Noname manuscript No.
(will be inserted by the editor)

Design Space Exploration and Optimization Using Self Organizing Maps

Thole Sidhant Pravinkumar ·
Palaniappan Ramu

Received: date / Accepted: date

Abstract Identifying Regions of Interest (RoI) in the design space is extremely useful while building metamodels with limited computational budget. Self Organizing Maps (SOM) is used as a visualization technique for design space exploration that permits identifying RoI. Conventional implementation of SOM is susceptible to folds or intersections that hinder visualizing the design space. This work proposes a modified SOM algorithm to address folds, allowing smoother input and performance space visualization. The modified algorithm enables identification of RoI and additional sampling in the identified RoI allows building accurate Kriging metamodel, which is then used for optimization. The proposed approach is demonstrated on benchmark nonlinear analytical examples and two practical engineering design examples. Results show that the proposed approach is highly efficient in identifying the RoI and in obtaining the optima with less samples.

Keywords Design Space Exploration · Self Organizing Maps · Design of Experiments

Thole Sidhant Pravinkumar
Indian Institute of Technology, Madras
E-mail: tholesidhantp@gmail.com

Palaniappan Ramu
Indian Institute of Technology, Madras
E-mail: palramu@iitm.ac.in

1 Introduction

In the context of design optimization that involve expensive computer models, metamodels are used to alleviate the prohibitive computational expense. In order to build an accurate metamodel, one needs to have information to some degree about the true function's nonlinearity in the design space. This information influences the choice of basis functions, kernels and the minimum sample size required. For the same sample size, accuracy of the metamodel deteriorates with the increase in size of design space and dimensions. Therefore, metamodel accuracy, number of samples, size of design space or number of dimensions are always a trade-off. For a given sample size, researchers address the above challenges using dimension reduction techniques (Chowdhury et al., 2009; Constantine et al., 2014; Berguin and

Mavris, 2015; Diez et al., 2015) and reduction of the design space size by decomposing it into multiple Regions of Interest (RoI) (Wang and Simpson, 2004; Wang and Shan, 2004; Missoum et al., 2007; Tseng et al., 2014). In the dimension reduction techniques, the underlying idea is to identify major factors and express the function in terms of these factors. Though these techniques are helpful in reducing computational time, the reverse mapping of the low dimension to original higher dimension is not possible. Hence, design space reduction techniques are an attractive alternate.

The blanket idea in design space reduction is to identify the 'sweet spots' or RoI within the design space. Usually, additional or sequential samples are generated at the RoI. This allows for building a better global metamodel with higher accuracy locally at RoI (Seung et al., 1992; Chen et al., 1997; Mackman and Allen, 2010; Deschrijver et al., 2011; Rosenbaum and Schulz, 2012; Rumpfkeil et al., 2012; Douak et al., 2012; Aute et al., 2013; Jiang et al., 2015; Jin et al., 2016; Beck and Guillas, 2016; Liu et al., 2018) or an accurate local metamodel only at the RoI (Ganapathy et al., 2015; Boursier Niutta et al., 2018), depending on the application. In both, the goal is to gain in accuracy with relatively lesser samples. Wang and Simpson (2004) used fuzzy clustering to reduce the design space and built metamodel in the reduced space, to find optima. Wang and Shan (2004) proposed a methodology which uses rough sets based method to decompose design space into subspaces for multi-objective optimization. Missoum et al. (2007) used computational geometry techniques and support vector machines to decompose the design space into safe and fail regions. The boundaries used for decomposition is then employed to estimate reliability. Tseng et al. (2014) proposed design space reduction using simplex approach to identify region exhibiting global optimal point and observed it to perform better than other traditional global search approaches in analog design applications. The overarching idea in all these approaches is to understand the design space so that the RoI can be identified.

In being able to understand or explore design spaces rather than just finding the optimal design, designers can make more informed decisions. This process called the Design Space Exploration (DSE) refers to locating the peaks and valleys, flat regions that correspond to robust regions, decomposing design space into favourable and unfavourable design regions, and providing insights on the function variation with respect to different variables (Koch et al., 2002; Shan and Wang, 2004; Curtis et al., 2013; Gan and Gu, 2019). It is very useful for a DSE approach to have visual capability because it drives informed decision making. In this work, we propose using Self Organizing Maps (SOM) to develop a unified framework for DSE, which can be used in higher dimensions as well.

SOM is an unsupervised neural network, which is generally used for data mapping and currently being successfully used in data mining and visualization (Kohonen, 1997; Torkkola et al., 2000; Song et al., 2018). SOM helps in representing multidimensional data onto multiple 2D plots. SOM can map non-linear relationships within multidimensional data into simple geometric relationship among their nodes in the 2D plot. The main advantage of SOM is that it preserves the topology of the input space. Obayashi and Sasaki (2003) demonstrated the use of SOM to visualize tradeoffs of Pareto solutions in objective space for engineering design obtained from evolutionary computation. Parashar et al. (2008) illustrated the use of SOM for visualization and design selection for multi objective Pareto data. Chu et al. (2010) employed SOM on preliminary optimization data to understand the relationship between design variables and objective function and thereby reducing the design space. Ito and Dhaene (2013) proposed SOM based adaptive sampling and show that feasible samples generated by SOM is larger compared to a random sampling approach. Qiu et al. (2016) used SOM to understand the relationship between design variables and response, and to identify the RoI in design space.

The performance of SOM is usually measured in terms of its Topographic Error (TE) and Quantization Error (QE). TE is a measure of how well SOM captures the structure of the function while QE is a measure of average distance of the data points to the nodes to which they are the closest. SOM are prone to intersections and folds, leading to high TE. Current work presents a modified SOM algorithm to address folds such that the TE is less. In the modified SOM algorithm, the proposal is to use only design variables to find the most similar SOM node. Later, the SOM nodes are updated based on the response values. The SOM is then used to identify the RoI in the function space. Upon identifying the RoI, additional samples are used to construct a highly accurate local metamodel. Demonstration of the proposed approach on analytical and engineering examples reveal the ability of the modified SOM to identify RoI effectively which then enables construction of local metamodels with better accuracy and with relatively less samples. It is important to note that SOM can operate on any number of dimensions. The rest of the paper is organized as follows: Section 2 discusses the existing visualization techniques and their limitations followed by SOM and the issues with the conventional SOM algorithm in Section 3. Section 4 presents the proposed modified SOM approach and its demonstration on numerical examples followed by discussion in section 5 and conclusion in Section 6.

2 Existing visualization techniques

Visual representation of the design space or the underlying function characteristics can drive informed decision making. Hence it is imperative for the DSE to possess a strong visual capability. Most of the tools and techniques used for DSE use projections of axes to visualize. Usually, the projections have limitations in capturing factor interactions and succumb in higher dimensions. Forrester et al. (2008) discuss techniques such as tile plot and nested axis plot to visualize design space. Other widely used techniques include scatter plot and parallel axes plots. None of these techniques capture factor interactions and often operate by fixing a factor(s) and varying other factors. In addition, they may also be limited on the number of dimensions they can operate on. In the following, we discuss the existing visualization methods on the 4 variable Griewank function, given in (1) for illustration purpose.

$$f(x) = \sum_{i=1}^{4} \frac{x_i^2}{4000} - \prod_{i=1}^{4} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{1}$$

Tile plots, shown in Figure 1 use matrices of function contours in which a baseline value in each dimension is picked and held constant while sweeping all combina-

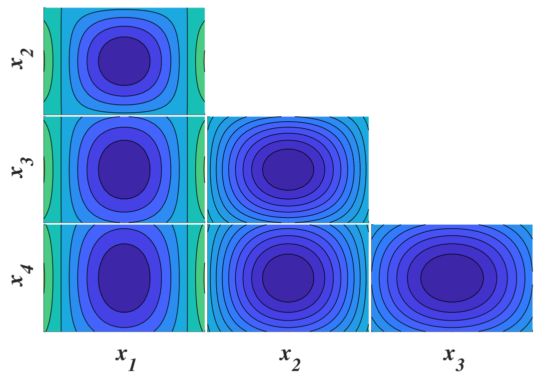

Fig. 1 Tile plot

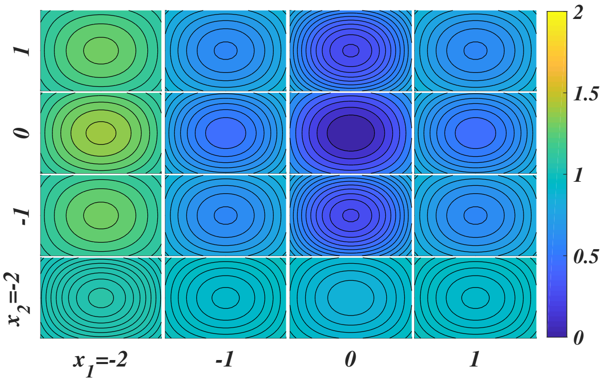

Fig. 2 Nested axes plot

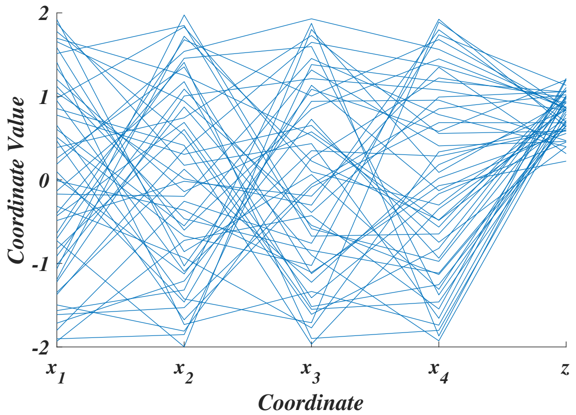

Fig. 3 Parallel axes plot

tions of other dimensions. In Nested axes plot shown in Figure 2 two variables are plotted against each other (x_3 and x_4) while the other two variables (x_1 and x_2) are fixed. The variables in the outer loop are varied forming a tile of plots. Here, one is limited to only plot four input dimensions. From a visualization perspective, in both of these methods, other than the variables that are plotted, remaining variables are kept constant. As a consequence, factor interactions are not captured.

A multidimensional visualization method that is widely used is the parallel axes plot. In parallel axes plot shown in Figure 3, dimensions are represented as abscissa and each data point is represented as poly-line connecting respective variable values. Though this approach can be used to visualize any number of dimensions, under many data points and large dimensions, the plot becomes cluttered. Radial coordinate visualization (RadVis) maps high dimensional data to 2D. It considers a point in 2D space connected by equally spaced points on the circle where number of points is equal to the number of dimensions, with a spring of constant equal to the value of the coordinate in the respective dimension. This point is allowed to move freely and the point where it comes to equilibrium is the 2D mapping of higher dimensional points. Though it is very simple to construct, it is unable to convey the shape of the data cloud. Ibrahim et al. (2016) used 3D-Rad Vis to visualize shape of multi objective optimization data. Bubble chart uses color and the size of the bubble as additional features to represent additional dimensions. However bubble charts are applicable only upto 5 dimensions (Ashby, 2000). Scatterplots are used for bivariate data analysis. In scatterplot two attributes are projected on x-y axes and others are kept constant. Generally matrix of multiple scatterplots are used to understand the relation between different attributes. If the number of attributes increases, the number of plots increases quadratically. Therefore, it is desirable to develop an easily implementable approach to visualize design space for DSE, which is independent of dimensions, assists identifying RoI, allows capturing factor interactions and correlations.

3 Self Organizing Maps

SOM is a type of Artificial Neural Network which is based on unsupervised learning. It takes high dimensional data as input and produces a low dimensional, generally, two dimensional representation of the input data. It also preserves the topology of the high dimensional input space. Here topology preservation implies high dimension to low dimension mapping that conserves the relative distances between input data points. SOM also possess generalization capability which implies that the map can characterize inputs it has never encountered before. SOM is a two dimensional grid of

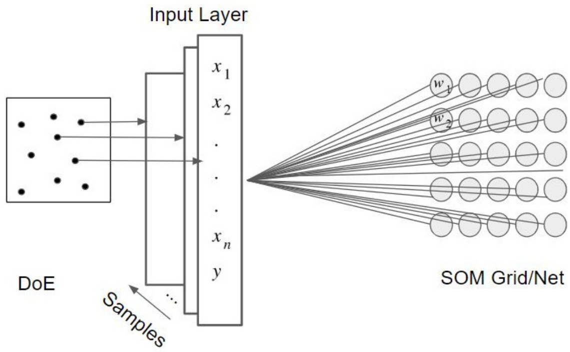

Fig. 4 SOM structure

nodes as shown in Figure 4. Each node corresponds to vector of weights which is also called codebook vectors of size equal to the number of columns in the dataset on which SOM is trained. Nodes can be connected to each other by rectangular or hexagonal topology. For example, if there are n design variables in the dataset namely $[x_1, x_2, ..., x_n]$ and response function y, then each input vector to SOM will be $v_i = [x_{i1}, x_{i2}, ..., x_{in}, y_i]$ and each SOM node w_j will have weight vectors $[m_{j1}, m_{j2}, ..., m_{jn}, m_{jout}]$. First n weights correspond to design variables and the last weight corresponds to response. Dimension of the weight vector corresponding to each of the node is the same as the dimension of the input vector which is equal to $n + 1$. Values of weights are usually initialized by linear initialization. In linear initialization, codebook vectors are initialized in space that is spanned by two eigenvectors which has the highest eigenvalues. Training consists of finding the winning node (node that is closest to the data point) using similarity measure and updation of weight vectors of nodes in neighbourhood of winning node and itself. After training is repeated for all the samples, SOM takes the shape of the input space. Detailed SOM algorithm is discussed below.

3.1 Conventional SOM algorithm

Input: Dataset comprising both design variables and responses. Let the number of data points in dataset be p.

Step1: Specify number of nodes k, in the SOM net and set the maximum iterations t_{max}. Usually k is taken as $5\sqrt{p}$ (Estevez, 2012).

Step 2: Compute the eigenvalues for the normalized data points. Record the largest two eigenvalues and their first two eigenvectors.

Step 3: For each node, all weight vectors w_j, are initialized in the direction along the recorded eigenvectors in Step 2.

Step 4: For i^{th} data point v_i, compute the distance between j^{th} node and the data point as given in (2):

$$D_{ij} = \sqrt{(v_i - w_j)^2} \tag{2}$$

where, $i \in [1, p]$, $j \in [1, k]$. For each v_i, the node corresponding to the least D_{ij} is called Best Matching Unit (BMU). v_i contains the design variables, and response $[x_{i1}, x_{i2}, ..., x_{in}, y_i]$ and w_j will have weights for j^{th} node $[m_{j1}, m_{j2}, ..., m_{jn}, m_{jout}]$, where n is the number of design variables, m_{out} is the weight that corresponds to y.

Step 5: Upon finding the BMU, the weights of j^{th} node and nodes in its neighbourhood will be updated as follows:

$$s = (t - 1) \times p + i \tag{3}$$

$$\phi(s) = e^{\left(\frac{-Dist^2}{2\sigma(s)^2}\right)} \tag{4}$$

$$\sigma(s) = \sigma_0 + \frac{(s-1) \times \sigma_0}{p \times t_{max} - 1} \tag{5}$$

where s is an update step, t is an iteration number and $Dist$ is the distance of a node to BMU at update step s, and the neighbourhood function $\phi(s)$ is given by (4). Size of neighbourhood at update step s is given by (5). Here, σ_0 is the initial radius and is taken as 1. Neighbourhood radius decays linearly with iterations. Every node in the neighbourhood radius of BMU will get updated according to (6).

$$w_j(s+1) = w_j(s) + \phi(s) \times L(s) \times (v_i - w_j) \tag{6}$$

Learning rate $L(s)$ is presented in (7)

$$L(s) = C \times \frac{\alpha_0}{C + s} \tag{7}$$

$$C = \frac{p}{100} \tag{8}$$

where α_0 is the hyper parameter with default value of 0.5. C is a constant and computed from (8).

Step 6: Compute Step 4 and Step 5 for each sample $i \in [1, p]$. This completes one iteration. Iterations are repeated until a prescribed number of iterations or error convergence metrics which are discussed later, are met.

Output: Trained SOM weights, U-matrix (Unified Distance Matrix), component planes.

3.2 Unified Distance Matrix (U-Matrix) and component planes

U-matrix represents distances between nodes of trained SOM. The distance between adjacent nodes are color coded in U-matrix as shown in Figure 5 for an arbitrary example. Information of distances can also be useful to identify the clusters of input space. Figure 6 represents the distribution of weight coordinates of SOM nodes which corresponds to the variable x. For example, if SOM trained over data with columns $[x, y, f(x, y)]$ to

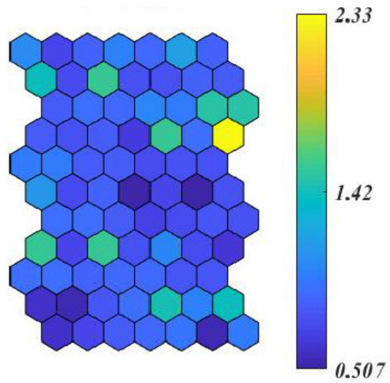

Fig. 5 U-matrix

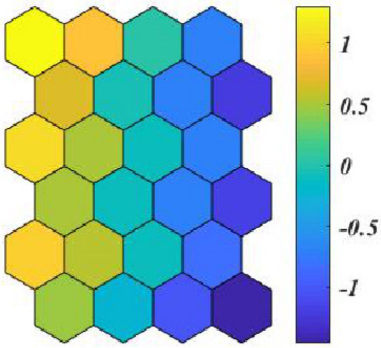

Fig. 6 Component plane of variable x

give weights $[m_1, m_2, m_f]$, the x component plane will represent values of m_1. Component planes that look similar suggest that the respective variables are correlated.

3.3 Folds in conventional SOM implementation

As mentioned earlier, conventional SOM tends to fold and self intersect (Lopez-Rubio, 2013) affecting visualization and understanding design space. Following examples illustrate the effect of folds on visualization interpretability for multi-dimensional design space.

Let us consider interpreting the function in (9), using SOM. Input consists of 20 LHS points and corresponding response values.

$$z = x^2 + y^2, where\ x, y \in [-2, 2] \quad (9)$$

Figures 7, 8 present the trained SOM net of dimension 6x4 over the original surface and its $x - y$ projection respectively. Figure 9 depicts the U-matrix and component plane. It can be observed from z component plane plot that the convexity of function is captured by SOM. In x component plane, as x increases from bottom right to top left diagonally, z component plane values first decrease and then increase, which is similar to the original function. Similarly, in y component

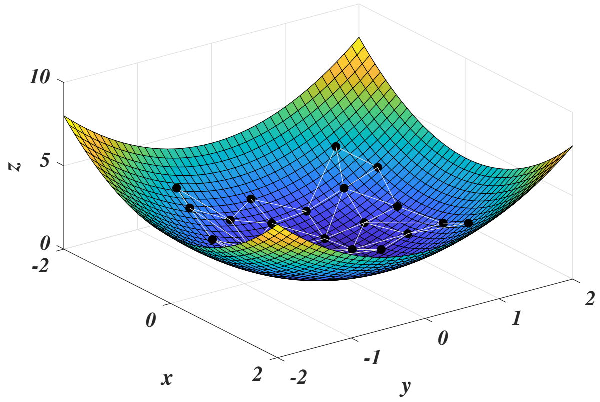

Fig. 7 SOM without folding over the convex surface given by (9)

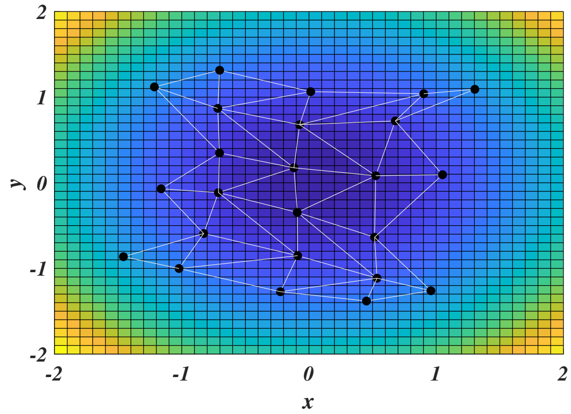

Fig. 8 SOM without folding over the convex surface given by (9). $x - y$ projection

plane as it increases from bottom left to top right, the z value first decreases and then increases. This exercise shows that SOM is capable of representing the shape of the function and can also provide information about trend of output variable with respect to each input variable, if there is no folding in the SOM. Figure 10 shows the SOM net trained on another set of 20 LHS points. Here, the net is folded leading to limitations in understanding the function. It can be observed that there is an abrupt change in the color of component plane and it can be attributed to the SOM self intersecting or folding. Comparing the component planes in Figures 9 and

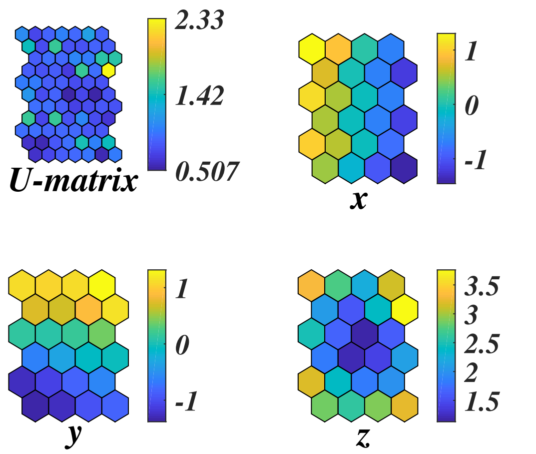

Fig. 9 U-matrix and component plane of SOM over (9)

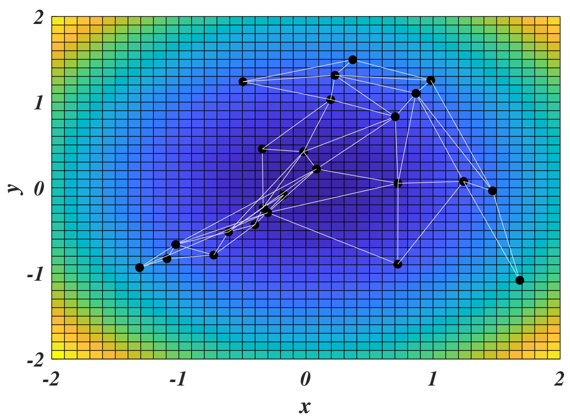

Fig. 10 Folded SOM over the convex surface given by (9). $x-y$ projection

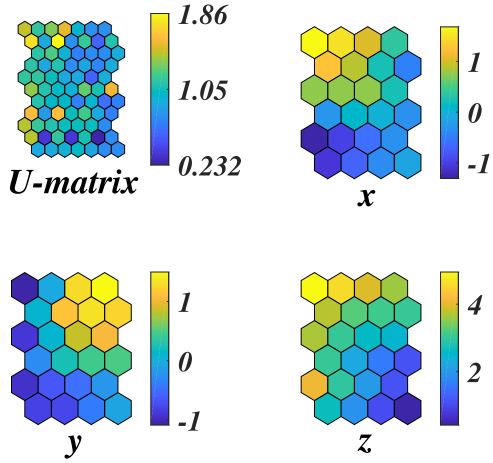

Fig. 11 U-matrix and component plane of folded SOM over (9)

11, one can directly understand how component planes in Figure 11 do not capture actual variation of function as in Figure 9. Therefore, while visualizing multivariate functions, self intersection or folds affect visualization of SOM. Function can be seen as mapping from elements of one set to another set where, one element of the first set should always map to one element of the second set. Folding in SOM is equivalent to a mapping where one element from the first set (weights corresponding to the design variables) will be mapped to multiple elements in the second set (weights corresponding to response).

3.4 Quality measures of SOM

An important feature of SOM is that, it preserves the topology of the input dataset (Kiviluoto, 1996). Topology is said to be preserved when two neighbouring data points are mapped to the two SOM nodes which are neighbours to each other. The quality measures that are widely used to measure the performance of SOM are: Topographic Error (TE) and Quantization Error (QE).

TE is a measure of how well the structure of the input space is preserved by SOM. It allows capturing local discontinuities (Kiviluoto, 1996). For topology to be preserved, the minimum requirement is that the 2^{nd} BMU should be in the immediate neighbourhood of 1^{st} BMU. That is, for each data point, the 2^{nd} least distant node from the data point should be near to the 1^{st} least distant node. Here, neighbourhood refers to 6 nodes that surrounds the BMU. TE is the ratio of number of mapped nodes whose 2^{nd} BMU is not in the immediate neighbourhood of 1^{st} BMU, to the total

$$\text{TE} = \frac{1}{p}\sum_{1}^{p} t(v_i)$$
$$\text{where, } t(v_i) = \begin{cases} 1\ if\ w^1_{ibmu}\ and\ w^2_{ibmu}\ are\ in\ neighbourhood\ of\ each\ other \\ 0\ otherwise \end{cases} \tag{10}$$

number of samples. TE is presented in the (10). w^1_{ibmu}, w^2_{ibmu} are first and the second nearest node to v_i respectively and p is the number of data points. However topographic error does not identify larger distortions in the manifold (Breard, 2017).

QE is the measure of the average distance between the data points and SOM nodes to which they are mapped. QE is given in (11).

$$\text{QE} = \frac{1}{p}\sum_{1}^{p} \parallel v_i - w_{ibmu} \parallel \tag{11}$$

where p is the number of samples, v_i is the i^{th} sample, w_{ibmu} is the weight vector of the SOM node that is mapped to v_i. Smaller the value of QE, better is the fit. Since QE considers only the distance between data points and their respective mapped node and not how the nodes are organized it cannot account for folds and intersections.

4 Methodology

In conventional SOM, during an iteration, the BMUs that were identified earlier might get altered due to the neighbourhood rule. The iterations are repeated to the specified convergence of error or allowable function evaluations. During such iterations, there is a possibility for the net to self-intersect and fold. In the current work, we propose a modified SOM algorithm to address the folds and self intersection issue in conventional SOM implementation. This can further be used to identify peaks and valleys in response, leading to zoning RoI for optimization.

4.1 Modified SOM algorithm

Our interest is to model functions of the form $y = f(x)$. Hence, first the BMUs are found by using data from the input space alone (as against using both input and output values in the conventional SOM) as shown in Figure 12 and the weights of the respective node and the ones in its neighbourhood are updated as the difference between the data point and respective response. This is

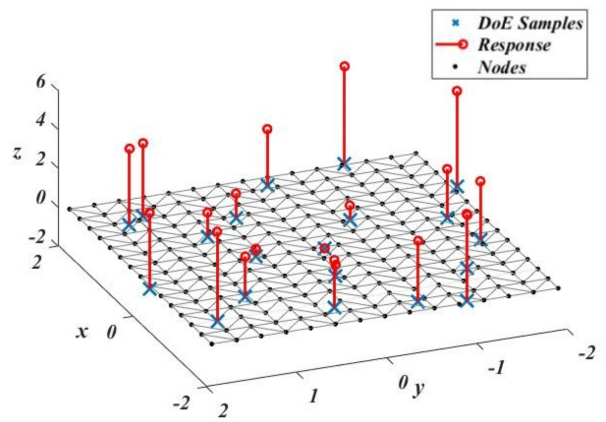

Fig. 12 Modified SOM algorithm: BMU selection

explained in Figure 13. The steps involved in the proposed algorithm are presented below:

Input: Dataset including design variables and their responses

Step 1 to Step 3: are same as conventional SOM al-

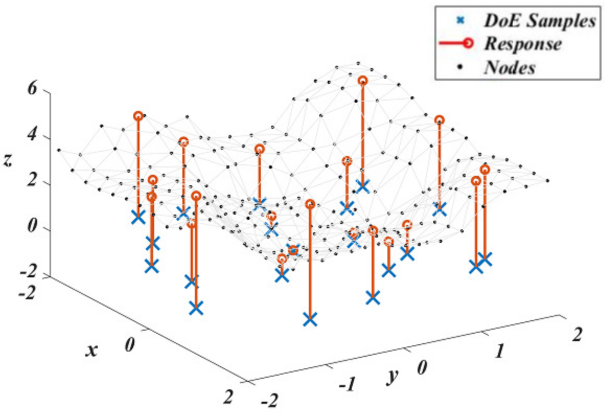

Fig. 13 Modified SOM algorithm: Update step

gorithm

Step 4: Compute BMU which is node j with least D'_{ij}. Consider only input design variables and their weights.

$$D'_{ij} = \sqrt{(v'_i - w'_j)^2} \tag{12}$$

For instance, in (12) v'_i contains the design variables $[x_{i1}, x_{i2}, ..., x_{in}]$ and w'_j will have weights $[m_{j1}, m_{j2}, ..., m_{jn}]$ for j^{th} node. Where n is the number of design variables. It is to be noted that in the modified algorithm we are not considering m_{jout} in w'_j and y_i in v'_i while finding BMU.

Step 5: Upon finding the BMU, the weight m_{jout} of BMU and its neighbourhood nodes will be updated as in (13) using (3), (4), (5) and (7):

$$m_{jout}(s+1) = m_{jout}(s) + \phi(s) \times L(s) \times (y_i - m_{jout}) \tag{13}$$

Since the update step uses only the response values and not the input variables, the self intersection or folding is avoided in this approach. We calculate TE and QE for the proposed modified SOM using weights corresponding to inputs only.

4.2 Numerical examples

In this section, we use the modified SOM algorithm to visualize the design space and find RoI for optimization of analytical functions and engineering examples. Examining component plane of response, one can select the threshold value that corresponds to the values of RoI. In a minimization problem, codebooks of nodes for which output weights are lesser than threshold value and respective samples to which they are mapped are filtered out. Using maximum and minimum values of these filtered samples, RoI can be identified. Further, we add more points in the RoI and build metamodel for optimization. Hyperparameters of metamodel are chosen using cross validation errors. Since, SOM is dependent on DoE, we have used 5 different DoEs for all the numerical examples and results are presented in section 5.

4.2.1 Six hump Camelback function

Six Hump Camelback function given in (14) is one of the well-known examples that is used for testing global optimization algorithms. Figure 14 shows the contour plot for the same.

$$z = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, x_{1,2} \in [-2, 2] \tag{14}$$

As the name suggests, this function has six local optima and it has two global optimal points which lie at (0.0898,-0.7127) and (-0.0898,0.7127). The function value is -1.0316 at both the global optima. Our goal here is to use modified SOM to identify RoI that correspond to optimal points (minima) with limited samples. LHS of size 20 is used. The ranges of x_1 and x_2

are $[-2, 2]$. z from (14) is computed at these 20 sample points. Modified SOM with 24 nodes is applied over the 20 samples with TE=0 and QE=0.42 compared to conventional SOM built on same data with TE=0.1 and QE=0.50. U-matrix and the component planes for proposed SOM is given in Figure 15. Since we are interested in regions where the minima will lie, we choose a threshold value of 8 in z component plane and identify regions that correspond to this threshold value. It can be deduced that lower values of z can be found in the middle ranges of x_2. However, x_1 has a spectrum of values for the z below threshold value as shown in Figure 15. The codebook vectors whose weight values corresponding to z are less than the threshold value are filtered, followed by filtering samples to which these nodes are mapped. From the filtered set of samples, new ranges of x_1 and x_2 are calculated. Reduced ranges are given as [-1.4694,1.3436], [-0.8185,1.4977] respectively. 9 samples from the initial 20 LHS points fall in this RoI. Further, 21 LHS samples are added into RoI and Kriging metamodel with Gaussian correlation, linear regression and R^2_{pred} value of 0.99 is built over these 30 samples. Optimization is carried out over this metamodel and results are presented in Table 1. It is to be noted that our focus is not to get the optima but zeroin on the RoI with limited samples and explore the design space in the process, with lower samples.

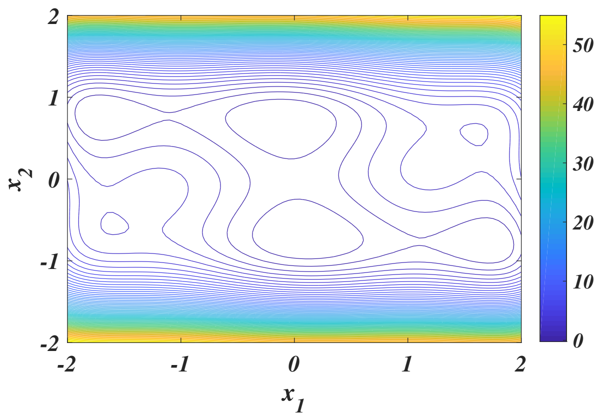

Fig. 14 Contours of Camelback function

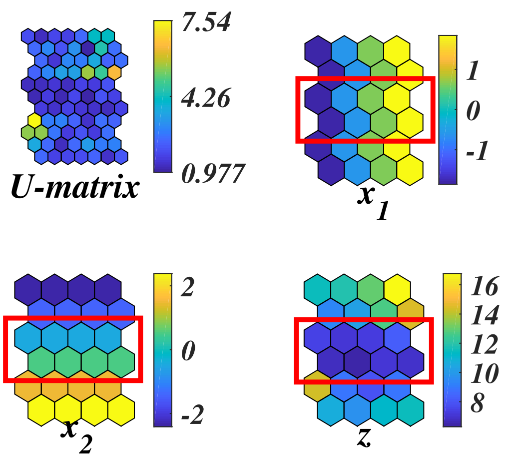

Fig. 15 U-matrix and component plane of Camelback function. Boxes represent the RoI in z component plane and corresponding ranges in x_1 and x_2 component plane.

4.2.2 Rosenbrock function

The Rosenbrock function given by (15) is represented in Figure 16.

$$z = (1 - x_1)^2 + 100(x_2 - x_1^2)^2, \quad x_{1,2} \in [-5, 5] \tag{15}$$

It has a global minima at (1,1) and the function value at this point is 0. A LHS with 50 data points is used. Modified SOM with 36 nodes is applied to these 50 samples with TE=0 and QE=0.91 compared to conventional SOM built on same data with TE=0.06 and QE=0.88. U-matrix and the component planes are presented in Figure 17. It can be observed that lower values of z corresponds to middle ranges of x_1 and low

Table 1 Camelback function: SOM based RoI identification and optimization

Algorithm	X	f_{val}	Number of samples	TE	QE
(Qiu et al., 2016)	(0.0895, -0.7105)	-1.0316	100	-	-
Modified SOM	(-0.0778,0.7253)	-1.0296	41	0	0.42

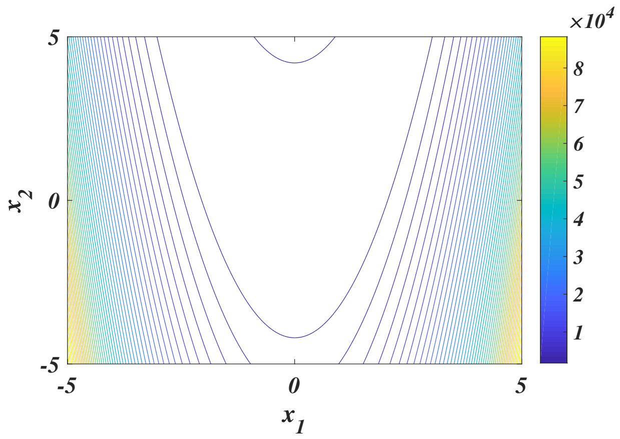

Fig. 16 Contours of Rosenbrock function

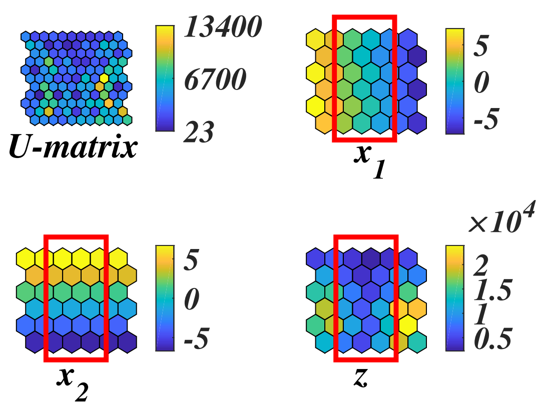

Fig. 17 U-matrix and component plane of Rosenbrock function. Boxes represent the RoI in z component plane and corresponding ranges in x_1 and x_2 component plane.

to high ranges in x_2. Since we are interested in a minima, a threshold value of 8000 was selected in z component plane. The codebook vectors whose z component weight values are less than 8000 are filtered out followed by samples to which these nodes are mapped. From this filtered set of samples, the new reduced ranges of x_1 and x_2 are [-3.3554,2.8335], [-4.9993,4.8790] respectively. 28 samples from the initial 50 DoE fall in this RoI. Further, 42 LHS samples are added in the RoI and Kriging metamodel with Gaussian correlation, quadratic regression function with R^2_{pred} value of 0.99 is built over these 70 samples and used in optimization. Optimization results are compared in Table 2 and it is clearly established that the proposed method is efficient in identifying RoI and optima with relatively less samples.

4.2.3 4 variable heat exchanger design example

In this section, a design example of heat exchanger (Seepersad et al., 2004; Qian et al., 2006) that is used to dissipate heat generated by electronic components such as a microprocessor is considered to demonstrate the proposed methodology. Device dimensions namely width (W), height (H) as shown in Figure 18 are kept constant at values of 9 and 17.4 mm respectively. The depth D of the heat exchanger is considered to be 25mm and is kept constant. The objective is to maximize the total rate of steady-state heat transfer. The mass flow rate and the temperature of entry air, thermal conductivity and, the temperature of the heat source are considered as design variables. The ranges, units of each of

Table 2 Rosenbrock function: SOM based RoI identification and optimization

Algorithm	X	f_{val}	Number of samples	TE	QE
(Qiu et al., 2016)	(1.0003,1.0010)	1.6083e-05	400	-	-
Modified SOM	(1.0149,1.0309)	2.9737e-04	92	0	0.91

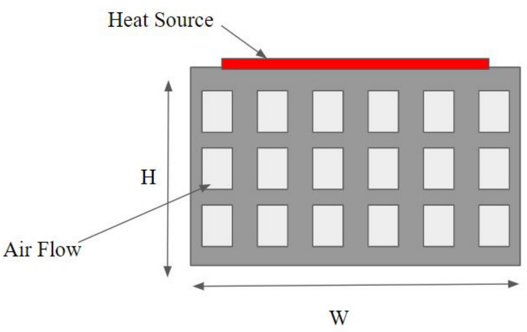

Fig. 18 Compact forced convection heat exchanger

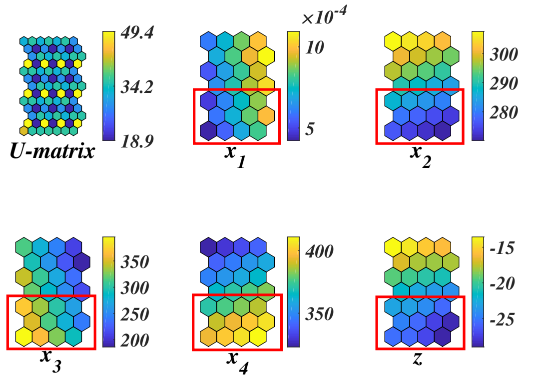

Fig. 19 U-matrix and component plane of heat exchanger example. Boxes represent the RoI in z component plane and corresponding ranges of variable component plane.

the variables are summarized in Table 3. Qian et al. (2006) considered 64 LHS DoE samples. We pretend that we have only 30 samples and select it randomly from the 64 samples. Modified SOM with 28 nodes is applied over the 30 samples. Since this is a maximization problem the steady-state heat transfer rate values are multiplied with -1. Proposed SOM with TE=0.1 and QE=13 is obtained compared to conventional SOM built on same data with TE=0.33 and QE=15.6. Figure 19 shows the U-matrix and component plane of trained SOM. Important design insights can be drawn from these plots. For example, low x_4 correspond to high z, lower to medium x_3 correspond to high z and henceforth. Also, x_4 is inversely correlated with z and x_2 is positively correlated with z. x_1 and x_3 are not linearly correlated with z. These observations match the analysis performed on the metamodel constructed with approximate simulation samples in Qian et al. (2006) but with 64 samples. Based on observations, a z component plane threshold value of -24 is chosen and the codebook vectors for which z weights which are less than -24 are filtered and the samples which are mapped to these nodes are filtered as well. Using the filtered samples, ranges of all design variables are recalculated and the reduced ranges are provided in Table 4. 13 samples out of the initial 30 samples fall in the reduced range. Another 6 data samples that fall in this range are taken from the initial 64 LHS design runs. These 19 samples are then used to build a Kriging metamodel with cubic correlation, quadratic regression and a R^2_{pred} of 0.99 in the RoI. An optimization on this metamodel results in optima at (0.0009,273.4300,353.7499,399.4500) and

Table 3 Design variables for heat exchanger example

Design Variables	Symbol	Unit	Ranges
Mass flow rate of entry air	x_1	Kg/s	[0.00055,0.001]
Temperature of entry air	x_2	K	[270,303.15]
Thermal Conductivity	x_3	W/mK	[202.4,360]
Temperature of heat source	x_4	K	[330,400]

Table 4 Reduced ranges of design variables for heat exchanger example

Variables	Reduced ranges
x_1	[0.0006,0.001]
x_2	[273.43,296.75]
x_3	[207.55,353.75]
x_4	[371.60,399.45]

function value at this point is 34.78. However we do not have access to the true data. Nevertheless, the optimal design variables obtained by Qian et al. (2006) are in the reduced design space that is reported in this work, which is obtained by 36 samples as against 64 used in Qiu et al. (2016).

4.2.4 6 variable High Strength Concrete (HSC) example

HSC is a composite of water, cement and other chemical admixtures. HSC is generally used in earthquake zones. We are using HSC cost vs performance trade-off as example to validate the proposed methodology. The goal is to identify the composition of HSC that will provide high performance and lower cost. Variables and their ranges are provided in Table 5. Compressive strength and slump are constraints and others are design variables. We use the data of 86 samples provided in Lim et al. (2004); Baykasoglu et al. (2009). We pretend that we have access to only 30 random samples out of the 86 sample dataset. SOM with 28 nodes is applied over these 30 samples. U-Matrix and component planes are presented in Figure 20. Proposed SOM with TE=0 and QE=6.84 is obtained, compared to conventional SOM built on same data with TE=0.13 and QE=5.76. It can be observed that low z correspond to low x_5, x_6 and high x_3, x_1. Using the z component plane data, threshold for filtering has been set at 45. The codebook vectors which have z value lesser than 45 and data samples to which they are mapped are filtered. From these filtered sample, ranges of all the design variables are re-

Table 5 Variables for HSC example

Variables	Symbol	Unit	Ranges
Water to Binder ratio	x_1	%	[31,45]
Water Content	x_2	Kg/m^3	[160,180]
Fine Aggregate	x_3	%	[37,53]
Flay Ash	x_4	%	[0,20]
Air entraining ratio	x_5	Kg/m^3	[0.036,0.078]
Superplasticizer	x_6	Kg/m^3	[1.89,8.5]
Compressive Strength	F_1	MPa	40
Slump	F_2	mm	[175,250]

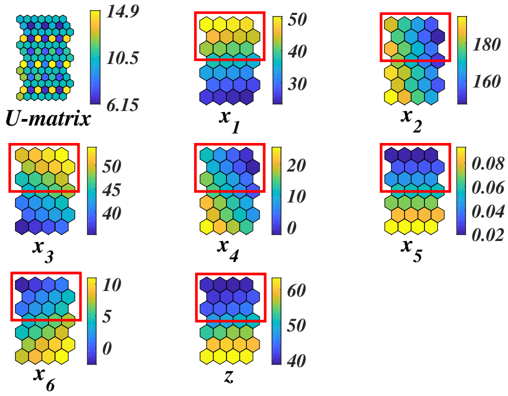

Fig. 20 U-matrix and component plane of HSC example. Boxes represent the RoI in z component plane and corresponding ranges of variable component plane.

Table 6 Reduced ranges of design variables for HSC example

Design variables	Reduced ranges
x_1	[40,45]
x_2	[160,180]
x_3	[42,53]
x_4	[0,20]
x_5	[0.036,0.045]
x_6	[1.89,4]

calculated and presented in Table 6. 13 samples of the initial 30 samples fall in the RoI. 12 additional samples from the initial 86 samples, fall in the reduced range. The total 25 samples are then used to build Kriging metamodel with exponential correlation, constant regression function and R^2_{pred} of 0.98. Optimization is carried out on this metamodel and the optima is obtained at (44.9997,170.0000,52.0000,0.0085,0.0380,1.8901) and the function value at this point is 35.0206. Qiu et al. (2016) reports an optima of 34.9097 but with 86 samples. The optimal design variables obtained by Qiu et al. (2016) are in the reduced design space that is reported in this work. The results presented in this section demonstrates the ability of the proposed SOM algorithm in allowing the designer to explore the design space, identify RoI and perform optimization, with lesser samples compared to similar approaches reported in literature. Though we have discussed upto 6 variable example, there is no limitation on the number of variables and the approach can seamlessly scale to any dimensions. Similarly, we have only solved unconstrained optimization examples. However, the idea can be extended to constrained optimization as well, which will be the scope for future work. It is to be noted that the results are completely dependent on the initial DoE, just like any other sampling approach. However, the proposed approach provides a visual cue on the parameter effects on the response or the inter parameter correlations and other related information which are important for informed decision making. The number of samples in the initial DoE or RoI is usually based on the computational budget available. However, a resource allocation problem setup for solving this is a potential future scope.

5 Discussion

In this section, we present results for all the examples with 5 different DoEs, discuss special case of the conventional SOM and, threshold selection.

5.1 Modified SOM performance for different DoEs

In order to account for sampling variability, we demonstrate the proposed approach on 5 different DoEs for all the examples discussed in Section 4. The respective results are presented in Table 7, 8, 9 and 10. The RoI is identified in all the DoEs, in all the examples. For Camelback and Rosenbrock functions, in order to compare the results for each DoE, the total number of samples in the RoI is kept the same irrespective of the number of samples that fall in the RoI from the initial sample.However, for the Heat exchanger and HSC examples, since we dont have control on the sample generation, the final number of samples in the RoI vary for each DoE.

5.2 Special case of the conventional SOM

The issue of fold in classical SOM implementation was discussed in Section 3.3. In addition to that, there are instances in which even when the SOM does not fold, the component planes are not reflective of the input space. One such example for the convex function discussed in Section 3.3 is presented here in Figure 21. Figure 22 presents the U-matrix and component planes for the same. It can clearly be observed that even though there is no fold in z there is an abrupt change in the component plane. This can limit the ability of SOM

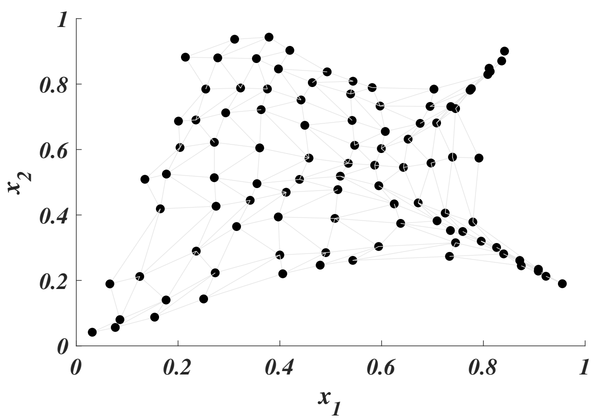

Fig. 21 SOM trained over convex function

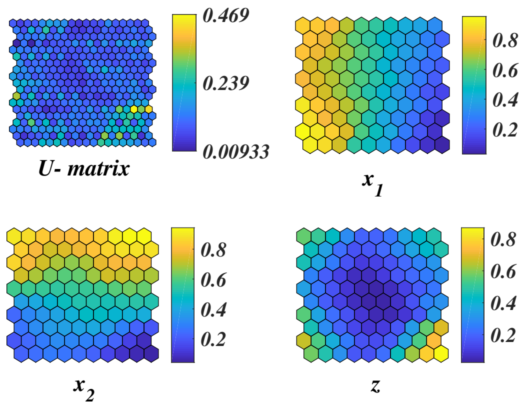

Fig. 22 U-matrix and component planes of trained SOM over convex function

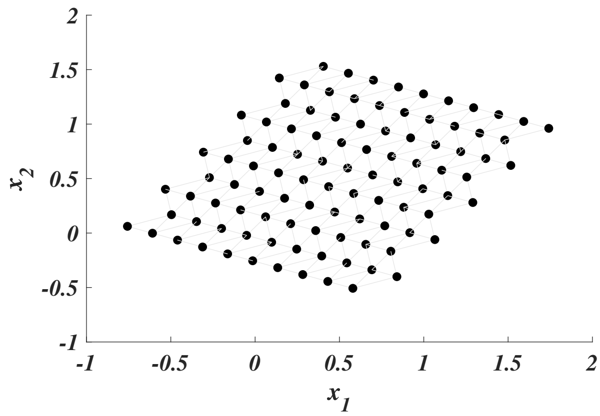

Fig. 23 Modified SOM trained over convex function

to depict trends of response with respect to the design variables. However, this is avoided in modified SOM algorithm implementation. Figure 23 and 24 represents the modified SOM over the same samples. x_1, x_2 component planes shows smooth trend and z captures the convexity of the function.

5.3 Threshold selection using SOM hit maps

SOM hit map provides the number of data points which are mapped to a certain node. Figure 25 depicts the SOM hit map over the U-matrix and component planes for the Camelback function example. Size of the red

Table 7 Camelback function: SOM based RoI identification and optimization for multiple DoEs

DoE #	x_1	x_2	# of samples in RoI	R^2_{pred}	x	f_{val}
1	$[-1.91, 1.80]$	$[-1.64, 1.29]$	13+17	0.9863	(0.09,-0.66)	-1.01
2	$[-1.86, 1.80]$	$[-1.63, 1.75]$	15+15	0.9967	(0.15,-0.76)	-1.00
3	$[-1.65, 1.96]$	$[-1.61, 1.16]$	10+20	0.9999	(-0.06,0.69)	-1.02
4	$[-1.96, 1.83]$	$[-1.19, 1.45]$	10+20	0.9765	(-0.09,0.69)	-1.03
5	$[-1.94, 1.97]$	$[-1.55, 1.61]$	13+17	0.9718	(-0.08,0.70)	-1.03

Table 8 Rosenbrock function: SOM based RoI identification and optimization for multiple DoEs

DoE #	x_1	x_2	# of samples in RoI	R^2_{pred}	x	f_{val}
1	$[-3.93, 3.65]$	$[-4.68, 4.88]$	35+35	0.99	(1.12,1.28)	0.1
2	$[-4.99, 3.76]$	$[-4.99, 4.94]$	40+30	0.99	(0.96,0.91)	0.002
3	$[-3.82, 4.38]$	$[-4.16, 4.72]$	37+33	0.99	(1.00,1.00)	0
4	$[-4.81, 4.08]$	$[-1.21, 4.91]$	27+43	0.99	(0.97,0.93)	0.03
5	$[-4.48, 4.41]$	$[-4.58, 4.83]$	41+29	0.99	(1.16,1.35)	0.03

Table 9 Heat exchanger example: SOM based RoI identification and optimization for multiple DoEs

DoE #	x_1	x_2	x_3	x_4	# of samples in RoI	R^2_{pred}	x	f_{val}
1	$[0.0006, 0.001]$	$[275.01, 296.33]$	$[273.31, 353.75]$	$[374.41, 397.78]$	6+1	0.99	(0.001,275.01,353.75,397.78)	-34.29
2	$[0.0007, 0.001]$	$[271.23, 281.47]$	$[232.64, 353.75]$	$[362.30, 398.52]$	5+3	0.98	(0.001,271.23,353.75,398.52)	-35.08
3	$[0.0006, 0.001]$	$[273.43, 299.22]$	$[273.31, 353.75]$	$[362.30, 398.45]$	14+3	0.99	(0.001,271.43,353.75,399.45)	-34.98
4	$[0.0006, 0.001]$	$[273.71, 299.22]$	$[253.25, 353.75]$	$[362.30, 399.45]$	14+5	0.99	(0.001,273.71,353.75,399.45)	-35.02
5	$[0.0006, 0.001]$	$[271.23, 296.75]$	$[323.15, 353.75]$	$[374.41, 399.45]$	5+1	0.98	(0.001,271.23,353.75,399.45)	-35.37

Table 10 HSC example: SOM based RoI identification and optimization for multiple DoEs

DoE #	x_1	x_2	x_3	x_4	x_5	x_6	# of samples in RoI	R^2_{pred}	x	f_{val}
1	$[35, 45]$	$[160, 180]$	$[42, 51]$	$[0, 20]$	$[0.038, 0.067]$	$[1.89, 4]$	17+23	0.99	(45,160,51,0,0.0380,1.89)	33.73
2	$[35, 45]$	$[170, 180]$	$[45, 51]$	$[0, 20]$	$[0.038, 0.067]$	$[1.89, 3.86]$	11+13	0.99	(45,170,51,0,0.067,1.89)	31.76
3	$[35, 45]$	$[160, 180]$	$[43, 51]$	$[0, 20]$	$[0.038, 0.059]$	$[2, 5.71]$	17+20	0.99	(45,160,51,0,0.0380,2)	34
4	$[35, 45]$	$[170, 180]$	$[42, 52]$	$[0, 20]$	$[0.038, 0.067]$	$[1.89, 3.86]$	16+17	0.99	(45,170,52,0,0.067,1.89)	32.26
5	$[40, 45]$	$[160, 180]$	$[44, 51]$	$[0, 20]$	$[0.038, 0.045]$	$[1.89, 4]$	13+20	0.99	(45,160,51,0,0.0380,1.89)	33.74

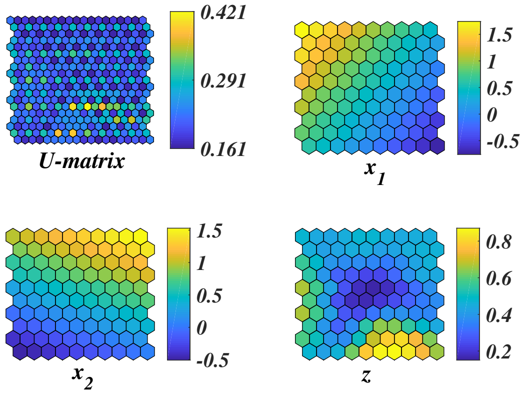

Fig. 24 U-matrix and component planes of trained modified SOM over convex function

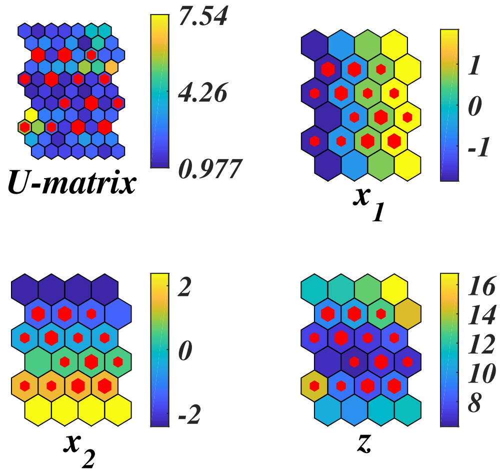

Fig. 25 SOM hit map over component planes for Camelback function

hexagon represents the number of samples mapped to the particular node. An empty hexagon means that the particular node has no hit. While solving minimization problem, one selects a lower value as the threshold and the nodes that correspond to the values lower than the threshold will become the reduced design space. While selecting a threshold, in addition to the value, we recommend to ensure that the respective node has at least one hit. Else, there is a chance that one might end up in a larger reduced space. Based on the recommendation, we select a threshold of 8 for the Camelback function example.

6 Conclusion

SOM is used to understand the design space, identify RoI and perform optimization. Modified SOM algorithm proposed in this work solves the issue of folds in conventional SOM and provides smoother component planes. Smoother component planes allows seamless visual representation of design space and also maps the relationships between response and the input design variables. Modified SOM can be trained with less number of samples and permits identify RoI. Such regions are then used for additional sampling and constructing accurate metamodels. In addition, SOM also permits understanding variable correlation and effects. Quality metrics for SOM is also discussed. The proposed modified SOM algorithm is successfully demonstrated for DSE and optimization of analytical and engineering examples, with less samples compared to that in literature.

Acknowledgements This work has been supported in part by a grant from Siemens Technology and Services Pvt Ltd. India.

Conflict of interest

The authors declare that they have no conflict of interest.

Replication of results

$MATLAB^{®}$ codes to reproduce modified SOM maps for the examples discussed are made available at this link.

References

Ashby MF (2000) Multi-objective optimization in material design and selection. Acta Materialia 48(1):359–369, DOI https://doi.org/10.1016/S1359-6454(99)00304-3, URL http://www.sciencedirect.com/science/article/pii/S1359645499003043

Aute V, Saleh K, Abdelaziz O, Azarm S, Radermacher R (2013) Cross-validation based single response adaptive design of experiments for Kriging metamodeling of deterministic computer simulations. Structural and Multidisciplinary Optimization 48(3):581–605, DOI 10.1007/s00158-013-0918-5

Baykasoglu A, Oztas A, Ozbay E (2009) Prediction and multi-objective optimization of high-strength concrete parameters via soft computing approaches. Expert Systems with Applications 36(3 PART 2):6145–6155, DOI 10.1016/j.eswa.2008.07.017

Beck J, Guillas S (2016) Sequential design with mutual information for computer experiments (MICE): Emulation of a tsunami model. SIAM-ASA Journal on Uncertainty Quantification 4(1):739–766, DOI 10.1137/140989613, 1410.0215

Berguin SH, Mavris DN (2015) Dimensionality Reduction Using Principal Component Analysis Applied to the Gradient. AIAA Journal 53(4):1078–1090, DOI 10.2514/1.J053372, URL https://doi.org/10.2514/1.J053372

Boursier Niutta C, Wehrle EJ, Duddeck F, Belingardi G (2018) Surrogate modeling in design optimization of structures with discontinuous responses: A new approach for ill-posed problems in crashworthiness design. Structural and Multidisciplinary Optimization

57(5):1857–1869, DOI 10.1007/s00158-018-1958-7

Breard G (2017) Evaluating self-organizing map quality measures as convergence criteria p 68, URL http://digitalcommons.uri.edu/theses/1033

Chen W, Allen JK, Schrage DP, Mistree F (1997) Statistical experimentation methods for achieving affordable concurrent systems design. AIAA Journal 35(5):893–900, DOI 10.2514/2.7464

Chowdhury R, Rao BN, Prasad AM (2009) High-dimensional model representation for structural reliability analysis. Communications in Numerical Methods in Engineering 25(4):301–337, DOI 10.1002/cnm.1118, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.1118

Chu XZ, Gao L, Qiu HB, Li WD, Shao XY (2010) An expert system using rough sets theory and self-organizing maps to design space exploration of complex products. Expert Systems with Applications 37(11):7364–7372, DOI 10.1016/j.eswa.2010.04.029, URL http://dx.doi.org/10.1016/j.eswa.2010.04.029

Constantine PG, Dow E, Wang Q (2014) Active Subspace Methods in Theory and Practice: Applications to Kriging Surfaces. SIAM Journal on Scientific Computing 36(4):A1500–A1524, DOI 10.1137/130916138, URL http://dx.doi.org/10.1137/130916138

Curtis SK, Mattson CA, Hancock BJ, Lewis PK (2013) Divergent exploration in design with a dynamic multiobjective optimization formulation. Structural and Multidisciplinary Optimization 47(5):645–657, DOI 10.1007/s00158-012-0855-8

Deschrijver D, Crombecq K, Nguyen HM, Dhaene T (2011) Adaptive Sampling Algorithm for Macromodeling of Parameterized \$S\$ -Parameter Responses. IEEE Transactions on Microwave Theory and Techniques 59(1):39–45, DOI 10.1109/TMTT.2010.2090407

Diez M, Campana EF, Stern F (2015) Design-space dimensionality reduction in shape optimization by Karhunen–Loève expansion. Computer Methods in Applied Mechanics and Engineering 283:1525–1544, DOI 10.1016/J.CMA.2014.10.042, URL https://www.sciencedirect.com/science/article/pii/S0045782514004125

Douak F, Melgani F, Alajlan N, Pasolli E, Bazi Y, Benoudjit N (2012) Active learning for spectroscopic data regression. Journal of Chemometrics 26(7):374–383, DOI 10.1002/cem.2443, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cem.2443

Estevez PJZP P (2012) DOI 10.1007/978-3-642-35230-0, URL https://www.springer.com/in/book/9783642352294

Forrester AIJ, Sóbester A, Keane AJ (2008) Engineering Design via Surrogate Modelling. DOI 10.1002/9780470770801

Gan N, Gu J (2019) Hybrid meta-model-based design space exploration method for expensive problems. Structural and Multidisciplinary Optimization 59(3):907–917, DOI 10.1007/s00158-018-2109-x

Ganapathy H, Ramu P, Muthuganapathy R (2015) Alpha shape based design space decomposition for island failure regions in reliability based design. Structural and Multidisciplinary Optimization 52(1):121–136, DOI 10.1007/s00158-014-1224-6

Ibrahim A, Rahnamayan S, Martin MV, Deb K (2016) 3D-RadVis: Visualization of Pareto front in many-objective optimization. 2016 IEEE Congress on Evolutionary Computation, CEC 2016 2016(July):736–745, DOI 10.1109/CEC.2016.7743865

Ito K, Dhaene T (2013) Self-Organizing Map Based Adaptive Sampling SELF-ORGANIZING MAP BASED ADAPTIVE SAMPLING (November 2014)

Jiang P, Shu L, Zhou Q, Zhou H, Shao X, Xu J (2015) A novel sequential exploration-exploitation sampling strategy for global metamodeling. IFAC-PapersOnLine 48(28):532–537, DOI 10.1016/j.ifacol.2015.12.183, URL http://dx.doi.org/10.1016/j.ifacol.2015.12.183

Jin Y, Li J, Du W, Qian F (2016) Adaptive Sampling for Surrogate Modelling with Artificial Neural Network and its Application in an Industrial Cracking Furnace. The Canadian Journal of Chemical Engineering 94(2):262–272, DOI 10.1002/cjce.22384, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cjce.22384

Kiviluoto K (1996) Topology preservation in self-organizing maps. IEEE International Conference on Neural Networks - Conference Proceedings 1:294–299

Koch PN, Evans JP, Powell D (2002) Interdigitation for effective design space exploration using iSIGHT. Structural and Multidisciplinary Optimization 23(2):111–126, DOI 10.1007/s00158-002-0171-9

Kohonen T (1997) Exploration of very large databases by self-organizing maps. IEEE International Conference on Neural Networks - Conference Proceedings 1, DOI 10.1109/ICNN.1997.611622

Lim CH, Yoon YS, Kim JH (2004) Genetic algorithm in mix proportioning of high-performance concrete. Cement and Concrete Research 34(3):409–420, DOI 10.1016/j.cemconres.2003.08.018

Liu H, Ong YS, Cai J (2018) A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design. Structural and Multidisciplinary Optimization 57(1):393–416, DOI 10.1007/s00158-017-1739-8

Lopez-Rubio E (2013) Improving the quality of self-organizing maps by self-intersection avoidance. IEEE Transactions on Neural Networks and Learning Systems 24(8):1253–1265, DOI 10.1109/TNNLS.2013.2254127

Mackman TJ, Allen CB (2010) Investigation of an adaptive sampling method for data interpolation using radial basis functions. International Journal for Numerical Methods in Engineering 83(7):915–938, DOI 10.1002/nme.2885, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.2885

Missoum S, Ramu P, Haftka RT (2007) A convex hull approach for the reliability-based design optimization of nonlinear transient dynamic problems. Computer Methods in Applied Mechanics and Engineering 196(29-30):2895–2906, DOI 10.1016/j.cma.2006.12.008

Obayashi S, Sasaki D (2003) Visualization and data mining of Pareto solutions using Self-Organizing Map. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 2632:796–809

Parashar S, Pediroda V, Poloni C (2008) Self Organizing Maps (SOM) for Design Selection in Robust Multi-Objective Design of Aerofoil. In: 46th AIAA Aerospace Sciences Meeting and Exhibit, Aerospace Sciences Meetings, American Institute of Aeronautics and Astronautics, DOI doi:10.2514/6.2008-914, URL https://doi.org/10.2514/6.2008-914

Qian Z, Seepersad CC, Joseph VR, Allen JK, Wu CF (2006) Building surrogate models based on detailed and approximate simulations. Journal of Mechanical Design, Transactions of the ASME 128(4):668–677, DOI 10.1115/1.2179459

Qiu H, Xu Y, Gao L, Li X, Chi L (2016) Multi-stage design space reduction and metamodeling optimization method based on self-organizing maps and fuzzy clustering. Expert Systems With Applications 46:180–195, DOI 10.1016/j.eswa.2015.10.033, URL http://dx.doi.org/10.1016/j.eswa.2015.10.033

Rosenbaum B, Schulz V (2012) Comparing sampling strategies for aerodynamic Kriging surrogate models. ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik 92(11-12):852–868, DOI 10.1002/zamm.201100112, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/zamm.201100112

Rumpfkeil M, Yamazaki W, Dimitri M (2012) A Dynamic Sampling Method for Kriging and Cokriging Surrogate Models. DOI 10.2514/6.2011-883, URL https://arc.aiaa.org/doi/abs/10.2514/6.2011-883

Seepersad CC, Dempsey BM, Allen JK, Mistree F, McDowell DL (2004) Design of Multifunctional Honeycomb Materials. AIAA Journal 42(5):1025–1033, DOI 10.2514/1.9594, URL https://doi.org/10.2514/1.9594

Seung HS, Opper M, Sompolinsky H (1992) Query by Committee. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Association for Computing Machinery, New York, NY, USA, COLT '92, pp 287–294, DOI 10.1145/130385.130417, URL https://doi.org/10.1145/130385.130417

Shan S, Wang GG (2004) Space exploration and global optimization for computationally intensive design problems: A rough set based approach. Structural and Multidisciplinary Optimization 28(6):427–441, DOI 10.1007/s00158-004-0448-2

Song L, Guo Z, Li J, Feng Z (2018) Optimization and knowledge discovery of a three-dimensional parameterized vane with nonaxisymmetric endwall. Journal of Propulsion and Power 34(1):234–246, DOI 10.2514/1.B36014

Torkkola K, Gardner R, Kaysser-Kranich T, Ma C (2000) Exploratory Analysis of Gene Expression Data Using Self-Organizing Maps. Proceedings of the Joint Conference on Information Sciences 5(2):782–785

Tseng HH, Wang SW, Chen JY, Liu CNJ (2014) A novel design space reduction method for efficient simulation-based optimization. Proceedings - IEEE International Symposium on Circuits and Systems (1):381–384, DOI 10.1109/ISCAS.2014.6865145

Wang GG, Shan S (2004) Design space reduction for multi-objective optimization and robust design optimization problems. SAE Technical Papers DOI 10.4271/2004-01-0240

Wang GG, Simpson T (2004) Fuzzy clustering based hierarchical metamodeling for design space reduction and optimization. Engineering Optimization 36(3):313–335, DOI 10.1080/03052150310001639911, URL https://doi.org/10.1080/03052150310001639911