8th World Congress on Structural and Multidisciplinary Optimization
June 1-5, 2009, Lisbon, Portugal

Design Space Reduction in Optimization Using Generative Topographic Mapping

Asha Viswanath, Alexander I.J. Forrester, Andy J. Keane

Computational Engineering and Design Group, School of Engineering Sciences, University of Southampton, Highfield, Southampton, SO17 1BJ, UK, email:av3x07@soton.ac.uk

1. Abstract

Dimension reduction in design optimization is an extensively researched area. The need arises in design problems dealing with very high dimensions, which increase the computational burden of the design process because the sample space required for the design search varies exponentially with the dimensions. This work describes the application of a latent variable method called Generative Topographic Mapping (GTM) in dimension reduction of a data set by transformation into a low-dimensional latent space. The attraction it presents is that the variables are not removed, but only transformed and hence there is no risk of missing out on information relating to all the variables. The method has been tested on the Branin test function initially and then on an aircraft wing weight problem. Ongoing work involves finding a suitable update strategy for adding infill points to the trained GTM in order to converge to the global optimum effectively. Three update methods tested on GTM so far are discussed.

2. Keywords: dimension reduction, GTM, response-surface, global optimization, exploration, weighted lower bound.

3. Introduction

In the design industry, the use of high-fidelity simulation models for optimization places a high demand on the computational cost. Though the advent of new advances in computing has tended to reduce the cost, the complexities of design problems, i.e. considering non-linearity in the model, using more complex solvers, etc., have proportionally increased, demanding such computations to be faster and more reliable. Hence the need for approximate models (also called surrogate models, meta models or response surface models) for design optimization arises. These can act as cheap alternatives to the original model and reduce the computational burden, whilst still providing improved designs [1]. The idea behind surrogate modeling is to analyze a set of initial designs to generate data points, using which an approximate model is constructed to fit the objective function and constraints. Optimization is then conducted using the approximate model. When the design problem includes a large number of design variables, say > 50, building useful surrogate models may require vast quantities, of the order of thousands, of data points to sample the search space. For an accurate prediction of the surrogate model, if the sample density is n-locations for one-dimension, then for k-dimensions, n^k observations are required, making a design of experiment (DOE) sample a costly affair. This has been referred to as 'curse of dimensionality'(Forrester *et.al.* [10]). Often, in high-dimensional functions, not all the measured variables are equally relevant in understanding the underlying objective function. Though predictive models *can* be constructed with high accuracy from high-dimensional data, it is still desirable to reduce dimensionality and to find ways of expressing the objective function with fewer dimensions. Many methods have been tried in the past towards this goal, but none stands as being suitable for all design problems. Moreover,the common approach in most of the 'screening' methods is to identify the variables relevant for design problem and discard the rest by fixing them at constant values during the optimization. This may not always be an attractive feature since the relevance of the fixed variables may emerge later during design process. Hence, the need for a technique which takes into account the effect of all the variables, whilst still reducing dimensionality. Towards this end, latent variable models, which represent the probability distribution of high-dimensional data in a low-dimensional space of latent variables without removing any variable information, have been found to be useful. Among the considerable number of latent models listed in the literature, are principal component analysis (PCA)[11], factor analysis (FA)[2], probabilistic PCA (PPCA)[3], elastic nets[8] and Kohonen's Self Organizing Map (SOM)[12]. The model which drew most attention from researchers and engineers was Generative Topographic Mapping (GTM)[4]. GTM is, in principle, similar to FA and PPCA, but the significant difference is that GTM allows a non-linear relation between the latent set and training dataset. Moreover, as compared to SOM, the method has a rigorous

mathematical formulation. GTM is not a new approach in design optimization since it has been used as an effective visualization tool. This work, however, explores the possibilities of using GTM in reducing the design search space for global numerical optimization.
A sparse sample fails, in most cases, to find the global optimum unless effective update strategies have been applied. The update methods add infill points to the initial training sample set so that consecutive cycles of training lead to the global optimum of the function. In most cases, there are issues of the search getting stuck in a local optimum. This can be avoided by well-planned updates. In this work, we examine some update methods for GTM. The paper is divided into four sections: section 4 discusses the method of GTM applied to optimization, section 5 discusses the update strategies examined by the authors, section 6 shows results of the method applied to test problems and section 7 discusses the scope of ongoing research.

4. GTM for optimization

The method of GTM models the probability density of N data points of high dimensional D–space, $\mathbf{T} = \{\mathbf{t_1}, \ldots, \mathbf{t_N}\} \forall \mathbf{t} = \{t_1, \ldots, t_D\}$, in terms of a grid of K latent variables $\mathbf{X} = \{\mathbf{x_1}, \ldots, \mathbf{x_K}\} \forall \mathbf{x} = \{x_1, \ldots, x_L\}$ of lower dimension L. The probability distribution in data space is assumed to be based on Gaussian mixture distributions having a variance β^{-1} and centres as a function $\mathbf{y}(\mathbf{x}, \mathbf{W})$, $\mathbf{W}$ being weights. $\mathbf{W}$ and β are the parameters to be determined through the maximization of the log-likelihood of the model using an E-M or other suitable training algorithm. The Gaussian centres $\mathbf{y}(\mathbf{x}, \mathbf{W})$,which are assumed to be a regression equation of the weights $\mathbf{W}$ and a grid of basis functions $\mathbf{\Phi}(\mathbf{x})$, map every point in latent space to a point in data space. The latent points are confined to an L–dimensional manifold non-linearly embedded in the D–dimensional data space. GTM uses Bayesian inference to provide posterior information about the latent points responsible for the data points based on the observation of maximized log-likelihood, for which the prior considered is that all the data points are accounted for by the same latent point. A detailed derivation and explanation of the GTM method is available in Bishop ([4]). Figure 1 shows how a 1D GTM manifold resides in the 2D countour space of the 2D Branin function $f(t_1, t_2)$, in which the dimension reduction was from 2D data space to 1D latent space. Each point on the manifold is coloured with the corresponding function value obtained from GTM training and they match, in most places, with the contour colours,i.e., the actual function value. The manifold points are hence encircled in white to be differentiated from the contour. Note that the training points shown by red crosses in the contour may not exactly lie on the manifold, they only determine the shape of the GTM manifold since GTM estimation is similar to a neural network training.
Our GTM-based optimization algorithm is shown in Figure 2. This method follows a response-surface based approach, starting with an experimental design (DOE) and the trained GTM being used as the surrogate. The benefits of GTM as a surrogate is that the linear combination of basis functions which it uses can be tuned and it has a statistical interpretation since it uses Bayesian inference for the training. Hence,we combine statistical analysis of data along with optimization methods. Data samples are generated by DOE and trained using GTM. The validation data-set was taken from the sample and the rest used for training. The optimization update search is performed in the latent space and the best point obtained is transformed to data space and augmented to the training data set. Iterations are continued until there is no further improvement in the design. This follows the 'two-stage' response surface based optimization approach([7]). It was found useful to obtain a set of best points rather than one, from the latent space as infill points to be appended to the training set. The DOE was also conducted for different sets of samples and the optimum obtained averaged.

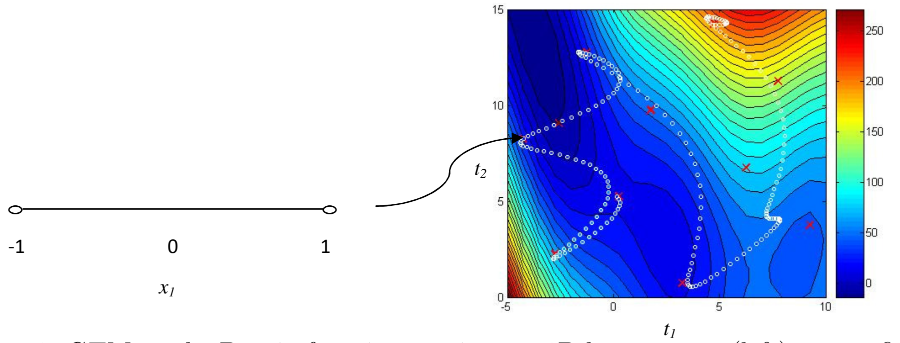


Figure 1: GTM on the Branin function – points on 1D latent space (left) are confined to a manifold lying in 2D data space (right).

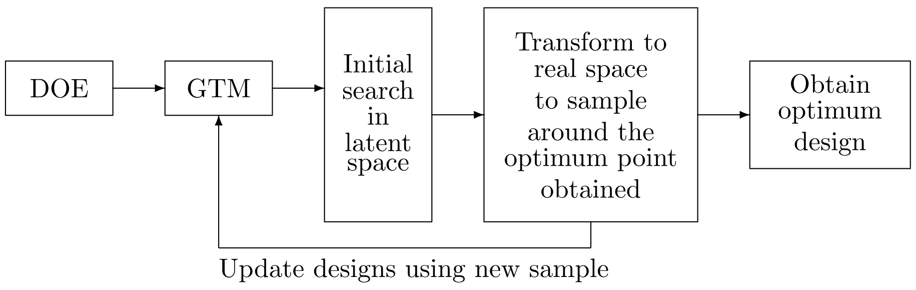

DOE
GTM
Initial search in latent space
Transform to real space to sample around the optimum point obtained
Obtain optimum design
Update designs using new sample

Figure 2: Algorithm of reduction process using GTM.

4.1. Selection of parameters

There are a number of hyper-parameters involved in the GTM method and an intelligent choice of these parameters is essential for the method to accurately model the distribution of the dataset. Some of them, like the basis functions to be used, the number of basis functions, the choice of prior over weights $\mathbf{W}$ during regularization and the number of latent points have already been investigated in the detailed work on GTM by Markus [13]. In this work, interest is focused on optimization of the objective function in the reduced latent space generated by GTM and hence we examine the parameters that may be crucial in getting a reasonable optimum.

The best number of the E-M cycles for training the GTM can be determined by plotting the error function of the model which is to be minimized, negative of the log-likelihood in this case, for a training data-set and a validation data-set against the number of cycles [3]. The validation data set can be either independant or taken from the training data itself in the case of limited available data. The number of cycles which give the least value of error function for the validation data is selected as the number of E-M cycles. Choosing more cycles may lead to over-fitting and less, in under-fitting the model.

The latent space is a uniformly spaced mesh of points. The choice of the number of these points ($meshsize^{latentspacedimension}$) determines the resolution of the model in the latent space. A denser mesh of points may be able to capture the distribution more accurately, but at the cost of computational time and burden. GTM has mainly been used for visualization so far and hence the latent space dimension does not normally exceed two, but in the case of dimension reduction, we wish to deal with latent space dimensions slightly more than two in order to be able to capture the optimum design effectively. Hence it would be useful to test different mesh sizes for different samples and different latent space dimensions to decide on the best choice. The metric used for deciding this criterion is the Root Mean Sqaure Error (RMSE). It is calculated with validation data-set as follows:

$$RMSE = \sqrt{\frac{\sum_i^{n_v}(y^i - \hat{y}^i)^2}{n_v}}, \tag{1}$$

where y is the function value calculated from the validation data set having n_v data points and $\hat{y}$ is the predicted function values retrieved from the GTM training. The number of latent points having least value of RMSE or which does not further reduce RMSE considerably is chosen.

4.2. GTM algorithm

The GTM procedure described above has been implemented in MATLAB as a GTM toolbox by Markus [14], which provides a detailed description of the steps involved and the program subroutines. Using these subroutines, the optimization has been implemented in the following steps.

Steps:

1. *Generate data sets.* The data set matrix ($\mathbf{T}$) contains normalized variables and the objective function and is of dimension $N \times (D+1)$. A space-filling Morris-Mitchell Latin hypercube [16] with maxi-min criterion is used for the DOE. The validation data-set is chosen randomly from the DOE and the rest of the data points used as the training set.

2. *GTM training.* As described in the literature of GTM, the set-up and training of GTM is performed using the toolbox subroutines. There were two changes made in this routine for the purpose of optimization. The width of the basis function, σ was considered constant throughout the training process for a data-set. This is now a hyper-parameter which has to be estimated apriori to GTM training. Since there was no specific method to determine σ, a range of values from $[2^{-2} - 2^2]$ is considered and for each value, the GTM trained and the value which gave the maximum log-likelihood while training is taken as the best value of σ [5]. Another improvement in training is the use of a genetic algorithm(GA) for further training the GTM after it has been trained using the E-M cycles. This was found to increase the likelihood further and hence make the GTM model the data-set more accurately.

3. *Search in latent space.* The latent space is searched for suitable update (as per update methods discussed in next section) points using a GA on latent variables $\mathbf{x}$ with bounds $[-1, 1]^L$ and the search function being the regression equation $\mathbf{y}(\mathbf{x}, \mathbf{W}) = \Phi(\mathbf{x})\mathbf{W}$ where $\mathbf{W}$ is the GTM trained weight matrix and Φ is the basis function matrix. This equation maps any latent point $\mathbf{x}$ to data point $\mathbf{y}$. Hence, similar to $\mathbf{t}$, $\mathbf{y}$ has D columns of data variables and a $D + 1^{th}$ column of function values, given by GTM training. The actual function value is then evaluated using the variables into $\mathbf{y}$ and this value used by the GA.

4. *Viewing the posterior means of the data set and getting back to real space.* The data points in $\mathbf{T}$ can be viewed in latent space by evaluating the posterior means as $\mathbf{x}_{mean} = \sum_k^K \mathbf{x}_k p(\mathbf{x}_k|\mathbf{t})$ [4]. These positions of data points in latent space $\{\mathbf{x}_{mean1}, \ldots \mathbf{x}_{meanN}\}$ can be substituted in Φ in the regression equation to get their corrsponding values in data space.

5. *Updating the training set.* The update point is added to the training data-set and the above steps from step 2 continued for a set number of iterations.

5. Update strategies for GTM

The optimum obtained from the trained GTM of the initial training data set will not necessarily be the global optimum of the design problem. It is essential to add more points to the data set so that the subsequent GTM training has more information about the data to model it better. Adding these infill points to the data set, called updating, must be in such a way that the global optimum is reached with the minimum number of additions/iterations. There are many update methods available in the literature of surrogate models, e.g., prediction based exploitation[10], error based exploration[10], the expected improvement method[15], a weighted statistical lower bound method[9] which balances prediction based exploitation and error based exploration by some weights, conditional likelihood approaches[10] and so on. As is the case for standard surrogate model based optimization, the most efficient method is unclear. The following sections describe two of the above methods and an additional exploration method implemented here for GTM.

5.1. Exploitation

This is the most common update strategy followed in surrogate modelling. The infill point added at each iteration is the minimum of prediction $\hat{f}$ obtained from each re-trained GTM. The minimum of the prediction is found by a GA search (Step 3 of section 4.2). This prediction based exploitation does not in some cases converge to the global minimum, indeed sometimes not even to a local minimum. Hence there is a need to introduce an effective algorithm which would lead to the global minimum, coupled with the exploitation method. For this purpose, an exploration method was tested.

5.2. Exploitation with exploration

The design space can be explored beyond the GTM manifold to see if any optimal points have been left untouched by the GTM manifold. Prediction based search methods are confined to the manifold created by the GTM. Hence if the manifold fails to pass through global basin of attraction, then the optimum may lie in an unexplored region. To avoid this, a slight amount of noise is supplied to the manifold so as to allow the search to pass through a neighbourhood of the manifold rather than strictly on the manifold. But this did not improve the solution and how much noise to be added for a given data-set was uncertain. We found that a technique is required to search other parts of the design space away from the manifold. Towards this end, a small number of updates are performed using the exploitation method and the GTM trained each time is stored. Then, points away from all the GTM manifolds created so far are found by an exploration method and are used as the next infill points. This method follows the principles of the dynamic hill climbing method [6], in which, after a few searches in the design space, if the search gets stuck in some local optimum, a point far away from the previous point is used as the next update so as to start a new search in unexplored regions of the space. In the case of GTM, the real space is searched for regions where the manifolds haven't passed through and the centroid of such clusters of points is found using a k-means algorithm. These centroids are taken as the next updates. This method was found to be useful to make the manifold take a different path altogether and hence to visit unexplored areas of the design space. After an exploration, more cycles of exploitation updates are performed.

5.3. Weighted Lower bound

The weighted lower bound(WLB) method provides a balance between a local and global search. The predictor and some space-filling measure are combined using a weighing factor as

$$WLB(t) = w\hat{f}(\mathbf{t}) - (1-w)s(\mathbf{t}) \tag{2}$$

The prediction $\hat{f}$ is obtained from the GTM , w is a weight metric and s is a space filling metric which is calculated as the product of the range of observed function values and the Euclidean distance of the prediction point to the nearest training data point. Here, w=1 results in a local search using the prediction based exploitation and w=0 gives a space-filling global search analogous to error based exploration. The WLB is then optimized using a GA (Step 3 in Section 4.2).

6. Illustrative examples and Results

The test functions used to implement the method descibed above are the 2D Branin function, for visualization of the working of GTM method and the different update strategies, and an engineering application problem of an aircraft wing weight function with 10 independent variables. These are next discussed in detail with figures illustrating the results of GTM on each of them.

6.1. Branin function

The modified form of the Branin function[10], having just one global optimum instead of three (as in the ordinary Branin function), is given by

$$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right) + 10\left[\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 1\right] + 5x_1, x_1\epsilon[-5,10], x_2\epsilon[0,15] \tag{3}$$

The function contours and the trained GTM manifold are shown for five exploitation updates in Figure 4 (a)-(e). The crosses in the figure represents the training points along with update points. The GTM manifold is visibly spreading out over the contours. It is coloured according to the function values that the GTM training gives and the update points are found on this manifold for each iteration. To search in the neighbourhood of the manifold, adding a Gaussian noise ranging from magnitude of 0.01(very close to manifold) to 0.1(away from manifold) was tested, but none of the noise values gave a better solution than the optimum value obtained on the manifold. So the exploration method was tested to change the shape of manifold and the manifold was indeed found to follow a different path covering previously unexplored regions of the design space. The selection of ten exploration updates is shown in Figure 4(f). Five exploitation updates after adding the exploration points are shown in Figure 5 (a)-(e). These plots were for one DOE of the Branin function. The method was repeated over fifty different DOE's and a global optimum of -16.12 was obtained. Figure 5 (f) shows the averaged runs. The WLB method was tested for different values of the weight metric w and the observation was plotted as the average function value obtained after ten runs against w, shown in Figure 3. The best value for w was found to be 0.9. This value could differ based on the problem.

6.2. Aircraft wing weight function

The wing weight function* is given by

$$W = 0.036 S_w^{0.758} W_{fw}^{0.0035} (\frac{A}{\cos^2 \Lambda})^{0.6} q^{0.006} \lambda^{0.04} (\frac{100 t_c}{\cos \Lambda})^{-0.3} (N_z W_{dg})^{0.49} + S_w W_p. \tag{4}$$

where S_w - wing area (ft^2), W_{fw} - weight of fuel in the wing (lb), A - aspect ratio, Λ - quarter-chord sweep (deg), q - dynamic pressure at cruise (lb/ft^2),λ - taper ratio, t_c - aerofoil thickness to chord ratio,

*formula adapted from Raymer's aircraft conceptual design [17]

N_z - ultimate load factor (1.5 x limit load factor), W_{dg} - flight design gross weight (lb), W_p - paint weight (lb/ft^2).
For this function, the variables N_z, W_{dg}, A, S_w and t_c are more dominant than the rest of the variables. GTM also gave similar results when the 2D latent space plots of the variables and the function were examined in Figure 6 (a). While N_z, W_{dg}, A, S_w show direct similarity in pattern with the function, t_c shows a close inverse relation – which is evident from Eq.4. The number of E-M cycles required to train the GTM was found to be 12 and the mesh size of latent space for each dimension were found from plots of RMSE against varying mesh sizes. The contour plot of RMSE with varying latent space dimension(L) and varying sample sizes(N) in Figure 6 (b) shows that RMSE decreases with increasing sample sizes and increasing dimension, as expected. Statistical t-tests were carried out to determine the most appropriate value of (N), which was found as 110. Optimization was performed in the latent space using 1D, 2D, 3D and 4D GTM, each averaged over 115 runs and the function values are plotted against sample size in Figure 6 (c)-(f). The exploitation coupled with exploration updates and updates using WLB were then tested on the function. Table 1 shows the values of number of latent space points ($K = meshsize^L$) and the number of basis function (M) used for each L, the optimum averaged over 115 DOEs for the different update schemes and the time taken per DOE for the simple exploitation scheme. The simple exploitation scheme used 100-point DOE with 10 exploitation updates. The exploitation with exploration scheme used 70-point DOE with 30 exploration updates and 10 exploitation updates, 5 each before and after exploration. In the WLB update scheme, w=0.9 and w=1.0 (simple exploitation itself) was found to perform better, indicating that the predictor cannot be improved by further exploration of the GTM manifold. But neither of the update strategies gave the global optimum of 124.9. The most probable reason for this is that the dimension reduction is not accurate in the case of 2D and 3D and for 4D, the hyperparameters can be tuned better to improve the solution, but at the cost of computational time. The amount of time taken for a 4D GTM for higher values of M and K was considerable.

Table 1: Comparison of GTM for different L

K	M	L	Averaged optimum wing weight (lb)			Time taken/DOE (sec) in simple exploitation
			with simple exploitation	with exploration	with WLB,w=0.9	
240	110	1	159.42	161.13	160.01	31.9
20^2	9^2	2	157.2	155.49	156.86	47.9
15^3	7^3	3	146.98	151.15	152.16	287.4
10^4	4^4	4	151.2	151.1	156.26	750.4

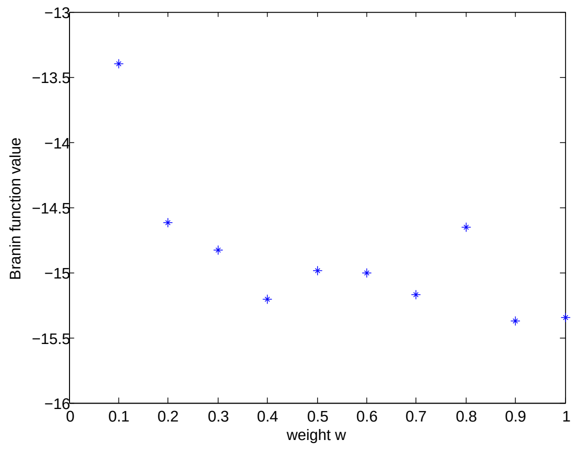
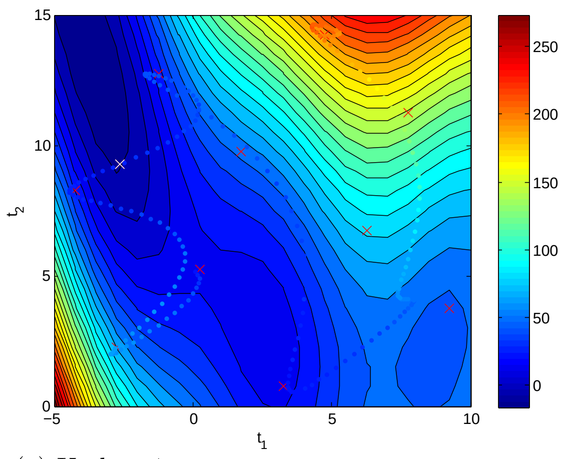
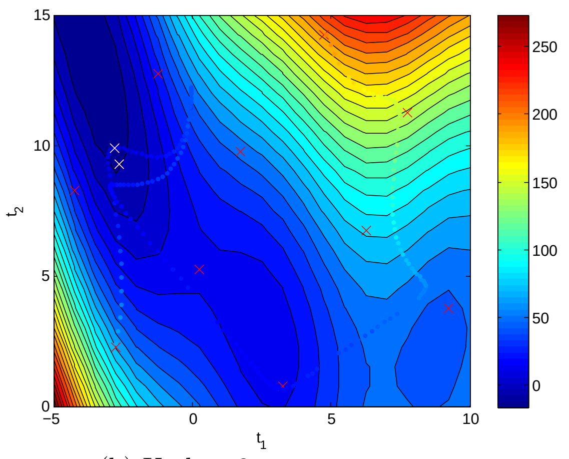


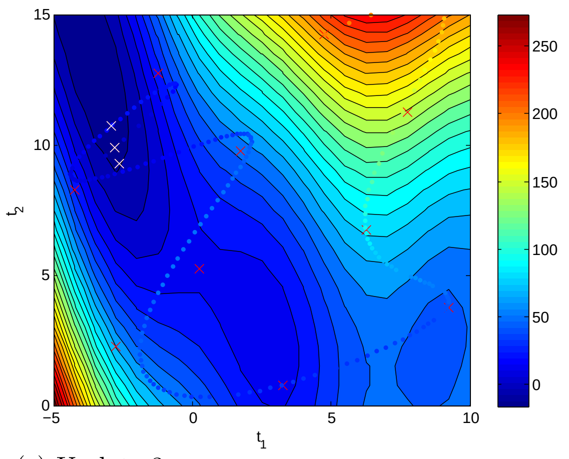
Figure 3: Weighted lower bound update on Branin function



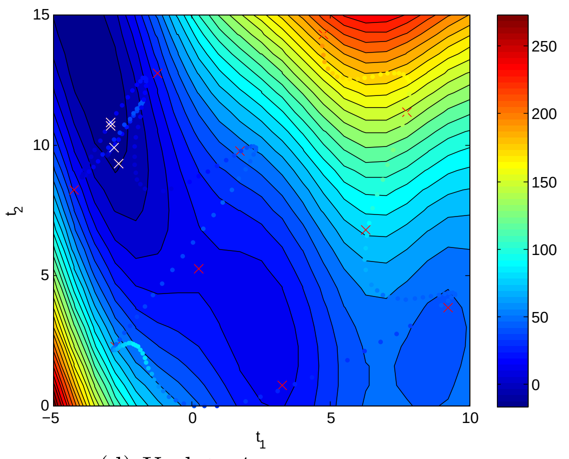
(a) Update 1



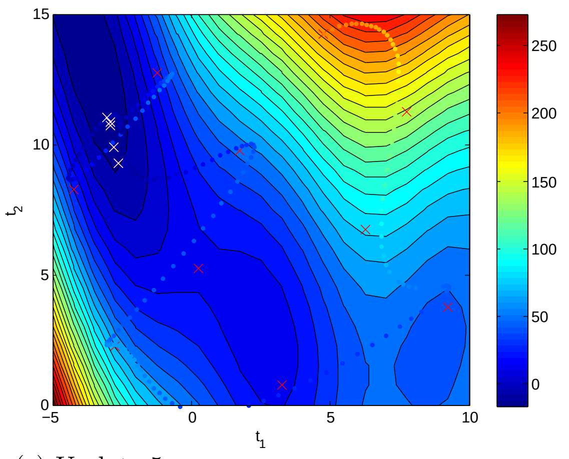
(b) Update 2



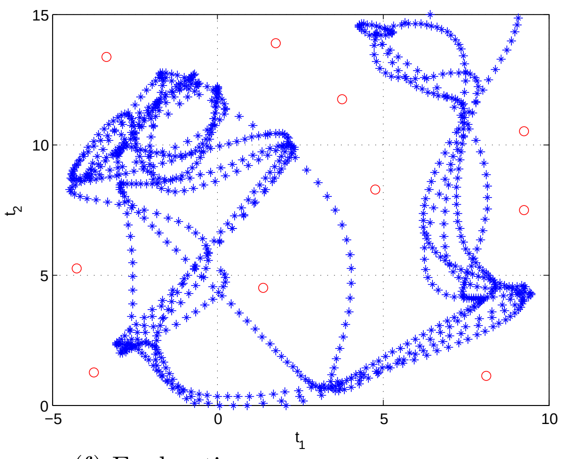
(c) Update 3



(d) Update 4

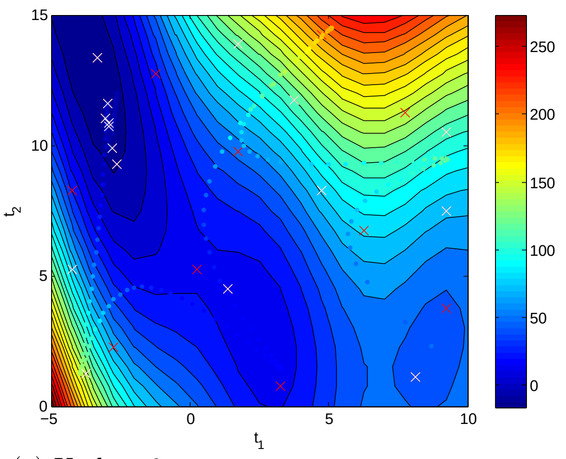


(e) Update 5

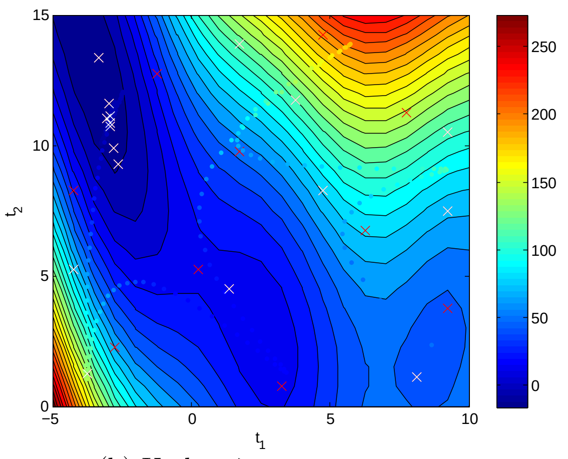


(f) Exploration

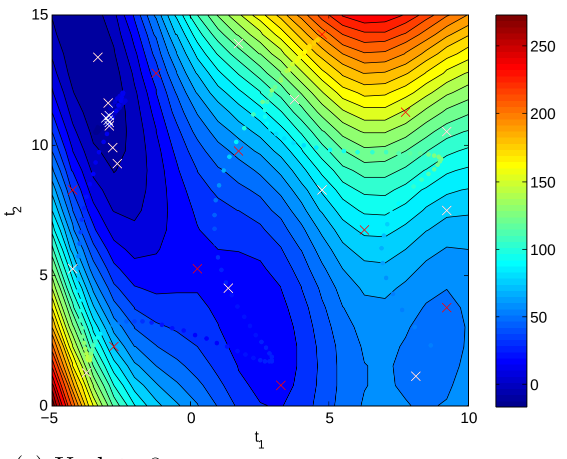
Figure 4: GTM manifolds for Branin function. The white crosses show update points and red crosses are training points. (a-e) Initial 5 exploitation updates, (f) exploration with k-means algorithm, the red circles are exploration update points.



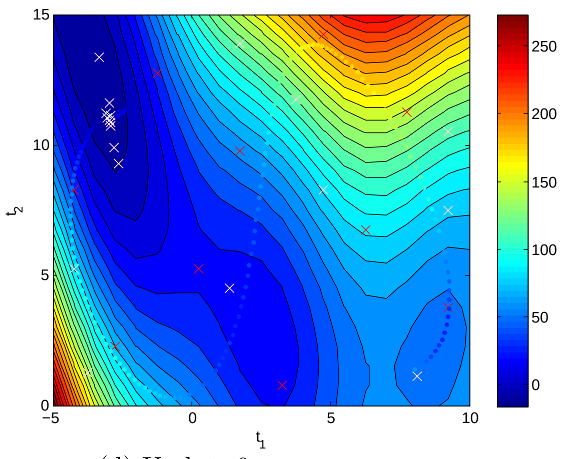
(a) Update 6



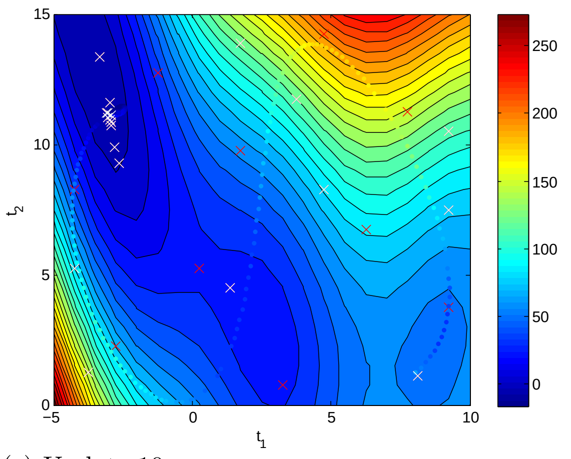
(b) Update 7



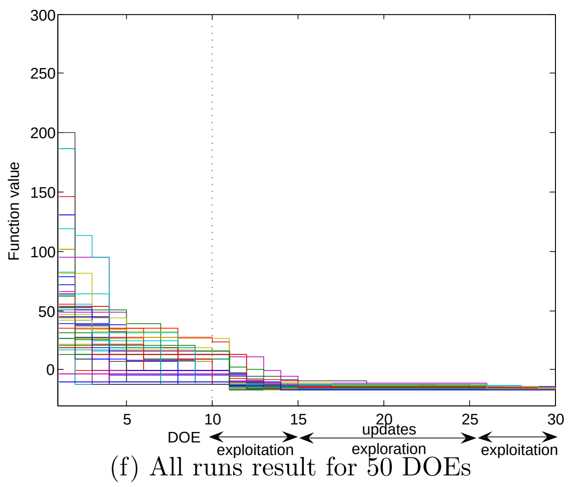
(c) Update 8



(d) Update 9

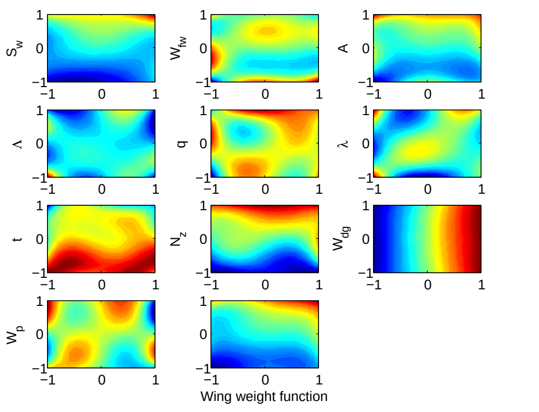


(e) Update 10

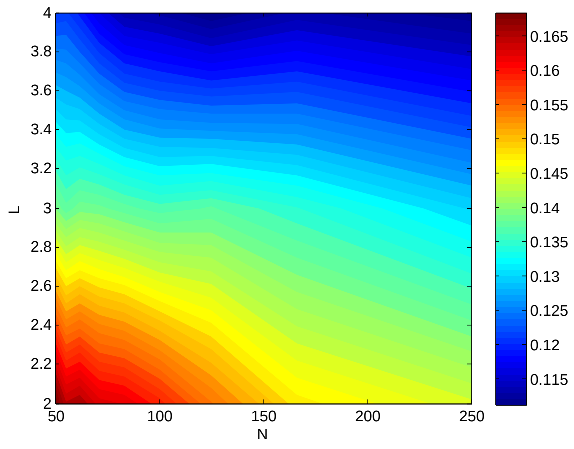


(f) All runs result for 50 DOEs

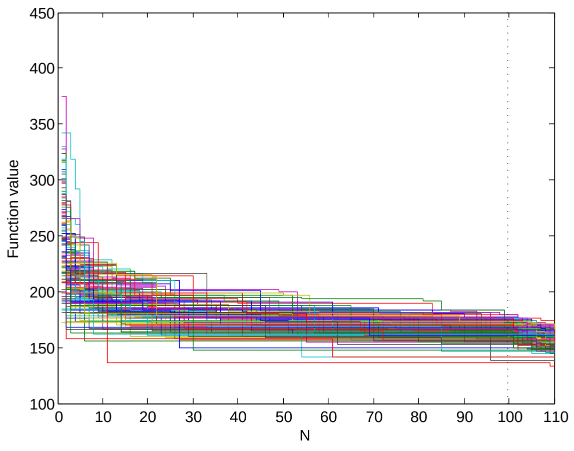
Figure 5: GTM manifolds for Branin function.(a-e) 5 more exploitation updates after adding exploration points, (f) All runs of 50 DOEs with 5 exploitation updates, 10 exploration points and 5 more exploitation updates, averaged optimum function value = -16.12



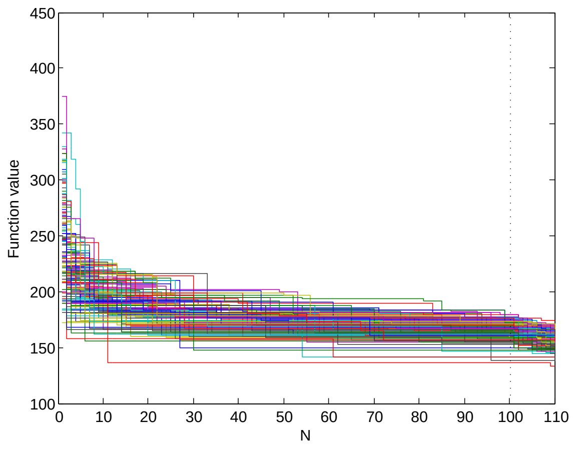
(a) Plot of variables and function in 2D latent space



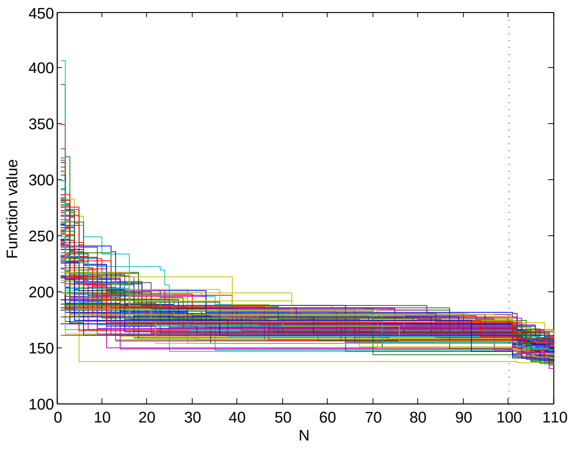
(b) RMSE contour for N and L



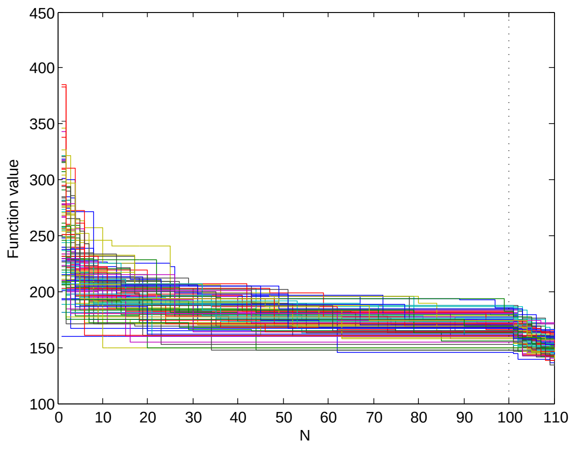
(c) 1D GTM



(d) 2D GTM



(e) 3D GTM



(f) 4D GTM

Figure 6: Optimization using GTM on the wing weight function

7. Future scope of research

This paper discusses the application of GTM in numerical optimization. The method developed using GTM followed a response surface based algorithm. The hyper-parameters of the method have to be carefully chosen for each problem as the GTM training was found to be sensitive to their selection. It was observed that GTM could give optimal results even with a small sample size if the dimension reduction is effective and the update method robust. In this work, three update methods were tested, but they were not found robust in the case of a ten dimensional function and hence, there is a need to seek out improvements in the schemes. Also the latent space dimension,L, which would be most efficient for a particular problem needs to be found with some care, in which case L can also be considered a parameter to be determined prior to GTM training. The aim is to find the least latent space dimension (best dimension reduction possible) for accurate prediction of the global optimum (best optimizer) with least amount of function evaluations (least computational effort). GTM certainly seems a promising step towards this goal.

8. Acknowledgements

This work was supported by Rolls-Royce Plc. and the UK DTI.

9. References

[1] Andy J. Keane and Prasanth B. Nair . *Computational approaches for aerospace design.* Wiley and Sons, 2000.

[2] D. J. Bartholomew. *Latent Variable Models and Factor Analysis.*Charles Griffin and Co. Ltd, London,1987.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[4] C. M. Bishop, M. Svensén, C.K.I. Williams, GTM: The Generative Topographic Mapping, *Neural Computation*, 10, 215–234, 1998.

[5] C. M. Bishop, M. Svensén, and C. K. I. Williams. Developments of the GTM Algorithm. *Neurocomputing*, **21**,203–224 , 1998.

[6] Deniz Yuret and Michael De La Maza. Dynamic hill climbing: Overcoming the limitations of optimization techniques *In The Second Turkish Symposium on Artificial Intelligence and Neural Networks* **2**, 208–212, 1993.

[7] Donald R. Jones, A Taxonomy of Global Optimization Methods Based on Response Surfaces, *Journal of Global Optimization*, 21, 345–383, 2001.

[8] R. Durbin and D. Willshaw. An analogue approach to the travelling salesman problem. *Nature*, **326**, 689–691, 1987.

[9] A. I. J. Forrester. Efficient global aerodynamic optimization using expensive computational fluid dynamics simulations. *Ph.D. Thesis, University of Southampton*, 2004.

[10] A. I. J. Forrester,A. Sóbester and A. J. Keane, *Surrogate Models in Engineering Design: A practical guide*,Wiley and Sons, 2008.

[11] I. T. Joliffe . *Principal Component Analysis.* Springer- Verlag, New York, 1986.

[12] T. Kohonen. *SelfOrganizing Maps.* SpringerVerlag, Berlin, 1995.

[13] Markus Svensén. GTM: The Generative Topographic Mapping. *Ph.D. Thesis, Aston University*, 1998.

[14] Markus Svensén. *The GTM Toolbox - User's guide*, Version 1.01, October 1999.

[15] Matthias Schonlau. Computer experiments and global optimization. *Ph.D. Thesis, University of Waterloo*, 1997.

[16] Max D. Morris . Factorial sampling plans for preliminary computational experiments. *Technometrics*, **33**, 161–174, 1991.

[17] D. P. Raymer, *Aircraft Design:A Conceptual Approach*, Education Series,4th Ed.,AIAA, Washington, DC, 2006.