AIAA JOURNAL
Vol. 49, No. 6, June 2011

Dimension Reduction for Aerodynamic Design Optimization

Asha Viswanath,* A. I. J. Forrester,† and A. J. Keane‡
University of Southampton, Southampton, England SO17 1BJ, United Kingdom

DOI: 10.2514/1.J050717

The search for an optimal design in a high-dimensional design space of a multivariate problem requires a sample size proportional or even exponential to the number of variables of the problem. This curse of dimensionality places a computational burden on the cost of optimization, especially when the problem uses expensive high-fidelity simulations and may force one to try to reduce the dimensions of a problem. Traditional variable screening techniques reduce the dimensionality of the problem by removing variables that seem irrelevant to the design problem. This practice fails when all the variables are equally relevant in the problem or when some variables are relevant only in some parts of the design space. The present work describes a dimension reduction method called generative topographic mapping based on nonlinear latent variable models that transform a high-dimensional data set into a low-dimensional latent space, without removing any variables. It is first illustrated on a two-dimensional Branin function and then applied to a 30-dimensional airfoil problem. The method is then compared with a global optimizer (a genetic algorithm), other dimension reduction methods (principle component analysis and Gaussian process latent variable models), and Kriging surrogate models. The method improves when the initial sample used for dimension reduction is filtered to contain only good designs.

Nomenclature

C_D	=	drag coefficient
C_L	=	lift coefficient
D	=	dimension of training data set
K	=	number of latent data points
$\mathbf{k}$	=	kernel matrix for Gaussian process latent variable model
L	=	dimension of latent space
N	=	number of training data points
$\mathbf{T}$	=	training data set
$\mathbf{W}$	=	matrix of weights
$\mathbf{X}$	=	latent data set
$\mathbf{Y}$	=	prediction data set
$\hat{\mathbf{y}}$	=	prediction from model
β	=	inverse of variance of Gaussian mixture distribution
μ	=	vector of basis function centers
σ	=	width of basis function
$\mathbf{\Phi}$	=	matrix of basis function activations

I. Introduction

DESIGN optimization problems involving more than 20 variables and using high-fidelity simulation models for objective function and constraint evaluation often present a computational challenge. This is because the sample size required to search the design space for the optimum is often exponential to the number of variables (dimensions) of the problem [1]. For example, consider a minimization problem of an aircraft wing weight function that involves only 11 design variables to completely describe the wing (surface area, aspect ratio, and so on). The design variables are subject to some bounds, and there are infinite values each variable can take. We consider a range of 10 values for each of the variables, so that for 11 variables, we have a design of experiment (DOE) of 10^{11} values. More sample points means a greater number of expensive simulations. The use of surrogate models (also referred to as response surface models or metamodels) [2] in such problems, as cheap alternatives to the original model, has helped to reduce the cost of such optimization. The idea behind surrogate modeling is to analyze the initial design using full fidelity software to generate data points, from which an approximate model is constructed to fit the objective function and constraints. Optimization is then applied to the approximate model, identifying new designs (data points to be evaluated), and the process continues until the best designs are obtained. Kriging surrogate models, which use stochastic processes to model the problem functions, are especially good at modeling the nonlinear, multimodal functions occurring in engineering [3]. However, the method of Kriging cannot construct effective surrogates with a small number of initial designs. Although recent studies have shown a sample size of 10 times the dimension of the problem to be an effective choice for the construction of surrogates [4], a high-dimensional problem is still plagued by the curse of dimensionality, which motivates us to study methods to reduce the dimensions of a problem.

Often, in high-dimensional data sets, there may be either some correlations or nonlinear relations between variables, and not all the variables will be required to describe the objective function. For example, in the three-dimensional (3-D) Swiss roll [5], the true intrinsic dimensionality of two is due to the nonlinear relation between points on the roll. Criteria for defining the most important factors (variables), referred to as factor screening (also known as feature selection in statistics), have been well researched due to the growing needs of industry. The fractional and full factorial designs were the first to be used [2] with a two-level experiment requiring 2^D points, D being the number of variables. There are many other designs, like the Plankett–Burman designs [2], edge designs [6], sequential bifurcation [7], iterated fractional factorial designs [8], Iman and Conover's [9] sensitivity analysis using a rank-regression model, and Morris's one-at a-time design [10], that can be found in the screening literature, but all these designs have limitations on the number of dimensions of the problem that can be used. Moreover, the common approach in most screening methods is to identify the variables most relevant for design problems and discard the remaining variables by fixing them at constant values during any optimization. This may not always be an attractive feature, since the relevance of the fixed variables may emerge later during the design process. Hence, the need is for a dimension reduction (DR) method

Received 11 June 2010; revision received 7 January 2011; accepted for publication 15 January 2011.

*Research Student, Highfield, Computational Engineering and Design Group, School of Engineering Sciences; av3x07@soton.ac.uk.

†Lecturer, Highfield, Computational Engineering and Design Group, School of Engineering Sciences. Member AIAA.

‡Professor, Highfield, Computational Engineering and Design Group, School of Engineering Sciences. Member AIAA.

that can capture underlying patterns in the variables in a reduced space without removing any variables from the design, thereby enabling optimization to be performed in the reduced space.

In the broad field of statistics, many methods are used in data analysis to capture the important relation between variables in the data and then project it to a lower-dimensional space. Fodor [11] and van der Maaten et al. [12] provide surveys of the major DR techniques. The most well-known linear method is principal component analysis (PCA) [13], which finds the principal components as the new basis vectors for the data, ordered by their corresponding variances, with the vectors with the largest variance corresponding to the first principal component. Other DR methods in the literature include principle curves and surfaces [14,15], kernel PCA [16], probabilistic nonlinear PCA/Gaussian process latent variable models (GPLVMs), elastic nets [17], density networks [18,19], self-organizing maps (SOMs) [20], generative topographic mapping (GTM), curvilinear distance analysis (CDA) [21], multidimensional scaling (MDS) [22], Isomap (combination of CDA and MDS) [5], autoencoders [23], locally linear embedding (LLE) [24], partial least-squares regression [25], manifold sculpting [26], Laplacian eigenmaps [27], Hessian LLE [28], local tangent space alignment [29], maximum variance unfolding [30], and so on.

From this vast gamut of methods, the method suited for design optimization is one that is capable of learning, from a given DOE of the design problem, the internal model of the data and map that to a low-dimensional space such that any point on the low-dimensional space can also predict useful data not given in the DOE. So, an inverse mapping of latent points to a real data set must be possible, since the optimum obtained by searching the low-dimensional space has to be transformed to a real data point for the subsequent objective function evaluation. For this purpose, only parametric methods can be used, since they map the high-dimensional data to low-dimensional space using parameters in the model, which are estimated during the tuning process of the model to the data. Using these parameters, any low-dimensional data point can be inverse mapped to high-dimensional data point, thus allowing a design search for the optimum, possible in the low-dimensional space instead of the original data space. Nonparametric methods learn the local structure of the data by study of neighborhood graphs showing connections of points to their k-nearest neighbors trying to preserve this structure in the low-dimensional space, and hence cannot be inversely mapped. Also, most of the design problems have nonlinearity in data and, to appropriately model this, a nonlinear DR method has to be chosen, as linear methods may not always capture the nonlinear relations of data points effectively. Of the methods noted earlier, only nonlinear PCA methods and methods like SOM, GTM, and autoencoders are parametric methods, and they are thus useful for DR in design optimization. In this work, we focus on the nonlinear latent variable model (LVM) of GTM [31], a probabilistic formulation of SOM, and hence an improved method over SOM. LVMs may be used to explain the observed data in terms of fewer latent (hidden) variables, and they have been used effectively as DR tools in pattern recognition and machine learning [32]. GTM finds a low-dimensional nonlinear manifold embedded in the high-dimensional space. The method has previously been used for visualization purposes in design optimization [33]. Here, we use it to generate a low-dimensional surrogate model of the optimization problem and study its effectiveness in finding an optimal design.

We begin in the next section with a description of the methodology of GTM and a demonstration of the method, using a two-dimensional (2-D) problem reduced to one dimension. The GPLVM method, which GTM will be compared with, is then discussed briefly. The methods are then applied to an airfoil parameterization problem in wing design.

II. Dimension Reduction Using Generative Topographic Mapping

A. Methodology

The method of GTM can be used to model the distribution of a high-dimensional data set in terms of low-dimensional latent/hidden variables. A GTM model represents a high-dimensional data set $\mathbf{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_\mathbf{N}\}$ of D dimension and N points in terms of a low-dimensional L space of K latent variable points $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_\mathbf{K}\}$ through a mapping $\mathbf{y}(\mathbf{x}, \mathbf{W})$, which maps every point in the latent space to the centers of Gaussian mixture distributions of the data. Its parameters are the weights $\mathbf{W}$ of each latent point to the data point and the variance parameter β of the Gaussian mixture distribution. The points in latent space are confined to an L-dimensional manifold nonlinearly embedded in the D space, and they are much fewer in number compared with points in data space. Let $\mathbf{p}(\mathbf{x})$ be the probability distribution over the latent space, which will induce a probability distribution in the data space given by

$$p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \left(\frac{\beta}{2\pi}\right)^{-D/2} \exp\left\{-\frac{\beta}{2}\sum_{d}^{D}[t_d - y_d(\mathbf{x}, \mathbf{W})]^2\right\} \quad (1)$$

Integrating out the latent variable

$$p(\mathbf{t}|\mathbf{W}, \beta) = \int p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta)p(\mathbf{x})\,\mathrm{d}\mathbf{x} \quad (2)$$

and choosing $\mathbf{p}(\mathbf{x})$ as a set of K equally weighted delta functions on a regular grid, we get

$$p(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K}\delta(\mathbf{x} - \mathbf{x_k}) \quad (3)$$

$$p(\mathbf{t}|\mathbf{W}, \beta) = \frac{1}{K}\sum_{k=1}^{K} p(\mathbf{t}|\mathbf{x}_k, \mathbf{W}, \beta) \quad (4)$$

This model maps each latent point to the center of a Gaussian lying on the manifold embedded in D space. The centers of the Gaussian cannot move independently of each other, and so they are a constrained mixture of Gaussians depending only on the mapping $\mathbf{y}$. Also, all components of the mixture have the same variance β^{-1} and mixing coefficient $\frac{1}{K}$. GTM follows a Bayesian inference using a maximum likelihood estimation (MLE) to determine the parameters $\mathbf{W}$ and β. The likelihood function for the data set is

$$\mathcal{L} = \prod_{n}^{N} p(\mathbf{t}|\mathbf{W}, \beta) = \prod_{n}^{N}\left[\frac{1}{K}\sum_{k=1}^{K} p(\mathbf{t_n}|\mathbf{x}_k, \mathbf{W}, \beta)\right] \quad (5)$$

for which the log likelihood is

$$l = \sum_{n}^{N} \ell_n\left[\frac{1}{K}\sum_{k=1}^{K} p(\mathbf{t_n}|\mathbf{x}_k, \mathbf{W}, \beta)\right] \quad (6)$$

which is maximized with respect to both $\mathbf{W}$ and β using an expectation-maximization (EM) algorithm [34]. A linear regression model is chosen for $\mathbf{y}(\mathbf{x}, \mathbf{W})$ with a linear combination of a set of M fixed nonlinear radial basis functions of the form

$$\phi_m(x_k) = \exp\left\{-\frac{\|x_k - \mu_m\|^2}{2\sigma^2}\right\} \quad (7)$$

where μ_m are centers of the radial basis functions, and σ is their common variance. The regression equation in matrix form is

$$\mathbf{Y} = \Phi\mathbf{W} \quad (8)$$

$\mathbf{W}$ is an $M \times D$ matrix of weights, $\mathbf{Y}$ is a $K \times D$ matrix of the Gaussian mixture component centers, and $\mathbf{\Phi}$ is a $K \times M$ matrix of basis functions. In the start of the maximization of likelihood, $\mathbf{W}$ is initialized using the first L principal components of the data set and later updated during the EM algorithm. GTM acts as a supervised learning process in which the training data includes the variables

along with their response values, making the data set dimension $D + 1$ during computations. A detailed mathematical derivation of the nonlinear relations and an explanation of the method is available in Bishop et al. [35].

B. Generative-Topographic-Mapping-Based Optimization

Our GTM-based optimization (GTMBO) algorithm is shown in Fig. 1 and more fully described in the Appendix. This method follows a surrogate-model-based approach, starting with a DOE sample and then using the GTM training; a manifold is generated for this DOE, which acts as the surrogate. The latent points corresponding to this low-dimensional manifold are then searched for the best point that, following full analysis, is added to the training data set before retraining the GTM. Any optimization algorithm is used to search the latent points for the optimum value. The search/update iterations are continued until there is no further improvement in the design. In this paper, in order to show robustness of the method, a range of different Latin-hypercube-based DOEs are used and the optimum obtained is averaged. The error of the model is quantified using a root-mean-square error (RMSE) metric on a separate validation data set. The GTM uses Bayesian inference for its training using MLE; hence, we combine statistical analysis of data along with optimization methods by using GTM manifold as a surrogate to find the optimum. The novelty in this surrogate method is its ability to use a surrogate, which has reduced dimensions, and to perform the optimization in that reduced space rather than the real data space. GTM creates a low-dimensional manifold trained to fit the DOE, and it is on this manifold we perform our optimization. The manifold being low-dimensional aids a fast optimization. The rest of the procedure is the same as any surrogate modeling method. From experiments, we also found that retaining only the best 20% of initial designs for GTM training can improve the usefulness of the GTM manifold during this process and, consequently, the optimum it finds.

There are a number of hyperparameters involved in the GTM method, and an intelligent choice of these parameters is essential for the method to accurately model the distribution of the data set. Some of them, like the basis functions to be used, the number of basis functions, the choice of prior overweights $\mathbf{W}$ during regularization, and the number of latent points, have already been investigated in the detailed work on GTM by Bishop et al. [36]. In this work, the interest is on the optimization of the objective function in the reduced latent space generated by GTM; hence, we examine the parameters that may be crucial in getting a reasonable optimum:

The first parameter is training cycles. Theoretically, the number of EM cycles used for training the GTM can be determined by plotting the negative of the log likelihood of the model that is to be minimized for a training data set and a validation data set against a number of cycles [37]. In this work, we have chosen the number of cycles that are required for convergence of the likelihood value.

The second parameter is latent points. The latent space is a uniformly spaced mesh of points. The choice of the number of these points ($K = \texttt{meshsize}^L$) 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 additional computational effort. In the past, GTM has been used for visualization, and hence the latent space dimension typically does not exceed two. But in the case of DR for optimization, we wish to deal with higher latent space dimensions in order to be able to capture the optimum design effectively. It may be noted that the computational cost of GTM grows exponentially with the number of latent dimensions. The lattice structure considered for the latent space now assumes enormous size, and we are again faced with the curse of dimensionality, but this time it is for the latent space. This phenomenon has been discussed before in the literature, and the solution found was a random sampling. The theory of random projections first discussed by Kaski [38] and Kohonen et al. [39] provides useful information for adopting such an approach. Hence, the approach we used for dealing with high-dimensional latent spaces is to use a random sampling of the latent space, as proposed by MacKay [18]. It is also useful to test different values of K for different samples and different latent space dimensions to decide on the best choice. The metric used for deciding this criterion is the RMSE. It is calculated with the validation data set as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_i^{n_v} (y^i - \hat{y}^i)^2}{n_v}} \tag{9}$$

where y is the function value calculated from the validation data set having n_v data points, and $\hat{y}$ is the predicted function value retrieved from the GTM. The number of latent points that have the smallest RMSE, or do not further reduce RMSE considerably, is chosen: a scheme built from the null hypothesis used in polynomial curve fitting [40].

C. Illustration of Generative-Topographic-Mapping-Based Optimization: Two-Dimensional Branin Function

The modified form of the Branin function [41] is used to illustrate the method. The function to be minimized is given by

$$f(t) = \left(t_2 - \frac{5.1}{4\pi^2} t_1^2 + \frac{5}{\pi} t_1 - 6\right) + 10\left[\left(1 - \frac{1}{8\pi}\right)\cos t_1 + 1\right] + 5t_1$$
$$t_1 \epsilon [-5, 10], \qquad t_2 \epsilon [0, 15] \tag{10}$$

This modification to the original Branin function is performed so as to make the optimization problem harder, as the function now has two local minima and one global minimum instead of three equal global minima. A not-so-efficient optimizer can easily get stuck in the local optima and fail to locate the actual optimum.

Figure 2 shows how a one-dimensional (1-D) GTM manifold resides in the 2-D contour space of the 2-D Branin function $f(t_1, t_2)$, in which the DR was from 2-D data space to 1-D latent space. Each point on the manifold is colored with the corresponding function value obtained from GTM training, and they match (in most places) with the contour colors, i.e., the actual function value. The manifold points are hence encircled in white to be differentiated from the contours. Note that the training points, shown as $\times$s, may not exactly lie on the manifold; they only determine the shape of the GTM manifold. It is important to understand that if the GTM is used to model the data space, only those combinations of the original variables that lie on the manifold can be reached. Thus, when the GTM is sampled during optimization, only points along the manifold will be investigated.

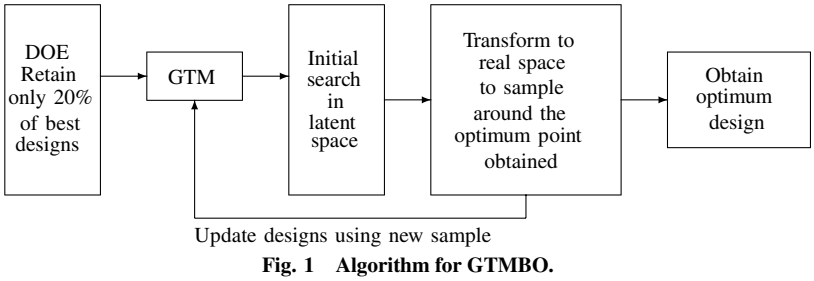


Fig. 1 Algorithm for GTMBO.

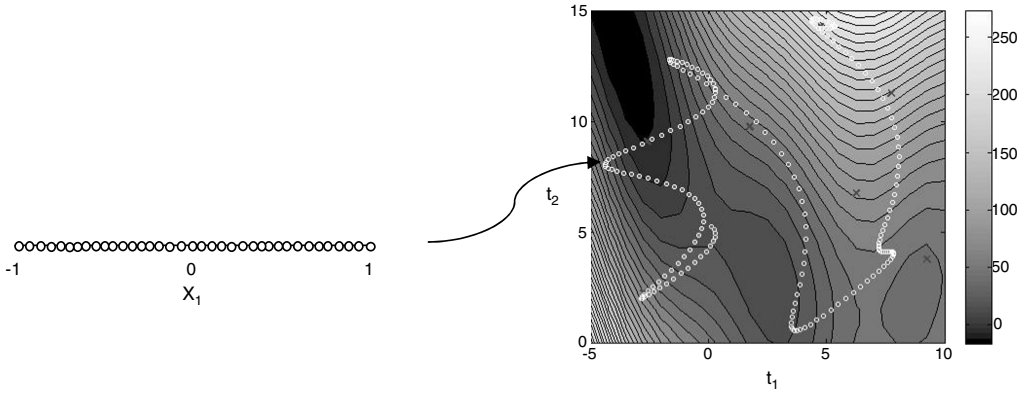


Fig. 2 GTM of Branin function: points in 1-D latent space (left) are confined to a manifold lying in 2-D data space (right).

The method of GTM is implemented here via MATLAB, using the toolbox routines provided by Nabney [42]. A 10-point space-filling Latin hypercube DOE is used as the training sample. Here, we did not use the 20% best designs, since there are only few data points and the dimension of the problem is low. The GTM parameters estimated (as discussed in the previous section) are 20 latent points, 8 basis function centers and 10 EM cycles of training. The true function contours and the training of the GTM manifold for the 10 EM cycles, along with optimization prediction and updates of the function, are shown in Figs. 3a–3f. A black $\times$ in the figure represents the training points. The GTM manifold has an initial configuration of a straight line, with $\mathbf{W}$ being the principal components of the data. It is colored according to the function values that the GTM training gives. The GTM training is performed until the likelihood function value converges to a maximum value (10 cycles in this case). The manifold is visibly spreading out over the contours as training progresses. The minimum of the function given by GTM prediction is shown in Fig. 3e, which is not exactly the global optimum but very close to it. The shapes the manifolds take do not in any way depend on the function; instead, they depend solely on the training data provided to it. So a sample that is spread out evenly in the design space is desirable for the GTM model to be trained to the landscape of the function, and hence to find the optima. In this case, the manifold has ventured close to all the optima, only because the training data were space-filling design. It is to be noted that GTM does not find the global optimum unless the manifold exactly passes over it. This is where updating the data set helps converge the manifold to the optimum. The update points are found on the manifold for each iteration. After five updates on the Branin function, the manifold has changed shape and the optimum has improved from the first prediction, as shown in Fig. 3f. The update points are shown as a white $\times$s. The total computational time is 7 s, with 5 s for sampling the DOE and 2 s for building, training, and optimizing the GTM.

III. Dimension Reduction Using Gaussian Process Latent Variable Model

To provide a comparator with GTM, the DR method of GPLVMs introduced by Lawrence in 2003 [43] (see also [44]) is discussed next. Many variants of the method have been developed over recent years [45–48]. The GPLVM method is also referred to as nonlinear probabilistic PCA. GPLVM uses a nonlinear kernel function and finds the solution of parameters by maximizing the likelihood. The latent dimension points are initially positioned with respect to each kernel and are iteratively changed to best suit the high-dimensional data set by maximizing the likelihood using an optimization process. The latent space variables $\mathbf{X}$ are not known a priori, so they are initialized using PCA. Then, corrected estimations of $\mathbf{X}$ are performed iteratively until convergence is achieved or until a maximum number of iterations is reached.

For the kernel, here, we use a radial basis function with a width parameter σ, which is determined similarly to how the width was estimated in GTM. The steps involved are essentially the same as those of GTM, starting with a data set to be trained. The new latent points obtained after training are used to find the optimum of the function in latent space. The prediction of GPLVM for any new point in the latent space $\tilde{\mathbf{x}}$ in data space is then given by

$$\tilde{\mathbf{y}} = \hat{k}(\tilde{\mathbf{x}})\mathbf{k}^{-1}\mathbf{X} \tag{11}$$

where $\hat{k}$ is the kernel function value evaluated with the new latent point and $\mathbf{k}$ is the radial basis kernel matrix. The actual function is evaluated for $\tilde{\mathbf{y}}$ and appended to the training data set vector as an update.

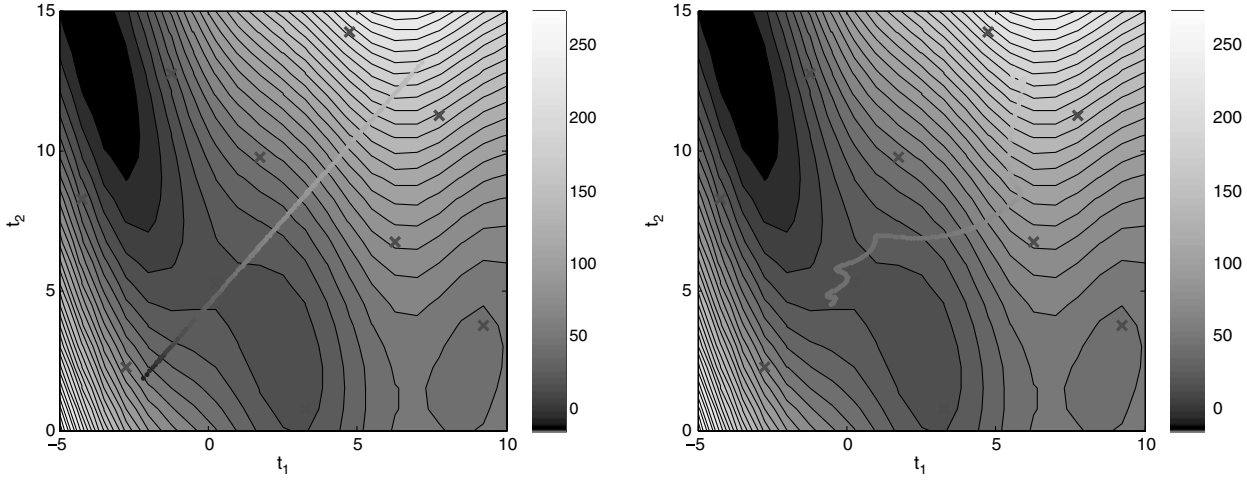
IV. Application to Transonic Airfoil

A. Test Problem

An airfoil design problem is next used to test the DR approaches. The RAE-2822 airfoil is parameterized using nonuniform rational B-splines (NURBS) for the upper and lower surfaces [49]. The positions and weights of the control points defining the NURBS are optimized using a quasi-Newton BFGS optimization, as per Lepine et al. [50]. The resulting parameterization of the RAE-2822 is shown in Fig. 4. It consists of a total of 18 control points, for which the degrees of freedom are the variables of the design. The leading- edge control point starts from the x-axis value, $x = 0$, and the y-axis value, $y = 0$, and trailing-edge control point is at $x = 1$ and $y = 0$. The leading- and trailing-edge control points are fixed, and the two control points on the line, $x = 0$, are only permitted to move vertically to maintain curvature continuity at the leading edge, while the remaining control points can move in both axes. Hence, there are 15 design variables on each of the upper and lower surfaces. The weight of each control point remains fixed for the purposes of the following optimizations, but they could be permitted to vary, thereby increasing the dimensionality of the problem. The airfoil is optimized to minimize the drag coefficient C_D using the full-potential solver VGK (viscous Garabedian and Korn) [51] (see Appendix for details about the solver and the flow parameters used in this problem). The RAE-2822 has a drag coefficient of 0.0087 at the flow conditions considered in this study. The shape and pressure distribution are shown in Fig. 5. The VGK solver takes just 4 s per simulation, and so it does not offer a major computational challenge, but the differences in computational time, with and without a surrogate model and DR, can still be appreciated. The speed of the solver allows extensive averaging to be carried out, thus giving a more accurate picture of the performance of the methods tested.

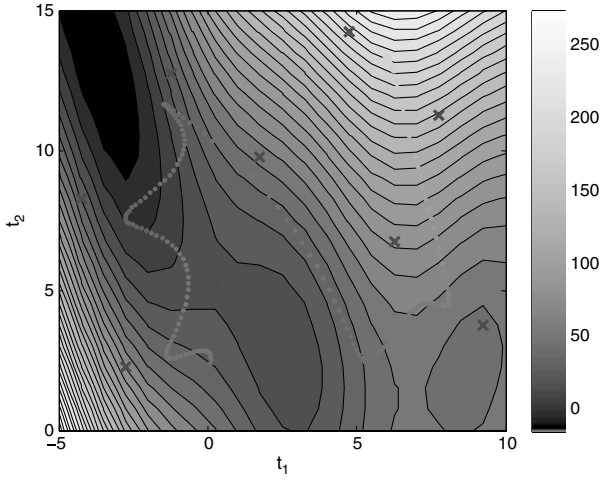
B. Direct Optimization Using Genetic Algorithm

A genetic algorithm (GA) is a stochastic search method and, given sufficient population size and generations, is effective in finding the global optimum of a problem. The random sampling it employs facilitates an extensive search of the design space. But, the number of function evaluations used can prove to be costly for expensive solvers. For the airfoil design problem, the VGK function evaluations take only 2 s; hence, a number of GA searches for averaging purposes

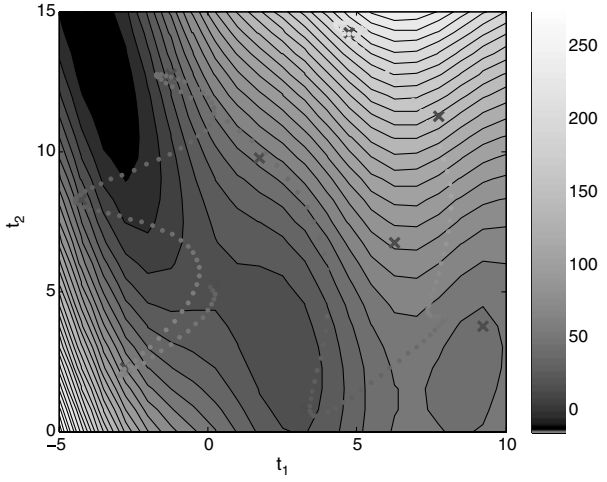


a) Initial configuration (principal component of data)

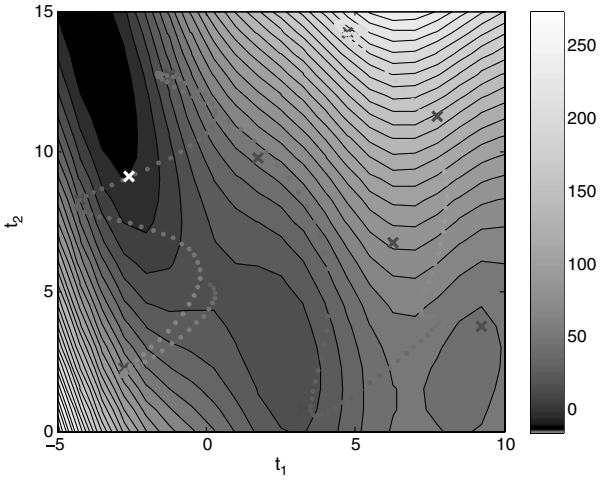
b) After 4 EM cycles



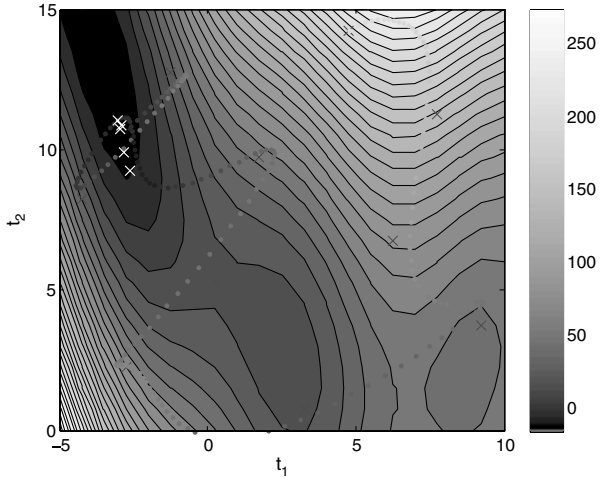
c) After 8 EM cycles



d) After 10 EM cycles



e) GTM-predicted optimum value on manifold



f) After 5 cycles of updating

Fig. 3 GTMBO on Branin function.

was possible. This also provides an estimate about the global optimum, which can be compared with that obtained from other methods.

A GA implemented in the MATLAB software is used here along with a steepest descent method (implemented in MATLAB as function `fmincon`) at the end of each GA run to obtain better convergence of the solution. A budget of 10,000 function evaluations was used by the GA, with a population size of 200 and 50 generations. But, the actual number of VGK evaluations that were successful was only, on average, 100 for each generation. The GA

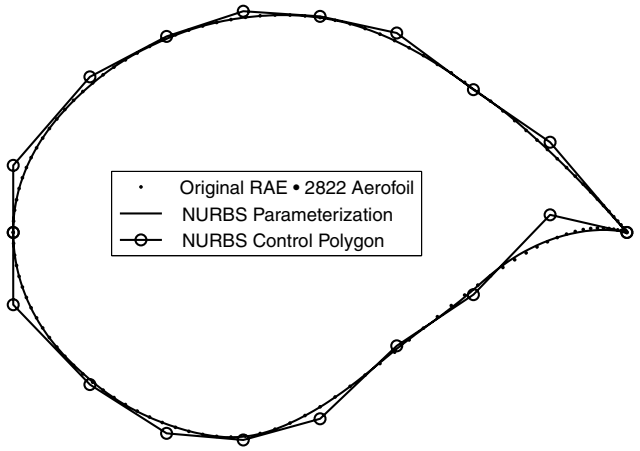


Fig. 4 NURBS parameterization of RAE-2822.

search was carried out with 50 different random seeds, and the results were averaged.

C. Dimension Reduction

The DR using GTM is performed in dimensions from one to four, and the results were compared to determine which dimension best captures the optimum. It starts with a PCA initialization and then continues with training with the EM algorithm to form a low-dimensional manifold embedded in the high-dimensional space. This low-dimensional latent space is then searched for the minimum drag value. The GTM model is built for this function using a 100-point space-filling Latin hypercube DOE. The parameterization was intentionally made to generate only a few good designs out of the 100, and so the designs successfully analyzed by VGK out of these was, on average, 15. In a 30-dimensional (30-D) hypercube, this represents very little data indeed. The best 20% of these designs are adopted for constructing the GTM manifold. This approach reduces the number of designs used in GTM training to around four or five. The function is modeled with latent space of dimensions one to four. The number of updates was set as 15 so as to explore the optimal point found by GTM. The GA is used to search the low-dimensional latent space for the best latent space vector that, when transformed to real space, gives the best 30-D vector of positions of control points corresponding to the predicted minimum C_D value. The best minimum C_D value obtained improved with increasing latent dimension until after two; there is no improvement showing that 3-D latent space can effectively model the 30-D design space. The latent points are placed in a 400^1, 30^2, 15^3, and 8^4 grid for one-, two-, three-, and four-dimensional (4-D) GTM, respectively. The number of basis functions is half that of the latent points, and 10 cycles of the EM algorithm were used. For higher dimensions of the latent space, the grid structure of latent points gives rise to a stack overflow due to the large number of points, and so a random sampling has to be resorted to after 4-D GTM. However, here, the function did not improve after four dimensions, and so we do not explore beyond 4-D latent spaces. The validation set sample size was 50. The results were averaged over 50 different DOEs.

D. Other Methods of Comparison

For purposes of comparison with the other methods, we choose GPLVM, the traditional method of Kriging surrogate models and the geometric filtration method implemented by Toal et al. [49]. The DOE sample used is the same as that adopted for the GTM that is a 100-point space-filling Latin hypercube sample. For simple Kriging and GPLVM, all the feasible designs generated by VGK out of the 100 initial designs are used to search for the optimum, since experiments showed that reducing the number of designs gave worse results for these methods.

GPLVM was performed using the DR toolbox developed by van der Maaten et al. [12]. As already noted, it uses the method of PCA to initialize the latent points and a conjugate gradient method to optimize the likelihood function. The kernel function used is a radial basis function. The dimension of the latent space was chosen as two, since 2-D GTM was found effective and the number of VGK updates was 15. The kernel width parameter σ was found by trying different values for σ, varying between 0.05 to 1, and using the value that best optimized the likelihood function (similar to the method of finding radial basis function widths in GTM).

Kriging was performed in the 30-D space using the GA to optimize the Kriging parameters and to search the resulting 30-D Kriging prediction with 15 updates to the training set. The method of Kriging is closely related to Gaussian process regression and is a good comparative surrogate method to GTM.

In the geometric filtration method, a 2-D PCA is performed on the best 20% of the sample before building a Kriging model in principle component space. This is similar to the filtration that we performed for GTM, and we performed the same reduction for PCA to a 2-D space. The GA is used to find the optimal values for the Kriging parameters.

E. Results and Discussion

The comparison of the optimum C_D obtained for all the preceding methods and the computational time each method takes are shown in Table 1. The method of GTM shows good results with 2-D latent space, indicating that a low dimension can effectively model the design space of different airfoil shapes generated by the 30-variable NURBS parameterization. The latent points required increases with the dimension of latent space; hence, higher-dimensional GTM involves more training time. Two-dimensional GTM gave good

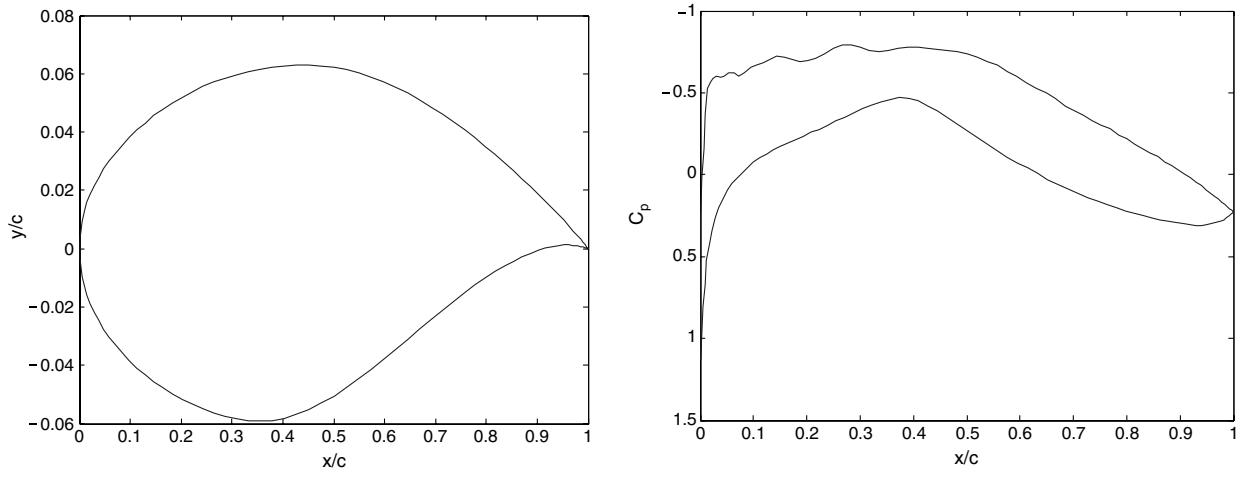


Fig. 5 RAE-2822 airfoil and pressure distribution.

Table 1 Comparison of different methods for minimum drag coefficient in optimization of RAE-2822

	Averaged optimum C_D (drag counts)		VGK evaluations/DOE		
Method	Mean C_D	Standard C_D	Attempted	Used	Time/DOE, min.
GA/gradient search	79	0.8	5000	3000	696.5
1-D GTM	89	3.93	115	17	5.7
2-D GTM	87	3.72	115	17	6.8
3-D GTM	87	3.65	115	17	11.5
4-D GTM	87	3.57	115	17	20
2-D GPLVM	93	6.51	115	30	4.6
Kriging	91	6.03	115	25	8.2
2-D geometric filtration	94	8.13	115	16	4

results in less time and is more attractive than any other dimension. The RMSE value for the validation data set containing 50 data samples gave a value of 18 for all dimensions of GTM. The direct GA search involved a great many function evaluations, which in this case is not very time consuming but would be unaffordable in the case of high-fidelity simulation models. Kriging with a geometric filtration via PCA is a very effective method, but the computational time for tuning the Kriging parameters can become higher than the GTM training for large numbers of samples. Hence, both in the aspects of computational time taken and in the case of function evaluations, the GTM method is a competitive approach. In the case of GPLVM, the time taken was similar to GTM, but the optimization of the likelihood function to obtain the latent space model sometimes fails due to ill-conditioned matrices in the kernel functions, and hence was not very desirable compared with the ease of using GTM. Various optimal airfoil shapes and their pressure distribution with different dimensions of GTM are shown in Fig. 6 and, with different comparison methods, they are shown in Fig. 7. Also, the optimization histories for 2-D GTM, which gave the best optimum values for 50 different DOEs, are shown in Fig. 8. The optimized design by 2-D GTM, as against the RAE-2822 original design, is also shown in the figure.

The 30-D space could be reduced to 2-D space by GTM, because the underlying geometry of the airfoils that were represented using the naive 30 NURBS variables might have a more sensible dimension of two, which may be derived by GTM. This supports some of the earlier representations of airfoil geometries, like the NACA definitions of airfoils with their thickness-to-chord ratio and camber, even though what the two latent dimensions of GTM translate to in real space may not be as intuitive. To investigate what the

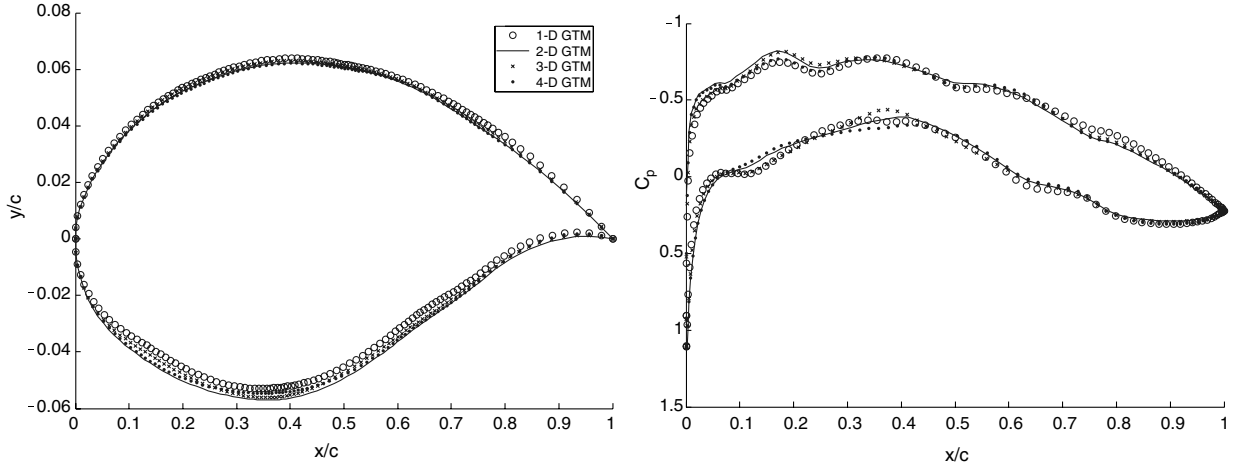


Fig. 6 RAE-2822 airfoil optimized by various dimensions of GTM along with pressure distribution.

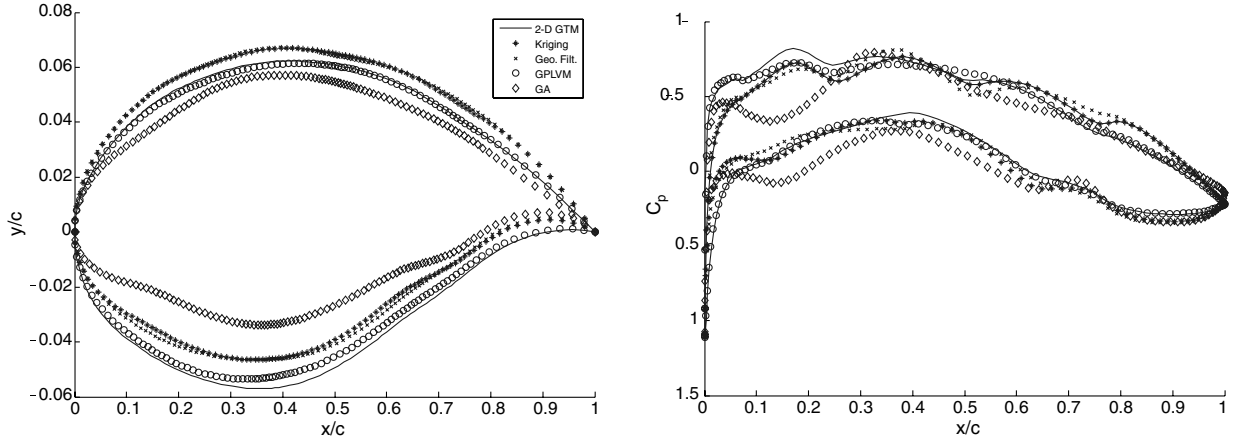


Fig. 7 RAE-2822 airfoil optimized by other methods of comparison along with pressure distribution.

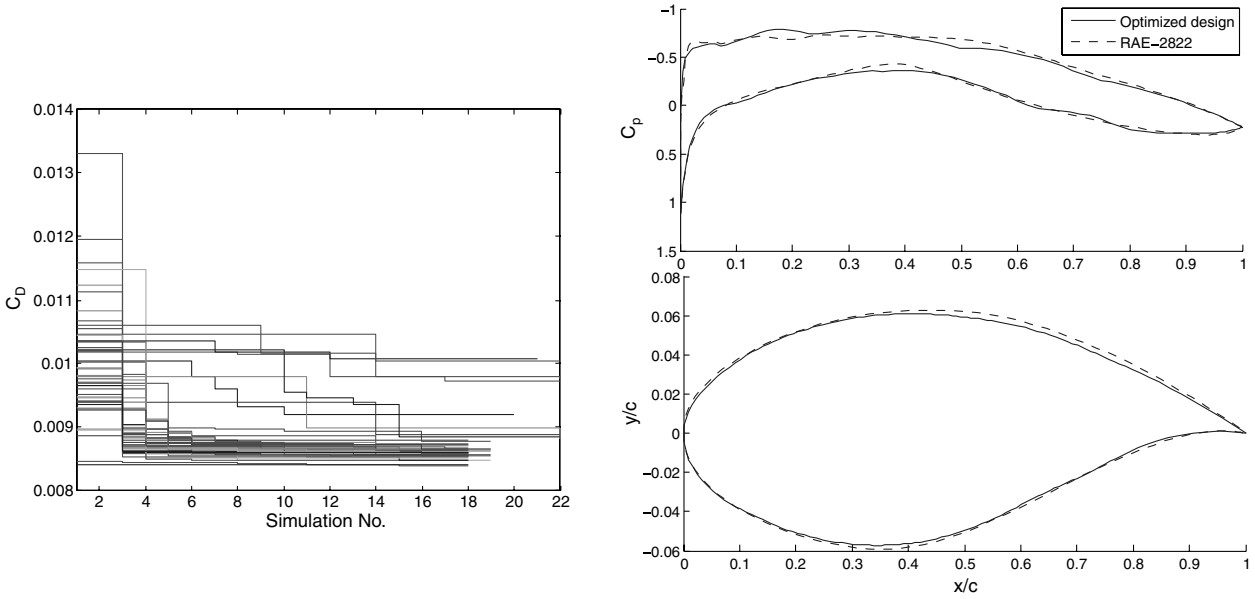


Fig. 8 Optimization histories for 50 DOEs and optimized design for 2-D GTM.

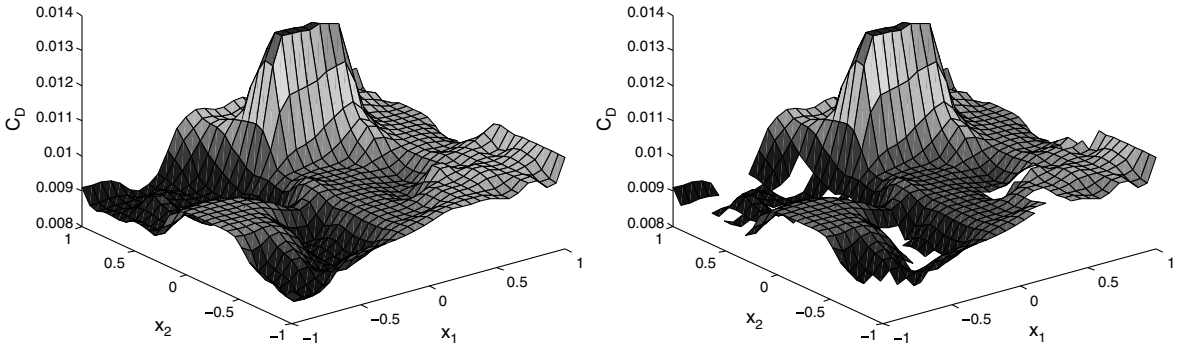


Fig. 9 Latent space manifold with 2-D GTM for objective function C_D.

GTM-predicted airfoils look like, we took one specific DOE, which is reduced to 2-D space using GTM, and plotted the C_D contour in the latent space, which can be obtained with a resolution of 30×30 latent grid. Figure 9 shows the actual and the corrected latent spaces. The corrected latent space is obtained after removing points that, when reconstructed back to real space, did not give meaningful designs (in this case, giving nonairfoil-like designs). Now that a function manifold is obtained, it is interesting to study how the corresponding airfoils with the 30-D representations look for this space, especially in regions that gave high C_D values, which should be some bad designs. Twenty-one airfoils (for convenience in plotting) are generated from the region in latent space, which showed a high peak by taking a constant x_1 value around that location (around 2.4) and an x_2 value in the range [0, 1] so as to capture the peak area. Figure 10 shows the true C_D values evaluated using the VGK code and GTM-predicted C_D values for 21 airfoils along the line $x_1 = 0.24$ and $x_2 \epsilon [0, 1]$. It can be observed that, at the peak of C_D, some airfoils show a poor geometry, which explains their high drag levels. So, the NURBS representations give many bad geometries, which are filtered by the GTM during its search in the latent space. The best value for C_D obtained from this DOE was 87 drag counts.

The effectiveness of any proposed method can only be appreciated when it is compared with similar techniques used in the past; hence, three other methods were discussed for this purpose.

The method of Kriging surrogate models is an effective comparison, as we are proposing GTM as a surrogate model. Kriging has been found to consume more computational time in tuning its parameters and to be less effective when supplied with a small sample set. It is this weakness that was exploited in this airfoil test case when VGK produced only a sample as few as 15 from the initial sampling plan of 100. The reason Kriging gives a higher value for the optimum might be this, and it can definitely be improved with more points in the sample set to build the Kriging model. But, this would mean more function evaluations. On the contrary, GTM works better with fewer designs than with more, as it helps focus the GTM manifold; out of the designs generated by VGK, only 20% are used. The computational time benefits cannot be truly appreciated in this problem, as the time for Kriging tuning is comparable with GTM training. But it will be more for Kriging in a problem that has a greater number of design variables and more sample points for building the Kriging model. Overcoming the problems in Kriging, the geometric filtration method is the next effective method to compare with GTM. It employs PCA to filter designs for Kriging, and since GTM also starts with a PCA initialization, it performs equally well and gives an optimum C_D value slightly better than geometric filtration. The computational time taken is also similar, since a Kriging model is built in a low-dimensional PC space, even though the tuning of Kriging parameters can again increase with more sample points. A comparison with another nonlinear DR method GPLVM is finally performed, as it is a nonlinear variant of PCA involving a maximum likelihood maximization principle similar to GTM; hence, it is expected to be as efficient as GTM. In terms of computational time (effectiveness in finding the optimum and number of function evaluations), it worked as well as GTM, although GTM gave a lower value for optimum C_D.

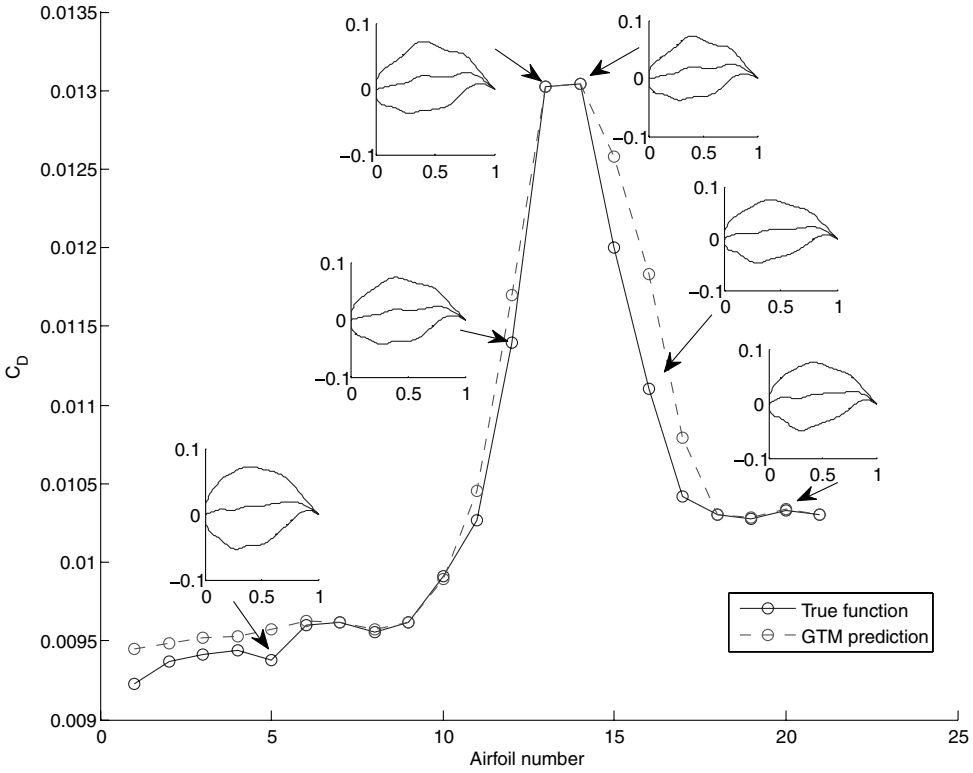


Fig. 10 Twenty-one true and GTM given C_D values along the latent space dimension of $x_1 = 0.24, x_2 \epsilon [0, 1]$, and six 30-D NURBS airfoils corresponding to some of the C_D values.

V. Conclusions

DR is highly desirable capability when dealing with optimization problems having high-dimensional data sets and expensive solvers. The parsimony achieved needs to be traded off against any computationally intensive training required by the DR methods. DR can also indicate the lowest dimension that will effectively define the problem completely. This information is sometimes useful to engineers for an understanding of underlying patterns in the problem and for visualizing the high-dimensional problem. Several statistical tools that provide nonlinear DR methods have been discussed in this work. The method of GTM has a mathematical basis in Bayesian modeling, and hence allows the benefits of statistical modeling to be combined with design optimization by using the method as a form of low-dimensional surrogate for optimization problems.

The method is illustrated on a 2-D problem of Branin function, and it is shown that the GTM manifold spans the design landscape in an effective way, so that on searching this low-dimensional space, we obtain the optimum, provided a space-filling design is used for the GTM training. It is shown that using the best 20% of the initial sampling plan can adequately train the GTM to find the optimum. With the insight from a small test case, GTM is applied to a bigger and real engineering problem from aerodynamic design. A NURBS parameterization of a RAE-2822 airfoil involving 30 design variables as the control points of the NURBS curve is modeled with GTM of dimensions from one to four. Increasing the number of latent dimensions from four presented an increase in the computational cost of the problem. However, it was found that two dimensions could be used effectively to find an optimum drag coefficient C_D. A comparison with direct search, Kriging-based surrogates, geometric filtration, and GPLVMs demonstrated that a GTM-based approach is competitive, both in terms of the quality of the final optima produced and the computational cost.

Appendix

I. Steps in Generative-Topographic-Mapping-Based Optimization

1) The first step is to generate data sets. The data set matrix $\mathbf{T}$ contains normalized variables and the objective function of dimension $(N \times D + 1)$. A space-filling Morris–Mitchell Latin hypercube [10] with max–min criterion is used for the DOE. The validation data set is chosen from a different DOE.

2) The second step is GTM training. The setup and training of the GTM can be performed using the toolbox subroutines available in Netlab [42]. There were three changes made in these routines for the purpose of optimization. First, for high-dimensional latent spaces with more than four dimensions, the latent space points and the basis function centers are generated from a space-filling Latin hypercube rather than a uniform grid of points; hence, a space-filling Latin hypercube sampling of the latent space is implemented. Latin hypercubes follow from a uniform grid of points; hence, the underlying prior distribution of the latent space is still considered as a uniform distribution. Second, the width of the basis function σ was originally considered as a constant throughout the training process for a data set. This is made a hyperparameter, which has to be estimated during training. A range of values from $[2^{-2}–2^2]$ is considered and, for each value, the GTM trained and the value that gives the maximum log likelihood while training is taken as the best value of σ [36]. Third, the training can be improved by the use of a GA for training the GTM first followed by EM cycles. This is because the EM algorithm is a local optimizer, so the likelihood of the function may be maximized further by using a global optimizer like the GA to start the search.

3) The third step is the search in latent space. The latent space is searched for a suitable update point using a GA on latent variables $\mathbf{x}$ with bounds $[-1, 1]^L$, and the searched function is the regression equation of the GTM. Similar to $\mathbf{t}$, $\mathbf{y}$ has D columns of data variables, and the $D + 1$th column contains the function value predicted by GTM training. This predicted value is the goal optimized by the GA. It was also helpful to have a steepest descent search after the GA search to give better convergence.

4) The fourth step is 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 [31]

$$\mathbf{x}_n^{\text{mean}} = \sum_k^K \mathbf{x}_k p(\mathbf{x}_k|\mathbf{t}_n) = \sum_k^K \mathbf{x}_k \mathbf{R}_{\mathbf{kn}} \tag{A1}$$

These positions of data points in latent space $\{\mathbf{x}_1^{\text{mean}}, \ldots, \mathbf{x}_N^{\text{mean}}\}$ can be substituted in Φ in the regression equation to get their corresponding values in data space.

5) The fifth step is updating the training set. The actual function is evaluated using the best variable found by the GA/gradient-based search after optimizing the GTM function value. This new value is appended to the data set vector $\mathbf{T}$ as an update, and the preceding steps from step 2 continued for 5 to 15 iterations.

II. VGK Method for Calculation of Drag Coefficient

VGK is a computational fluid dynamics method for 2-D single-element airfoil. It is an iterative solution to a viscous coupled finite difference code that solves the full-potential equations. It is coded in FORTRAN, and we have used the executable through MATLAB wrapper functions. VGK was developed at the Defense Evaluation and Research Agency (DERA) , Farnborough, and it is available from Engineering Sciences Data Unit (ESDU) International, plc, under the terms of an agreement with DERA. We used a fixed lift VGK approach, with the angle of attack α found by interpolation from values $\alpha = 0, 0.5$, and 1 to the fixed C_L value. The input parameters of the VGK used in this study are as follows: 1) fixed lift coefficient $(C_L) = 0.45$; 2) number of radial mesh lines in the fine grid is equal to 160; 3) number of iterations of the inviscid flow element using the coarse mesh is equal to 100; 4) number of iterations of the inviscid flow element using the fine mesh is equal to 200; 5) relaxation parameters associated with the viscous boundary conditions in the coarse-mesh $(\omega_c) = 0.15$ and $(k_c) = 5$, and fine-mesh $(\omega_f) = 0.075$ and $(k_f) = 5$; 6) artificial viscosity parameter $\text{EP}(\epsilon) = 0.8$; 7) partially-conservative parameter, $(\lambda) = 0.25$; 8) Mach number $(\text{EM}) = 0.7$; 9) Reynolds number, $Re = 6 \times 10^6$ cosh; and 10) transition points on upper and lower surfaces are equal to 0.07.

Drag determination in VGK code uses the far-field approach to find C_D here. The overall drag is expressed as the sum of viscous- and wave-drag contributions given by

$$C_D = \text{CDV} + \text{CDW} \tag{A2}$$

where CDV is viscous drag coefficient and CDW is wave-drag coefficient estimated using the first-order method denoted by C_{D1} or the improved method denoted by C_{D2}.

Acknowledgments

This work is supported by Rolls–Royce, plc., and the United Kingdom Department of Trade and Industry.

References

[1] Bellman, R., *Adaptive Control Processes: A Guided Tour*, Princeton Univ. Press, Princeton, NJ, 1961, p. 94.

[2] Myers, R. H., and Mongomery, D. C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, Wiley Series in Probability and Statistics, Wiley, New York, 1995, p. 73.

[3] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, Vol. 13, No. 4, 1998, pp. 455–492. doi:10.1023/A:1008306431147

[4] Loeppky, J. L., Sacks, J., and Welch, W. J., "Choosing The Sample Size of a Computer Experiment: A Practical Guide," *Technometrics*, Vol. 51, No. 4, 2009, pp. 366–376. doi:10.1198/TECH.2009.08040

[5] Tenenbaum, J. B., de Silva, V., and Langford, J. C., "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, Vol. 290, No. 5500, 2000, pp. 2319–2322. doi:10.1126/science.290.5500.2319

[6] Elster, C., and Neumaier, A., "Screening by Conference Designs," *Biometrika*, Vol. 82, No. 3, 1995, pp. 589–602. doi:10.1093/biomet/82.3.589

[7] Bettonvil, B., and Kleijnen, J. P. C., "Searching for Important Factors in Simulation Models with Many Factors: Sequential Bifurcation," *European Journal of Operational Research*, Vol. 96, No. 1, 1997, pp. 180–194. doi:10.1016/S0377-2217(96)00156-7

[8] Andres, T. H., and Hajas, W. C., "Using Iterated Fractional Factorial Design to Screen Parameters in Sensitivity Analysis of a Probability Risk Assessment Model," *Joint International Conference on Mathematical Methods and Supercomputing in Nuclear Applications*, edited by S. Küsters, and W. Werner, Vol. 2, Kernforschungszentrum Karlsruhe, Karlsruhe, Germany1993, pp. 328–337.

[9] Iman, R. L., and Conover, W. J., "Small Sample Sensitivity Analysis Techniques for Computer Models, with an Application to Risk Assessment," *Communications in Statistics: Theory and Methods*, Vol. 9, No. 17, 1980, 1749–1842. doi:10.1080/03610928008827996

[10] Morris, M. D., "Factorial Sampling Plans for Preliminary Computational Experiments," *Technometrics*, Vol. 33, No. 2, 1991, pp. 161–174. doi:10.2307/1269043

[11] Fodor, I., "A Survey of Dimension Reduction Techniques," Center for Applied Scientific Computing, Lawrence Livermore National Lab., TR UCRL-ID-148494, Livermore, CA, 1984.

[12] van der Maaten, L. J. P., Postma, E. O., and van der Herik, H. J., "Dimensionality Reduction: A Comparative Review," Tilburg Univ. TR 2009-005, Tilburg, The Netherlands, 2009.

[13] Joliffe, I. T., *Principal Component Analysis*, Springer–Verlag, New York, 1986, p. 78.

[14] Hastie, T., *Principal Curves and Surfaces*, Stanford Univ., TR 11, Stanford, CA, 1984.

[15] Hastie, T., and Stuetzle, W., "Principal Curves," *Journal of the American Statistical Association*, Vol. 84, No. 406, 1989, pp. 502–516. doi:10.2307/2289936

[16] Schölkopf, B., Smola, A., and Müller, K. R., "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," *Neural Computation*, Vol. 10, No. 5, 1998b, pp. 1299–1319. doi:10.1162/089976698300017467

[17] Durbin, R., and Willshaw, D., "An Analogue Approach to the Travelling Salesman Problem," *Nature*, Vol. 326, No. 6114, 1987, pp. 689–691. doi:10.1038/326689a0

[18] MacKay, D. J. C., "Bayesian Neural Networks and Density Networks," *Nuclear Instruments and Methods in Physics Research, Section A*, Vol. 354, No. 1, 1995, pp. 73–80. doi:10.1016/0168-9002(94)00931-7

[19] MacKay, D. J. C., and Gibbs, N., "Density Networks," *Proceedings of Meeting on Statistics and Neural Nets*, edited by Titterington, K., Edinburgh, Vol. 354, No. 1, Oxford Univ. Press, New York, 1997, pp. 73–80.

[20] Kohonen, T., *Self Organizing Maps*, Springer–Verlag, Berlin, 1995, p. 105.

[21] Demartines, P., and Hèrault, J., "Curvilinear Component Analysis: A Self-Organizing Neural Network for Nonlinear Mapping of Data Sets," *IEEE Transactions on Neural Networks*, Vol. 8, No. 1, 1997, pp. 148–154. doi:10.1109/72.554199

[22] Cox, T. F., and Cox, M. A. A., *Multidimensional Scaling*, Chapman and Hall, London, 1994, p. 31.

[23] Kramer, M. A., "Nonlinear Principal Component Analysis Using Autoassociative Neural Networks," *AIChE Journal*, Vol. 37, No. 2, 1991, pp. 233–243. doi:10.1002/aic.690370209

[24] Roweis, S. T., and Saul, L. K., "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, Vol. 290, No. 5500, Dec. 2000, pp. 2323–2326. doi:10.1126/science.290.5500.2323

[25] Rosipal, R., and Krämer, N., "Overview and Recent Advances in Partial Least Squares," *Subspace, Latent Structure and Feature Selection Techniques*, edited by Saunders, C., Grobelnik, M., Gunn, S., and Shawe-Taylor, J., Vol. 3940, Lecture Notes in Computer Science, Springer–Verlag, Berlin, 2006, pp. 34–51.

[26] Gashler, M., Ventura, D., and Martinez, T., "Iterative Non-Linear Dimensionality Reduction with Manifold Sculpting," *Advances in Neural Information Processing Systems*, Vol. 20, MIT press, Cambridge, MA, 2008, pp. 513–520.

[27] Belkin, M., and Niyogi, P., "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," *Advances in Neural Information Processing Systems*, Vol. 14, MIT Press, Cambridge, MA, 2001, pp. 586–691.

[28] Donoho, D. L., and Grimes, C., "Hessian Eigenmaps: New Locally Linear Embedding Techniques for High-Dimensional Data," *Proceedings of the National Academy of Sciences*, Vol. 102, No. 21, 2005, pp. 7426–7431. doi:10.1073/pnas.0500334102

[29] Zhang, Z., and Zha, H., "Principal Manifolds and Nonlinear Dimension Reduction via Local Tangent Space Alignment," *SIAM Journal on Scientific and Statistical Computing*, Vol. 26, No. 1, 2004, pp. 313–338.

doi:10.1137/S1064827502419154
[30] Weinberger, K. Q., Sha, F., and Saul, L. K., "Learning a Kernel Matrix for Nonlinear Dimensionality Reduction," *Proceedings of the 21st International Conference on Machine Learning*, edited by C. E. Brodley, 2004, pp. 839–846.
[31] Svensén, M., "GTM: The Generative Topographic Mapping," Ph.D. Thesis, Aston Univ., Birmingham, England, U.K., 1998.
[32] Everitt, B. S., *An Introduction to Latent Variable Models*, Chapman and Hall, London, 1984, p. 1.
[33] Holden, C. M. E., "Visualization Methodologies in Aircraft Design," 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY, AIAA Paper 2004-4449, 2004.
[34] Dempster, A. P., Laird, N. M., and Rubin, D. B., "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society, Series B (Methodological)*, Vol. 39, No. 1, 1977, pp. 1–38.
doi:10.2307/2984875
[35] Bishop, C. M., Svensén, M., and Williams, C. K. I., "GTM: The Generative Topographic Mapping," *Neural Computation*, Vol. 10, No. 1, 1998, pp. 215–234.
doi:10.1162/089976698300017953
[36] Bishop, C. M., Svensén, M., and Williams, C. K. I., "Developments of the GTM Algorithm," *Neurocomputing: Variable Star Bulletin*, Vol. 21, Nos. 1–3, 1998, pp. 203–224.
doi:10.1016/S0925-2312(98)00043-5
[37] Bishop, C. M., *Neural Networks for Pattern Recognition*, Oxford Univ. Press, New York, 1995, p. 262.
[38] Kaski, S., "Dimensionality Reduction by Random Mapping: Fast Similarity Computation for Clustering," *Proceedings of the IEEE International Joint Conference on Neural Networks*, Vol. 1, IEEE Publ., Piscataway, NJ, 1998, pp. 413–418.
[39] Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Paatero, V., and Saarela, A., "Self Organization of Massive Document Collection," *IEEE Transactions on Neural Networks*, Vol. 11, No. 3, 2000, pp. 574–585.
doi:10.1109/72.846729
[40] Ralston, A., and Rabinowitz, P., *A First Course in Numerical Analysis*, McGraw–Hill, New York, 1978, p. 254.
[41] Forrester, A. I. J., Sóbester, A., and Keane, A. J., *Surrogate Models in Engineering Design: A Practical Guide*, Wiley, New York, 2008, p. 196.
[42] Nabney, I. T., *Netlab: Algorithms for Pattern Recognition*, 1st ed., Springer, London, 2001.
[43] Lawrence, N. D., "Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data," *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*, edited by S. Thrun, L. K. Saul, and B. Schölkopf, MIT Press, Cambridge, MA, 2004.
[44] Lawrence, N. D., "Probabilistic Non-Linear Principal Component Analysis with Gaussian Process Latent Variable Models," *Journal of Machine Learning Research*, Vol. 6, 2005, pp. 1783–1816.
[45] Lawrence, N. D., and Quiñonero Candela, J., "Local Distance Preservation in the GP-LVM Through Back Constraints," *Proceedings of the 23rd International Conference on Machine Learning*, ACM, New York, 2006, pp. 513–520.
[46] Grochow, K., Martin, S. L., Hertzmann, A., and Popovic, Z., "Style-Based Inverse Kinematics," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004)*, ACM, New York, 2004, pp. 522–531.
[47] Wang, J. M., Fleet, D. J., and Hertzmann, A., "Gaussian Process Dynamical Models," *Advances in Neural Information Processing Systems 18*, edited by Y. Weiss, B. Schölkopf, and J. Platt, MIT Press, Cambridge, MA, 2006, pp. 1441–1448.
[48] Urtasun, R., Fleet, D. J., and Fua, P., "3-D People Tracking with Gaussian Process Dynamical Models," *IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Publ., Piscataway, NJ, 2006, pp. 238–245.
[49] Toal, D. J. J., Bressloff, N. W., and Keane, A. J., "Geometric Filtration Using Proper Orthogonal Decomposition for Aerodynamic Design Optimization," *AIAA Journal*, Vol. 48, No. 5, 2010, pp. 916–928.
doi:10.2514/1.41420
[50] Lepine, J., Guibault, F., and Trepanier, J., "Optimized Nonuniform Rational B-Spline Geometrical Representation for Aerodynamic Design of Wings," *AIAA Journal*, Vol. 39, No. 11, 2001, pp. 2011–2041.
doi:10.2514/2.1206
[51] "VGK Method for Two-Dimensional Aerofoil Sections," IHS ESDU, Rept. 96028, London, 1996.

A. Messac
Associate Editor