The data is distributed and we cannot see any clear clusters on the U-matrix. The regions marked shows that data is very close there.
- In the above Figure 6.16, although most of the data is at the regions pointed out, we are not interested in those regions. We are interested in finding the range of the variables at our region of interest.

3) Errors

- Quantization error: 2.415
- Topographic error: 0.000

4) Observations between PCA and SOM

- Visualizations using scatter plot becomes difficult as the number of components increase. SOM helps us in knowing the interesting region range of the components.
- The interesting region ranges are identified and reduction in design space is shown in table 6.1.

Test No: 8

Name of the Dataset	Number of Components	Decision Variable
Axis Parallel Hyper Ellipsoid [17]	10	Objective

1) From Component Planes Map

The first map shows the U-matrix and next 10 maps show the different components in the data and the last one shows the decision variable component, the objective function.

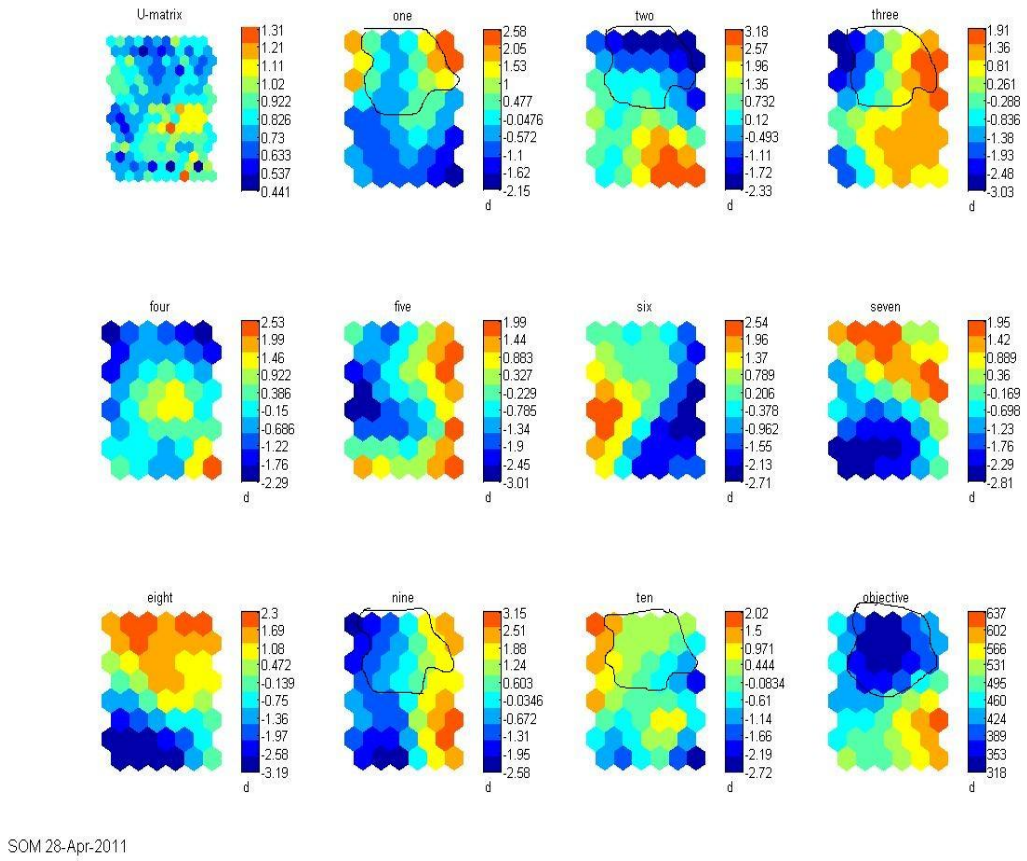


Fig: 6.17 Figure showing U-matrix, component planes and objective function of the data.

We have to minimize the objective function. Let our interested region lies between 318 to 424, which is indicated by a mark on the objective function map. Then from the map, we get all the component values as follows:

Component number	Search space	% Estimated Reduction in search space.
One	-0.572 to 2.05	40
Two	-2.33 to 0.12	50
Three	-1.93 to 1.91	10
Four	-1.76 to 0.386	50
Five	-1.34 to 1.99	15
Six	-1.55 to 0.789	50
Seven	-0.169 to 1.95	50
Eight	1.08 to 2.3	80
Nine	-1.95 to 1.88	15
Ten	-0.61 to 1.5	50

6.2: showing the results obtained, estimated reduction in search space of individual components.

- Overall reduction in search space is 41%.

2) Hits figure

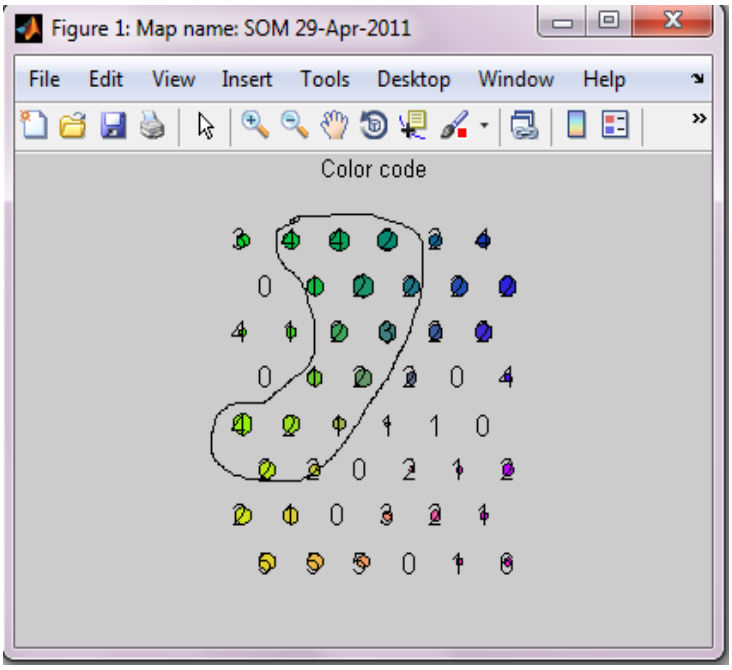


Fig: 6.18 Figure showing the number of hits by data in each cell.

The above figure shows the distribution of the data. The region marked is the place where most of our input data lies and it can also be visualized from the U-matrix diagram in the component planes map.

Most of the data is present in the interesting region of the objective function. These kind of datasets give us the range of the components more exactly.

3) Obtained Errors

- Quantization error: 2.463
- Topographic error: 0.050

CHAPTER 7

Conclusion and Future Work

7.1 Conclusion

While constructing MDO systems, visualization of the design search space is very important. As it helps in saving the design time, gives efficient design with the help of backward design. The visualizations generated should be easy in understanding and comparing them with other visualizations. When the data needs Vector Quantization and Multi dimensional scaling in visualizing the design space, techniques more efficient than PCA and SOM should be applied as they have their limitations. Usage of expert systems should be introduced to guide the engineer with the past knowledge.

7.2 Future Work

- Inclusion of Rough Sets in DSV helps in dimensionality reduction of the design variables, by which design search space is reduced.
- The concept of semantic mapping can be introduced in DSV to extract interesting features in the design space.
- We can also extend this as a plug-in to MDO problems by considering multiple objective functions.

REFERENCES

[1] Xue-Zheng Chu, Liang Gao, Hao-Bo Qiu,Wei-Dong Li, Xin-Yu Shao. "An expert system using rough sets theory and self-organizing maps to design space exploration of complex products"(2010).

[2] Andy J.Keane, Prashanth B.Nair. "Computational Approaches for Aerospace Design". John- Wiley and sons.616 pages, June 2005. ISBN 0-470-85540-1.

[3] Juha Vesanto. "SOM-Based Data Visualization Methods". Nov 1999.

[4] www.ocw.mit.edu/courses/engineeringsystemsivision/esd77multidisciplinarys ystem- design-optimization-spring-2010/lecture-notes/.

[5] www.en.wikipedia.org/wiki/Self-organizing_map

[6] www.cis.hut.fi/projects/somtoolbox

[7] www.combio.cs.brandeis.edu/COSI101A/slides/SOM.pdf

[8] Juha Vesanto, Johan Himberg, Esa Alhoniemi. "SOM Toolbox for Matlab", 5.April 2000. ISBN 951-22-4951-0.

[9] Tetsuo Furukawa. **"**SOM of SOMs: Self-organizing Map Which Maps a Group of Self-organizing Maps". Volume 22 Issue 4, May, 2009, Journal Neural Networks archive Elsevier Science Ltd.

[10] C. M. E. Holden and A. J. Keane, "Visualization Methodologies in Aircraft Design", in Proc. 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conf., AIAA 2004-4449, Albany (2004).

[11] Lindsay I Smith, "A tutorial on Principal Components Analysis", February 26, 2002.

[12] Box, G. E., Hunter,W.G., Hunter, J.S., Hunter,W.G., "Statistics for Experimenters: Design, Innovation, and Discovery", 2nd Edition, Wiley, 2005, ISBN 0471718130.

[13] www.en.wikipedia.org/wiki/Design_of_experiments

[14] www.ftp.mobarakeh-steel.ir/Matlab/cd1/pdfhelp/help/pdf.../mbcmodel.pdf

[15] www.casde.iitb.ac.in/mdo/

[16] www.ics.uci.edu/~mlearn/MLRepository.html

[17] www.geatbx.com/docu/fcnindex-01.html#P89_3085

[18] www.en.wikipedia.org/wiki/Voronoi_diagram

Appendix

User manual for SOM MATLAB GUI

som_select function is used to generate the GUI to select the cells on the SOM map. For this function to work, we should generate the SOM figure either by using *som_show* function or *som_show_add*. The handle generated by the SOM figure is passed to the *som_select* function.

This som_select function uses many sub functions such as draw_colorselection, find_patch, som_select_gui, etc.,

som_select generates a GUI showing the number of classes, each class with a different colour. The GUI contains two radio buttons each for select and clear and three pushbuttons each for close, clear and ok.

Role of mouse buttons:

Left button: Pressing the left mouse button creates a dot on the map, and pressing for the second time draws a line from first point to the second point. In this way a closed figure, polygon can be formed.

Right button: clicking on the right button in the polygon either select/clear the cells.

Middle button: It is used to select/clear the cells individually on the map.

Select Radio Button:

Choose the select button, draw a polygon on the map and select a colour/label on the GUI, then the cells in the region are converted to this colour/label.

Clear Radio Button:

This button clears (removes the cells from the map) the cells on the map in the polygon selected.

Use of SOM GUI in Design Space Visualization

In DSV, we are interested only in some of the regions on the map. so, we select a region on the SOM Figure removing the uninterested regions/cells using this GUI and we get the class labels. Actually we should be able to select a region on the map and give a label to it. This is mentioned under the future work.

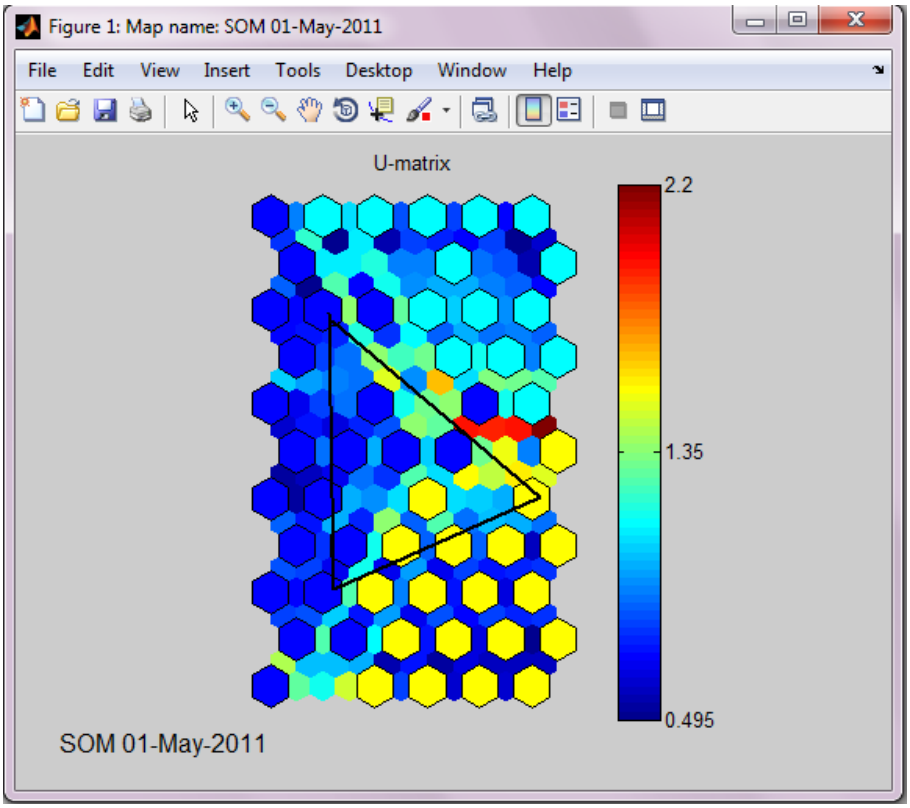


Figure showing GUI for SOM map showing selection of a region on the map.

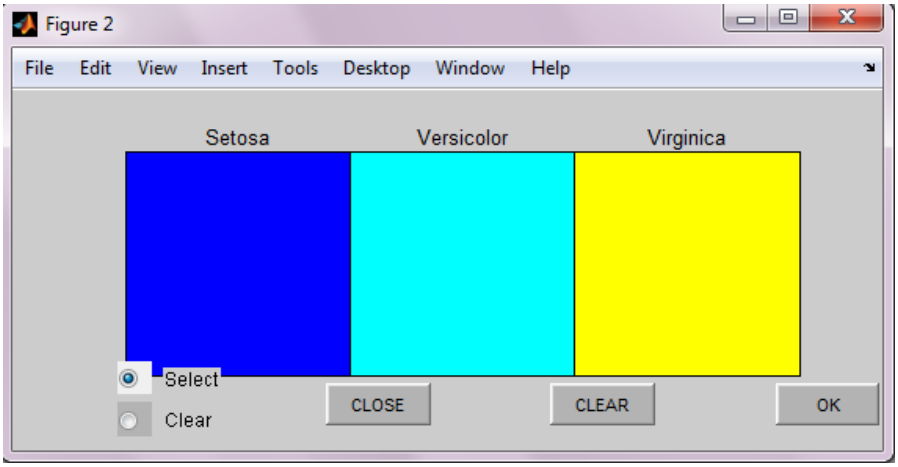


Figure showing selection of classes on the GUI.