Contents lists available at ScienceDirect

Progress in Aerospace Sciences

journal homepage: www.elsevier.com/locate/paerosci

Recent advances in surrogate-based optimization

Alexander I.J. Forrester [*], Andy J. Keane

Computational Engineering and Design Group, School of Engineering Sciences, University of Southampton, SO17 1BJ, UK

ARTICLE INFO

Available online 10 January 2009

ABSTRACT

The evaluation of aerospace designs is synonymous with the use of long running and computationally intensive simulations. This fuels the desire to harness the efficiency of surrogate-based methods in aerospace design optimization. Recent advances in surrogate-based design methodology bring the promise of efficient global optimization closer to reality. We review the present state of the art of constructing surrogate models and their use in optimization strategies. We make extensive use of pictorial examples and, since no method is truly universal, give guidance as to each method's strengths and weaknesses.

Contents

[*] Corresponding author.
E-mail address: Alexander.Forrester@soton.ac.uk (A.I.J. Forrester).

0376-0421/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.paerosci.2008.11.001

1. Introduction

An overview of surrogate-based analysis and optimization was presented in this journal by Queipo et al. [1]. They covered some of the most popular methods in design space sampling, surrogate model construction, model selection and validation, sensitivity analysis, and surrogate-based optimization. More recently Simpson et al. [2] presented a general overview of how this area has developed over the past 20 years, following the landmark paper of Sacks et al. [3]. Here we take a more in depth look at the various methods of constructing a surrogate model and, in particular, surrogate-based optimization. Our review is by no means exhaustive, but the methods we cover are those we feel are the most promising, based on the cited references coupled with our own experience. Parting from a common trend in review papers, we do not include a large, industrial type problem which may not be of interest to all readers. Instead we have employed small illustrative examples throughout the paper in the hope that methods are explained better in this way.

The use of long running expensive computer simulations in design leads to a fundamental problem when trying to compare and contrast various competing options: there are never sufficient resources to analyse all of the combinations of variables that one would wish. This problem is particularly acute when using optimization schemes. All optimization methods depend on some form of internal model of the problem space they are exploring—for example a quasi-Newton scheme attempts to construct the Hessian at the current design point by sampling the design space. To build such a model when there are many variables can require large numbers of analyses to be carried out, particularly if using finite difference methods to evaluate gradients. Because of these difficulties it is now common in aerospace design to manage explicitly the building and adaptation of the internal model used during optimization—these models are here termed surrogate models although they are also often referred to as meta models or response surfaces. This review is concerned with this approach to design search and, in particular, the construction of surrogates and their refinement. Fig. 1 illustrates the basic process (the steps remain the same for any optimization-based search):

1. first the variables to be optimized are chosen, often due to their importance, as determined by preliminary experiments;
2. some initial sample designs are analysed according to some pre-defined plan;
3. a surrogate model type is selected and used to build a model of the underlying problem—for surrogate-based search this process can be quite sophisticated and time consuming;
4. a search is carried out using the model to identify new design points for analysis;
5. the new results are added to those already available and, provided further analyses are desired, the process returns to step 3.

These steps assume that an automated process has been established to carry out design analyses when given a selection of design inputs, and this is a far from trivial process in most aerospace applications. Typically it requires a scheme to create meshed water-tight geometries from a vector of design parameters either using CAD or some specialized product specific code, followed by the use of mesh generation. The resulting mesh is then used in a finite volume or finite difference scheme to solve the underlying equations and is then followed by some design evaluation process to calculate performance metrics such as lift, drag, stress, etc. Throughout the rest of this paper we assume such a process is established, although we allow for the possibility that it may fail to return usable results (perhaps because of convergence failure, for example) or may return results that are contaminated with noise due to round off, discretization or convergence errors.

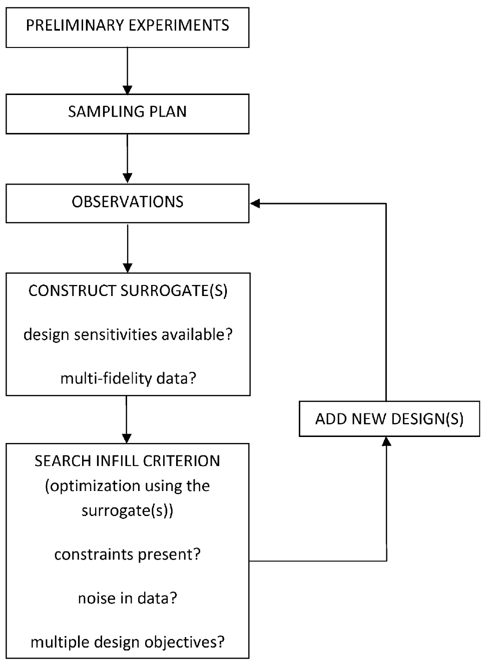

Fig. 1. A surrogate-based optimization framework.

2. Initial sampling

Assuming that we already have a parameterized design coupled to a method of evaluation, the first step in the surrogate-based optimization process is to choose which parameters we wish to vary—our *design variables*. This may be patently obvious in a familiar design problem, but in a new design problem there may be many variables, only a subset of which we can optimize. The problem of obtaining enough information to predict a design landscape in a hyper-cube of increasing dimensions—the *curse of dimensionality*—is what holds us back in terms of the number of variables we can optimize. The amount of information we can obtain will, of course, depend on the computational (or experimental) expense of computing objective and, perhaps, constraint functions. Thus the number of variables we can optimize is a function of this expense. Choosing the variables that will be taken forward to optimize usually requires the design and analysis of some preliminary experiments. This process may in fact be revisited several times in the light of results coming from surrogate-based searches. We will not cover such methods here and the reader may wish to consult Morris [4]. We endorse this reference as it makes the weakest assumptions regarding the type and size of the problem, assuming only that the function is deterministic.

With the design space identified, we must now choose which designs we wish to evaluate in order to construct the surrogate model—our *sampling plan*. It is worth noting here that this process is often referred to as a *design of experiments*, a term used for selection of physical experiments. Here we use the term *sampling plan* to refer to both physical and computational experiments.

To build global models of unknown landscapes, a sampling plan with a uniform, but not regular, spread of points across the design space makes intuitive sense. We also wish to use a sample of points whose projections onto each variable axis are uniform, the logic being that it is wasteful to sample a variable more than once at the same value. To this end we favour the space filling *maximin* [5] Latin hypercube [6] sampling techniques of Morris and Mitchell [7]. An in-depth description of this technique, including *Matlab* code can be found in Forrester et al. [8]. Note that here we are discussing the sample upon which an initial surrogate will be built. Further sampling—sometimes called *adaptive sampling*—can be carried out subsequently (the 'add new design(s)' box in Fig. 1) and will be discussed in due course.

It is also worth noting that for some classes of surrogates it is possible to estimate directly the quality of a sampling plan on the stability of model building, e.g. D-optimal designs when used with polynomial surrogates. If a particular surrogate type is to be used, whatever the outcome of the initial experiments, this may well influence the choice of sampling plan.

3. Constructing the surrogate

In everyday life we try to save time and make predictions based on assumptions. For example, when travelling on a road we can predict the rate of turn of a bend based on the entry and surrounding landscape. Without really considering it, in our mind we are constructing a surrogate using the direction of the road, its derivatives with respect to distance along the road (at least to second order), and local elevation information. This information is coupled with assumptions based on our experience of going round many bends in the past. We also formulate an estimate of the possible error in the prediction and regulate our entry speed based on this and road surface conditions. In unfamiliar surroundings, for example in a foreign country where road building techniques are different, we have to assume a very high error because our assumptions based on experience are likely to be incorrect, and reduce our speed of entry accordingly. In essence we calculate a suitably safe speed based on our prediction of curvature and subtract a safety margin based on our predicted error. In engineering design we are faced with different problems, but in essence we try to do with a surrogate model what we do everyday with our mind: make useful predictions based on limited information and assumptions.

Care must be taken that any assumptions are well founded. The first assumption we make with all the surrogate modelling techniques discussed here is that the engineering function is continuous—incidentally one we also make about roads, permitted due to the use of signs warning of junctions, etc. This is usually a well founded assumption, with some notable exceptions such as when dealing with aerodynamic quantities in the region of shocks, structural dynamics, and progressive failure analysis (e.g. crash simulation). This can be accommodated by using multiple surrogates, patched together at discontinuities, though we will not consider this here. This is the only assumption in Kriging (see Section 3.5), making it a versatile, but complicated method. Other methods may perform better if further assumptions prove to be valid.

A second assumption is that the engineering function (though not necessarily our analyses) is smooth. Again, this is usually a perfectly valid assumption. Methods such as moving least-squares (MLS) (Section 3.3), radial basis functions (RBFs) (Section 3.4), support vector regression (SVR) (Section 3.6), and a simplified Kriging model are based on this and the continuity assumption, as too is our mental road prediction method. Although the engineering function may be smooth, our analysis of it may not be. The smoothness assumption may then require 'noise' in the observed data, be it random physical error or computational error appearing as noise, to be filtered.

Further assumptions can be made as to the actual shape of the function itself, e.g. by applying a polynomial regression (see Section 3.2). We know of many engineering quantities that obey such forms (within certain bounds). For example stress/strain is often linear and drag/velocity quadratic. Clearly assumptions about the shape of the function can be useful, but may be unfounded in many problems. No doubt many road accidents have occurred when a bend suddenly tightened in front of a driver who wrongly assumed a circular turn was ahead.

It is worth bearing in mind what we want from our surrogate. Naturally we want an accurate prediction of the function landscape we are trying to emulate and, moreover, in the context of surrogate-based optimization, we want this prediction to be most accurate in the region of the optimum. In this section we will only be considering how to produce a prediction and will look at enhancing the accuracy in the region of the optimum in the remaining sections. As mentioned at the beginning of this paper, we will not be reviewing methods for surrogate model selection and validation. We shall though consider this topic in our final discussion section, and will now briefly describe *cross-validation*—an important and commonly used generalization error estimator.

3.1. Cross-validation

To compute the cross-validation error, the surrogate model training data are split (randomly) into q roughly equal subsets. Each of these subsets is removed in turn from the complete training data and the model is fitted to the remaining data. At each stage the removed subset is predicted using the model which has been fitted to the remaining data. When all subsets have been removed, n predictions $(\hat{y}^{(1)}, \hat{y}^{(2)}, \ldots, \hat{y}^{(n)})$ of the n observed data points $(y^{(1)}, y^{(2)}, \ldots, y^{(n)})$ will have been calculated. The cross-validation error is then calculated as

$$\varepsilon_{cv} = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2. \tag{1}$$

With $q = n$, an almost unbiased error estimate can be obtained, but one whose variance can be very high. Hastie et al. [9] suggest using somewhat larger subsets, with $q = 5$ or 10.

As will be seen in the following sections, the cross-validation error can be used for surrogate model parameter estimation, model selection and validation when it is too costly to employ a separate validation data set.

3.2. Polynomials

Although fast being replaced by RBF approaches, the classic polynomial response surface model (RSM) is the original and still, probably, the most widely used form of surrogate model in engineering design. We will not dwell for too long on the subject of polynomials as this ground has been covered many times before and better than we could hope to do so now. Of the various texts out there, one of the most popular is that by Box and Draper [10].

A polynomial approximation of order m of a function f of dimension $k=1$, can be written as

$$\widehat{y}(x,m,\mathbf{a}) = a_0 + a_1x + a_2x^2 + \cdots + a_mx^m = \sum_{i=0}^{m} a_i x^{(i)}. \tag{2}$$

We estimate $\mathbf{a} = \{a_0, a_1, \ldots, a_m\}^{\mathrm{T}}$ through a least squares solution of $\mathbf{\Phi a} = \mathbf{y}$, where $\mathbf{\Phi}$ is the Vandermonde matrix:

$$\mathbf{\Phi} = \begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^m \\ 1 & x_2 & x_2^2 & \ldots & x_2^m \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & x_n & x_n^2 & \ldots & x_n^m \end{pmatrix} \tag{3}$$

and $\mathbf{y}$ is the vector of observed responses. The maximum likelihood estimate of $\mathbf{a}$ is thus

$$\mathbf{a} = (\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{y}. \tag{4}$$

Estimating m is not so simple. The polynomial approximation (2) of order m of an underlying function f is similar in some ways to a Taylor series expansion of f truncated after $m+1$ terms [10]. This suggests that greater values of m (i.e. more Taylor expansion terms) will usually yield a more accurate approximation. However, the greater the number of terms, the more flexible the model becomes and we come up against the danger of over fitting any noise that may be corrupting the underlying response. Also, we run the risk of building an excessively 'snaking' polynomial with poor generalization. We can prevent this by estimating the order m through a number of different criteria [11].

One method is to hypothesise that the true function is indeed a polynomial of degree m and any deviations from this are simply normally distributed noise. If this is the case then only $a_0, a_1, \ldots, a_m$ are required and $a_{m+1}, \ldots, a_n = 0$. The null hypothesis method (see, e.g. [12, p. 254]) chooses m by minimizing

$$\sigma_m^2 = \frac{\delta_m^2}{n-m-1}, \tag{5}$$

where

$$\delta_m^2 = \sum_{i=1}^{n}\left(y^{(i)} - \sum_{i=0}^{m} a_i x^i\right)^2. \tag{6}$$

To choose m, σ_m^2 is calculated for $m = 1, 2, \ldots$ as long as there are significant decreases in σ_m^2. The smallest m beyond which there are no significant decreases in σ_m^2 is chosen.

A model with better generalization might be obtained by instead choosing m to minimize the cross-validation error (see Section 3.1 and e.g. [9]). Cross-validation can give an indication of the overall generalization quality of the polynomial. We can also estimate the average mean squared error (MSE) as

$$s_{\mathrm{avg}}^2 = \frac{(\mathbf{y}-\widehat{\mathbf{y}})^{\mathrm{T}}(\mathbf{y}-\widehat{\mathbf{y}})}{(n-m+1)}, \tag{7}$$

where $\widehat{\mathbf{y}}$ is a vector of predictions at the observed data points and n is the number of observations.

The local MSE is estimated as

$$s^2 = s_{\mathrm{avg}}^2 \mathbf{f}^{\mathrm{T}}(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi})^{-1}\mathbf{f}^{\mathrm{T}}, \tag{8}$$

where $\mathbf{f}^{\mathrm{T}}$ is the vector of functions in Eq. (2), e.g. $\{1, x, x^2, \ldots, x^m\}^{\mathrm{T}}$ for a one variable quadratic [13,14].

Polynomial surrogates are unsuitable for the non-linear, multimodal, multi-dimensional design landscapes we often encounter in engineering unless the ranges of the variables being considered are reduced, as in trust-region methods (by suitably reducing the variable ranges under study, problems can always be simplified). In high-dimensional problems it may not be possible to obtain the quantity of data required to estimate the terms of all but a low order polynomial. However, for problems with few dimensions, uni- or low-modality, and/or where data are very cheap to obtain, a polynomial surrogate may be an attractive choice. In particular, the terms of the polynomial expression obtained using one of the above methods can provide insight into the design problem, e.g. the effect of each variable in the design space can be easily identified from the coefficients. There may be scenarios when the analysis for which a surrogate is to be used in lieu of is too expensive to be searched directly, but is cheaper than the more complex methods which follow in this paper. Here, a quick polynomial surrogate may find its niche.

When dealing with results from deterministic computer experiments, as so often is the case in design optimization, the premise that the error between the polynomial model and the data is independently randomly distributed (which the least-squares estimation in (4) is based upon) is completely false. While assuming independently randomly distributed error is often useful in laboratory experiments, for a surrogate based on deterministic experiments the error is in fact entirely modelling error and can be eliminated by interpolating rather than regressing the data. Data from computer experiments can, however, *appear* to be corrupted by an element of random error, or 'noise'. This can, for example, manifest via errors due to the finite discretization of governing equations over a computational mesh. In such cases we may again wish to assume the error between the data and the surrogate does to some extent have a random element. The remaining methods in this paper allow the degree of regression to be controlled in some way.

3.3. *Moving least-squares*

The method of MLS [15,16] allows us to build a standard polynomial regression, an interpolation, or something somewhere between the two. MLS is an embellishment of the weighted least-squares (WLS) approach [17]. WLS recognises that all $\{y^{(i)}, \mathbf{x}^{(i)}\}$ pairs may not by equally important in estimating the polynomial coefficients. To this end, each observation is given a weighting $w^{(i)} \geqslant 0$, defining the relative importance of $\{y^{(i)}, \mathbf{x}^{(i)}\}$. With $w^{(i)} = 0$ the observation is neglected in the fitting. The coefficients of the WLS model are

$$\mathbf{a} = (\mathbf{\Phi}^{\mathrm{T}}\mathbf{W\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{Wy}, \tag{9}$$

where

$$\mathbf{W} = \begin{pmatrix} w^{(1)} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & w^{(n)} \end{pmatrix}. \tag{10}$$

A WLS model is still a straightforward polynomial, but with the fit biased towards points with a higher weighting. In a MLS model, the weightings are varied depending upon the distance between the point to be predicted and each observed data point. The weighting is controlled by a function which decays with increasing distance $|\mathbf{x}^{(i)} - \mathbf{x}|$. As in the RBF literature, a whole host of decay functions have been used, but a popular choice is the Gaussian function

$$w^{(i)} = \exp\left(-\frac{\sum_{j=1}^{k}(x_j^{(i)} - x_j)^2}{\sigma^2}\right) \tag{11}$$

(used, for example, by Toropov et al. [18]).

The coefficients of the MLS are found using Eq. (9) but, unlike normal least-squares and WLS, the calculation must be performed at every prediction and so the process is more computationally intensive. The cost of prediction is increased further by the need to estimate σ as well as choose the form of the underlying polynomial. The number of terms in the polynomial is no longer critical, however, since σ will, in some senses, take care of a poor choice of underlying function. This may be a considerable advantage in a multi-variable design space where the order of the polynomial is restricted by the amount of data required to estimate all the coefficients. There is a large expanse of literature on the selection of terms for reduced order models. Here we will only consider the choice of σ, which may be a nested part of the selection of terms. We can choose σ by minimizing a cross-validation error ε_{cv} (see Eq. (1)), either using a formal optimization algorithm or simply try a selection of σ's and choose the best.

Consider the one variable test function $f(x) = (6x-2)^2 \sin(12x-4)$, $x \in [0,1]$ to which normally distributed random noise has been added with standard deviation of one. A MLS surrogate is to be fitted to a sample of 21 points. Fig. 2 shows the MLS approximations found for a range of σ's. Clearly the method exhibits an attractive tradeoff between regression and interpolation and the ε_{cv} metric provides a basis for choosing the correct tradeoff—here $\sigma = 0.1$ works best.

MLS is becoming increasingly popular in the aerospace sciences. Kim et al. [19] present a derivative enhanced form of the method and MLS has been used in surrogate-based optimization by Ho et al. [20]. However, we have found no evidence of MLS being used as part of a global optimization procedure. Although Ho et al. [20] use a global algorithm (simulated annealing) to search a MLS surrogate (which is then followed by a further local direct search of the true function), the search cannot be considered global without a system of updating the surrogate to account for possible inaccuracies which may be obscuring the global optimum (see Section 4).

3.4. Radial basis functions

RBFs [21] use a weighted sum of simple functions in an attempt to emulate complicated design landscapes. Sóbester [22] uses the analogy of imitating the specific timbre of a musical instrument with a synthesizer, using a weighted combination of tones.

Consider a function f observed without error, according to the sampling plan $\mathbf{X} = \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}^{\mathrm{T}}$, yielding the responses $\mathbf{y} = \{y^{(1)}, y^{(2)}, \ldots, y^{(n)}\}^{\mathrm{T}}$. We seek a RBF approximation to $\widehat{f}$ of the fixed form

$$\widehat{f}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\psi} = \sum_{i=1}^{n_c} w_i \psi(\|\mathbf{x} - \mathbf{c}^{(i)}\|), \tag{12}$$

where $\mathbf{c}^{(i)}$ denotes the ith of n_c basis function centres and $\boldsymbol{\psi}$ is an n_c vector containing the values of the basis functions ψ themselves, evaluated at the Euclidean distances between the prediction site $\mathbf{x}$ and the centres $\mathbf{c}^{(i)}$ of the basis functions. Readers familiar with the technology of artificial neural networks will recognise this formulation as being identical to that of a single-layer neural network with radial coordinate neurons, featuring an input $\mathbf{x}$, hidden units ψ, weights $\mathbf{w}$, linear output transfer functions, and output $\widehat{f}(\mathbf{x})$.

There is one undetermined weight per basis function if fixed bases are used. Example basis functions include

- linear $\psi(r) = r$,
- cubic $\psi(r) = r^3$, and
- thin plate spline $\psi(r) = r^2 \ln r$.

More freedom to improve the generalization properties of (12)—at the expense of a more complex parameter estimation process—can be gained by using parametric basis functions, such as the

- Gaussian $\psi(r) = \mathrm{e}^{-r^2/2\sigma^2}$,
- multi-quadric $\psi(r) = (r^2 + \sigma^2)^{1/2}$, or
- inverse multi-quadric $\psi(r) = (r^2 + \sigma^2)^{-1/2}$.

Whether we choose a set of parametric basis functions or fixed ones, $\mathbf{w}$ is easy to estimate. This can be done via the interpolation condition

$$\widehat{f}(\mathbf{x}^{(j)}) = \sum_{i=1}^{n_c} w_i \psi(\|\mathbf{x}^{(j)} - \mathbf{c}^{(i)}\|) = y^{(j)}, \quad j = 1, \ldots, n. \tag{13}$$

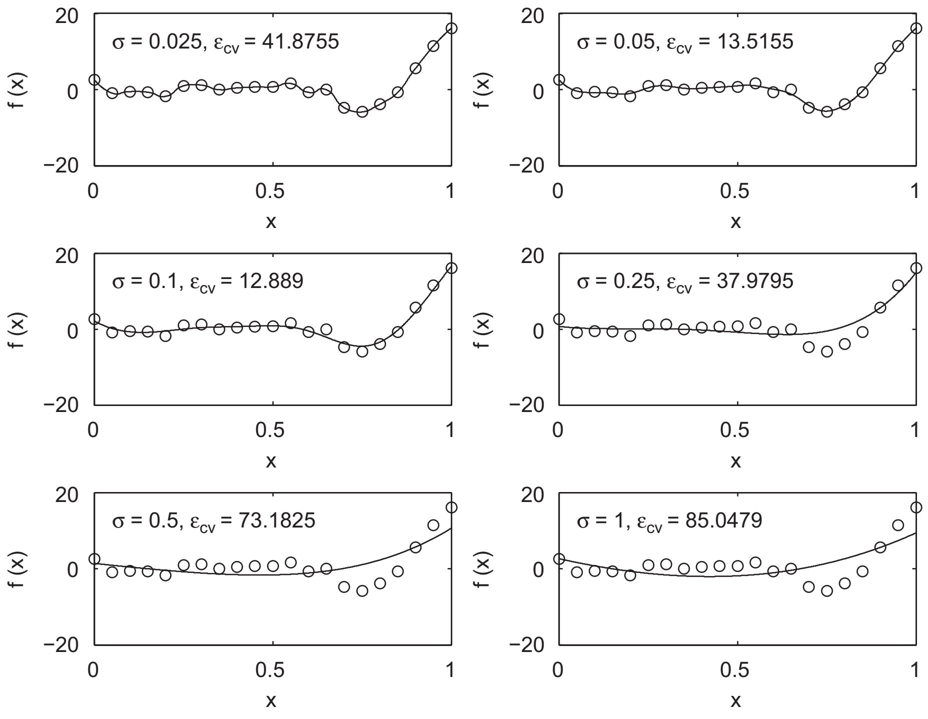

Fig. 2. MLS approximations of a noisy test function for varying σ.

While Eq. (13) is linear in terms of the basis function weights $\mathbf{w}$, the predictor $\widehat{f}$ can express highly non-linear responses. It is easy to see that one of the conditions of obtaining a unique solution is that system (13) must be 'square', that is, $n_c = n$. It simplifies things if the bases actually coincide with the data points, that is $\mathbf{c}^{(i)} = \mathbf{x}^{(i)}$, $\forall i = 1, \ldots, n$, which leads to the matrix equation

$$\mathbf{\Psi w} = \mathbf{y}, \tag{14}$$

where $\mathbf{\Psi}$ denotes the so-called *Gram matrix* and it is defined as $\mathbf{\Psi}_{i,j} = \psi(\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|)$, $i,j = 1, \ldots, n$. The fundamental step of the parameter estimation process is therefore the computation of $\mathbf{w} = \mathbf{\Psi}^{-1}\mathbf{y}$ and this is where the choice of basis function can have an important effect. For example, it can be shown that, under certain assumptions, Gaussian and inverse multi-quadric basis functions always lead to a symmetric positive definite Gram matrix [23], ensuring safe computation of $\mathbf{w}$ via Cholesky factorization—one reason for the popularity of these basis functions. Theoretically, other bases can also be modified to exhibit this property through the addition of a polynomial term (see, e.g. [24]).

Beyond determining $\mathbf{w}$, there is, of course, the additional task of estimating any other parameters introduced via the basis functions. A typical example is the σ of the Gaussian basis function, usually taken to be the same for all basis functions in all variables.

While the correct choice of $\mathbf{w}$ will make sure that the approximation can reproduce the training data, the correct estimation of these additional parameters will enable us to minimize the (estimated) generalization error of the model. This optimization step—say, the minimization of a cross-validation error estimate—can be performed at the top level, while the determination of $\mathbf{w}$ can be integrated into the process at the lower level, once for each candidate value of the parameter(s).

We have already indicated that the guarantee of a positive definite $\mathbf{\Phi}$ is one of the advantages of Gaussian RBFs. They also possess another desirable feature: we can estimate their prediction error at any $\mathbf{x}$ in the design space as

$$s^2(\mathbf{x}) = 1 - \boldsymbol{\psi}^{\mathrm{T}}\mathbf{\Psi}^{-1}\boldsymbol{\psi} \tag{15}$$

(see, e.g. [25] for further details and a derivation).

Such error estimates are invaluable in formulating infill criteria for use in surrogate-based optimization, which we will look at in subsequent sections.

3.4.1. RBF models of noisy data

If the responses $\mathbf{y} = \{y^{(1)}, y^{(2)}, \ldots, y^{(n)}\}^{\mathrm{T}}$ are corrupted by noise, the above equations may yield a model that overfits the data, that is, it does not discriminate between the underlying response and the noise. Perhaps the easiest way around this is the introduction of added model flexibility in the form of the *regularization parameter* λ [26]. This is added to the main diagonal of the Gram matrix. As a result, the approximation will no longer pass through the training points and $\mathbf{w}$ will be the least-squares solution of

$$\mathbf{w} = (\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{y}, \tag{16}$$

where $\mathbf{I}$ is an $n \times n$ identity matrix. λ should, ideally, be set to the variance of the noise in the response data $\mathbf{y}$ [24], but since we usually do not know that, we are left with the option of simply adding it to the list of parameters that need to be estimated.

Another means of constructing a regression model through noisy data using RBFs is to reduce the number of bases. This can be done using the SVR method which we will look at in Section 3.6. While SVR is perhaps the most elegant basis function selection method, a simpler way is to use forward selection (see, e.g. [27]). With $2^{n_c} - 1$ subsets of the n available centres to choose from, a truly optimal subset is unlikely to be found and so greedy forward selection algorithms which add one optimal centre at a time are often employed. Starting from an empty subset, the basis function which most reduces some error metric is chosen from n possible basis functions (centred at observed data points). The process is continued, adding one basis function at a time, until there is no significant decrease in the error metric. An improved subset might be achieved by using an exchange algorithm [28] to choose the subset at each stage of the forward selection process. The forward selection process is analogous to the null hypothesis method for polynomial fitting, but here we minimize $\delta^2/(n - n_c)$ [29].

3.5. Kriging

An increasingly popular basis function—so much so that we have given it its own section—is that used in Kriging[1]:

$$\psi^{(i)} = \mathrm{cor}[Y(\mathbf{x}^{(i)}), Y(\mathbf{x})] = \exp\left(-\sum_{j=1}^{k} \theta_j |x_j^{(i)} - x_j|^{p_j}\right), \tag{17}$$

where $Y(\cdot)$ are random variables which, rather counter intuitively, we assume the observed responses to be. Eq. (17) clearly has similarities with the Gaussian basis function in the previous section. Here there are more parameters: the variance of the basis function can be controlled in each of the k dimensions of the design space by θ_j and, instead of being fixed at 2, the exponent can be varied, again in each dimension, by p_j. These additional parameters of Kriging naturally lead to lengthier model training times, but this is mitigated by the possibility of improved accuracy in the surrogate. A common thread throughout the world of surrogate modelling is that we make assumptions as to the nature of the landscape we are emulating. Kriging is the least assuming method in this paper, in terms of the range of function forms it can emulate, and it is for this reason that it is so effective. Due to the expense of estimating the θ_j and p_j parameters, the method is of most use when the true function is particularly computationally intensive, e.g. a CFD-based calculation. The regurgitation of the derivation of the Kriging model is becoming somewhat trite, so we will satisfy ourselves with quoting the key equations and giving a few insights. We find the most useful derivation to be that in Jones [32]. There is an extensive section on Kriging, including *Matlab* code, in Forrester et al. [8].

The unknown parameters θ_j and p_j are chosen as MLEs or, to be precise, we maximize the natural logarithm of the likelihood with constant terms removed—the *concentrated ln-likelihood function*:

$$\ln(L) \approx -\frac{n}{2}\ln(\widehat{\sigma}^2) - \frac{1}{2}\ln(|\mathbf{\Psi}|), \tag{18}$$

where

$$\widehat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{1}\mu)^{\mathrm{T}}\mathbf{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\mu)}{n}, \tag{19}$$

and $\mathbf{\Psi}$ is an $n \times n$ matrix of correlations between the sample data, with each element given by Eq. (17). It is the maximization of (30)

[1] Matheron [30] coined the term *Krigeage*, in honour of the South African mining engineer Danie Krige, who first developed the method we now call *Kriging* [31]. Kriging made its way into engineering design following the work of Sacks et al. [3], who applied the method to the approximation of computer experiments. Krige's research into the application of mathematical statistics in ore valuation started during his time in the Government Mining Engineer's office and was based on very practical analyses of the frequency distributions of gold sampling values and of the correlation patterns between the individual sample values; also between the grade estimates of ore blocks based on limited sampling of the block perimeters and the subsequent sampling results from inside the ore blocks as they were being mined out. These distribution and correlation models led directly to the development of useful spatial patterns for the data and the implementation of the geostatistical Kriging and simulation techniques now in extensive use, particularly in mining circles worldwide.

that lies at the heart of the computational expense of the Kriging technique. Much research effort is directed at devising suitable training strategies and reducing the expense of the multiple matrix inversions required (see, e.g., [33,34]). Still though, this parameter estimation stage limits the method to problems of low dimensionality, with k usually limited to around 20, depending on the expense of the analyses the Kriging model is to be used in lieu of.

A by product of the parameter estimation is the insight into the design landscape we can obtain from the MLEs of θ_j and p_j. Fig. 3 shows how $\exp(-|x_j^{(i)} - x_j|^{p_j})$ varies with the separation between the points. The correlation is intuitive insofar as when the two points move close together, $x_j^{(i)} - x_j \to 0$, $\exp(-|x_j^{(i)} - x_j|^{p_j}) \to 1$ (the points show very close correlation and $Y(\mathbf{x}_j^{(i)}) = Y(\mathbf{x}_j)$) and when the points move apart, $x_j^{(i)} - x_j \to \infty$, $\exp -|x_j^{(i)} - x_j|^{p_j} \to 0$ (the points have no correlation). Three different correlations are shown in Fig. 3: $p_j = 0.1, 1$, and 2. It is clear how this 'smoothness' parameter affects the correlation, with $p_j = 2$ we have a smooth correlation with continuous gradient through $x_j^{(i)} - x_j = 0$. Reducing p_j increases the rate at which the correlation initially drops as $|x_j^{(i)} - x_j|$ increases. With a very low value of $p_j = 0.1$, we are essentially saying that there is no immediate correlation between the two points and there is a near discontinuity between $Y(\mathbf{x}_j^{(i)})$ and $Y(\mathbf{x}_j)$.

Fig. 4 shows how the choice of θ_j affects the correlation. It is essentially a width parameter which affects how far a sample point's influence extends. A low θ_j means that all points will have a high correlation, with $Y(x_j)$ being similar across our sample, while a high θ_j means that there is a significant difference between the $Y(x_j)$'s. θ_j can therefore be considered as a measure of how 'active' the function we are approximating is. Considering the 'activity' parameter θ_j in this way is helpful in high-dimensional problems where it is difficult to visualize the design landscape and the effect of the variables is unknown. By examining the elements of $\boldsymbol{\theta}$, and providing that suitable scaling of the design variables is in use, one can determine which are the most important variables and perhaps eliminate unimportant variables from future searches [8,35].

With the parameters estimated, we can make function predictions at an unknown $\mathbf{x}$ using

$$\hat{y}(\mathbf{x}) = \hat{\mu} + \boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}), \tag{20}$$

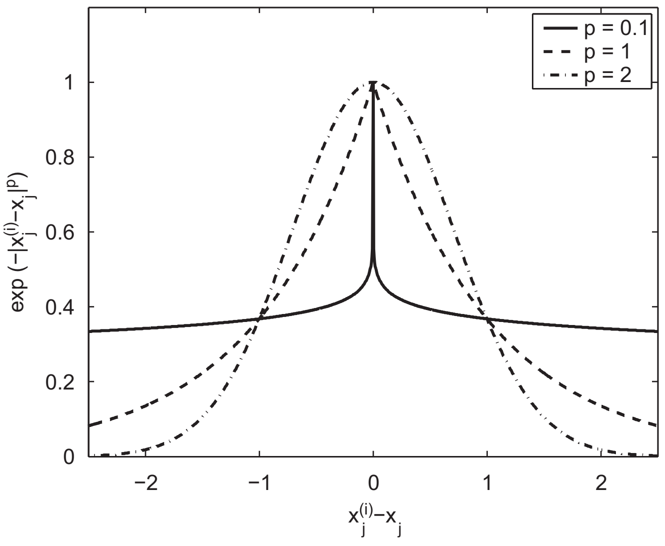

Fig. 3. Correlations with varying **p**.

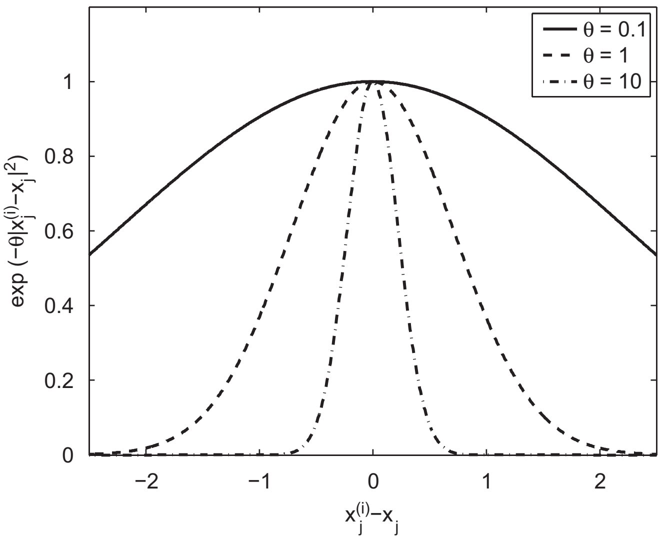

Fig. 4. Correlations with varying $\boldsymbol{\theta}$.

where

$$\hat{\mu} = \frac{\mathbf{1}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{y}}{\mathbf{1}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{1}}. \tag{21}$$

The predictive power of Kriging is rather impressive, as shown by the prediction of the popular Branin test function based on 20 observations shown in Fig. 5. This demonstration does, however, come with the warning that engineering functions are often not so smooth, noise free and predictable.

Along with other Gaussian process based models, one of the key benefits of Kriging is the provision of an estimated error in its predictions. The estimated MSE for a Kriging model is

$$s^2(\mathbf{x}) = \sigma^2\left[1 - \boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\boldsymbol{\psi} + \frac{1 - \mathbf{1}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\boldsymbol{\psi}}{\mathbf{1}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{1}}\right] \tag{22}$$

(see [3] for a derivation). It is because of its error estimates that we favour Kriging for use in surrogate-based optimization and Sections 4–6 will make extensive use of it.

The above equations are categorized as *ordinary* Kriging. This is the most popular incarnation of Kriging in the engineering sciences. There is also *universal*, *co-* and, more recently, *blind* Kriging. We will look at co-Kriging in Section 3.7.

3.5.1. *Universal Kriging*

In universal Kriging [36] the mean term is now some function of $\mathbf{x}$:

$$\hat{\mu} = \hat{\mu}(\mathbf{x}) = \sum_{i=0}^{m} \mu_i v_i(\mathbf{x}), \tag{23}$$

where the v_i's are some known functions and the μ_i's are unknown parameters. Usually $\hat{\mu}(\mathbf{x})$ takes the form of a low-order polynomial regression. The idea is that $\hat{\mu}(\mathbf{x})$ captures known trends in the data and basis functions added to this will fine-tune the model, thus giving better accuracy than ordinary Kriging where a constant $\hat{\mu}$ is used. However, we do not usually have *a priori* knowledge of the trends in the data and specifying them may introduce inaccuracies. Hence the popularity of ordinary Kriging.

3.5.2. *Blind Kriging*

Blind Kriging is a method by which the v_i's are identified through some data-analytic procedures. Hopefully, if the underlying trends can be identified, the ensuing model will be more

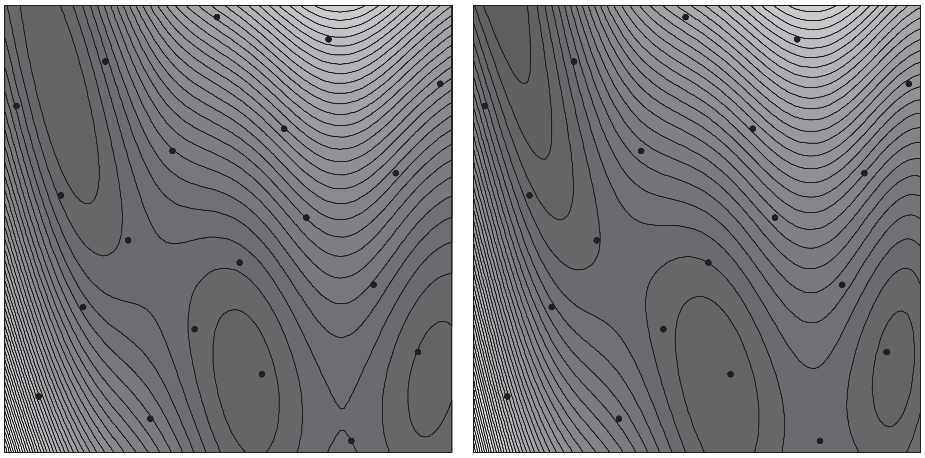

Fig. 5. The true Branin function (left) compared with a Kriging prediction based on 20 sample points (right), with $MSE = 9.30$.

accurate than ordinary Kriging. Joseph et al. [37] have certainly shown this to be the case for their engineering design examples. We will outline the process of building a blind Kriging prediction and leave the reader to consult Joseph et al. [37] for more details (our description is drawn from this reference). The above reference suggests identifying the v's through a Bayesian forward selection technique [38] and uses candidate variables of linear effects, quadratic effects, and two-factor interactions. The two-factor interactions are linear-by-linear, linear-by-quadratic, quadratic-by-linear, and quadratic-by-quadratic. This gives a total of $2k^2$ candidate variables, plus the mean term.

The linear and quadratic effects can be defined using orthogonal polynomial coding [39] and, with variables scaled $\in [0,1]$, are given by

$$x_{lin,j} = \frac{\sqrt{3}}{\sqrt{2}}2(x_j - 0.5) \quad \text{and}$$
$$x_{quad,j} = \frac{1}{\sqrt{2}}[12(x_j - 0.5)^2 - 2], \tag{24}$$

for $j = 1, 2, \ldots, k$.

To find the most important effect we need to find the maximum of the vector

$$\widehat{\boldsymbol{\beta}} = \mathbf{R}\mathbf{U}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{V}_m\widehat{\boldsymbol{\mu}}_m), \tag{25}$$

where $\mathbf{R}$ is a $(2k^2+1)\times(2k^2+1)$ diagonal matrix:

$$\mathbf{R} = \mathrm{diag}(1, r_{lin,1}, r_{quad,1}, r_{lin,2}, \ldots, r_{quad,k-1}r_{quad,k}), \tag{26}$$

where

$$r_{lin,j} = \frac{3 - 3\psi_j(1)}{3 + 4\psi_j(0.5) + 2\psi_j(1)} \quad \text{and}$$
$$r_{quad,j} = \frac{3 - 4\psi_j(0.5) + \psi_j(1)}{3 + 4\psi_j(0.5) + 2\psi_j(1)}. \tag{27}$$

$\mathbf{U}$ is an $n \times (2k^2+1)$ matrix whose first column is $\mathbf{1}$, with subsequent columns given by the interactions of the sample data:

$$\mathbf{W} = \begin{pmatrix} 1 & x_{lin,1}(x_1^{(1)}) & x_{quad,1}(x_1^{(1)}) & x_{lin,2}(x_2^{(1)}) & \cdots & x_{quad,k-1}(x_{k-1}^{(1)})x_{quad,k}(x_k^{(1)}) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{lin,1}(x_1^{(n)}) & x_{quad,1}(x_1^{(n)}) & x_{lin,2}(x_2^{(n)}) & \cdots & x_{quad,k-1}(x_{k-1}^{(n)})x_{quad,k}(x_k^{(n)}) \end{pmatrix}. \tag{28}$$

$\widehat{\boldsymbol{\mu}}_m$ is given by

$$\widehat{\boldsymbol{\mu}}_m = (\mathbf{V}_m^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{V}_m)^{-1}(\mathbf{V}_m^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{y}), \tag{29}$$

where $\mathbf{V}_m$ is an $n \times m$ matrix containing the m interactions which have been determined.

To build the blind Kriging model, first the parameters of the Kriging correlation function $\boldsymbol{\theta}$ and $\mathbf{p}$ are estimated as per ordinary Kriging. Then we compute $\widehat{\boldsymbol{\beta}}$ from Eq. (25) using $\mathbf{V}_0 = \mathbf{1}$ and $\widehat{\boldsymbol{\mu}}_0 = \widehat{\mu}$ (from ordinary Kriging). Now set $m = 1$ and choose v_m as the interaction corresponding to the maximum value of $\widehat{\boldsymbol{\beta}}$. We again compute $\widehat{\boldsymbol{\beta}}$ and now $\mathbf{V}_m$ is an $n \times (m+1)$ matrix whose first column is $\mathbf{1}$ and mth column is the column of $\mathbf{U}$ corresponding to the index of the maximum value of $\widehat{\boldsymbol{\beta}}$. $\widehat{\boldsymbol{\mu}}_m$ is found from Eq. (29).

The blind Kriging parameters $\boldsymbol{\theta}$ and $\mathbf{p}$ can now be estimated by maximizing the concentrated ln-likelihood

$$\ln(L) \approx -\frac{n}{2}\ln(\widehat{\sigma}_m^2) - \frac{1}{2}\ln(|\boldsymbol{\Psi}|), \tag{30}$$

where

$$\widehat{\sigma}_m^2 = \frac{(\mathbf{y} - \mathbf{V}_m\widehat{\boldsymbol{\mu}}_m)^{\mathrm{T}}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{V}_m\widehat{\boldsymbol{\mu}}_m)}{n}. \tag{31}$$

The blind Kriging predictor is

$$\widehat{y}(\mathbf{x}) = \boldsymbol{v}(\mathbf{x})^{\mathrm{T}}\widehat{\boldsymbol{\mu}}_m + \boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{V}_m\widehat{\boldsymbol{\mu}}_m), \tag{32}$$

where $\boldsymbol{v}(\mathbf{x})$ is an $m+1$ element column vector of interactions for the point to be predicted.

Predictions can now be used to calculate a cross-validation error and the above process iterated to reduce this error up to $m = 2k^2$ times. Joseph et al. [37] stop iterating when the cross-validation error begins to rise consistently and choose the m variables corresponding to the smallest error. They also note that it is not necessary to estimate $\boldsymbol{\theta}$ and $\mathbf{p}$ at every step, just the first and the last.

The ordinary Kriging prediction in Fig. 5 has $MSE = 9.30$ based on a 101×101 grid of points. Following the above procedure for building a blind Kriging model leads to the prediction in Fig. 6 with $MSE = 0.56$. The variables chosen using the Bayesian forward selection process were (in order of selection) $v_1 = x_{lin,1}x_{lin,2}$, $v_2 = x_{quad,1}$, $v_3 = x_{quad,2}$ and $v_4 = x_{lin,1}x_{quad,2}$. The addition of these variables affects the prediction accuracy as shown in Fig. 7. While the Branin prediction in Fig. 5 is impressive, the improvements brought about by the additional terms in the underlying function used in blind Kriging are very promising. We use the cautious term 'promising' because here we are looking at an analytical test function in just two dimensions, with a large quantity of observed data. In this case, and in Joseph et al. [37] (who use a small amount of data in high-dimensional, non-analytical problems), blind Kriging performs very well, but we would like to see it applied to more problems before we commit ourselves entirely. One should also bear in mind that the blind Kriging process is

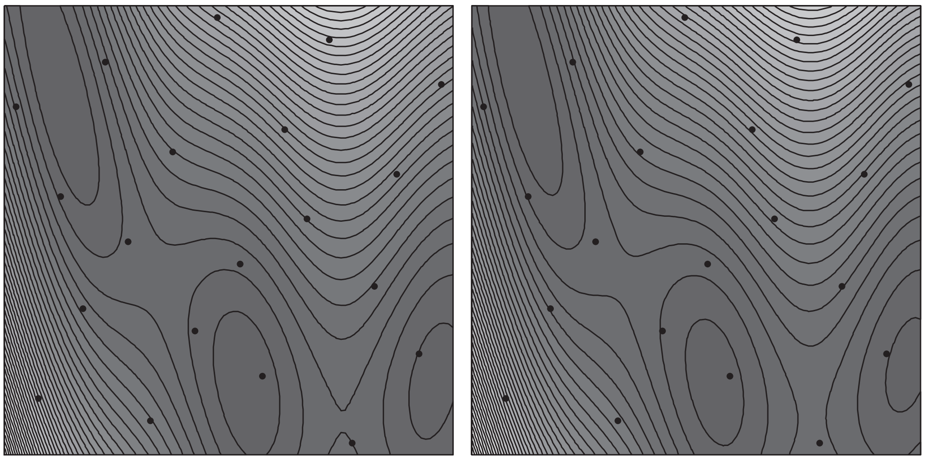

Fig. 6. The true Branin function (left) compared with a blind Kriging prediction based on 20 sample points (right), with $MSE = 0.56$.

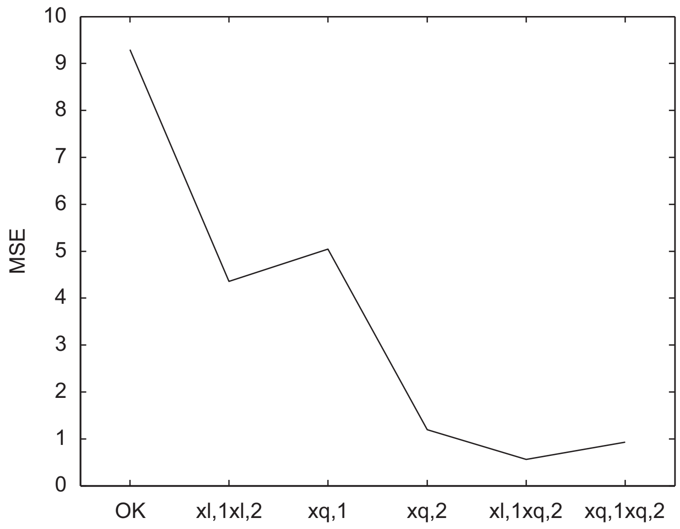

Fig. 7. MSE in blind Kriging prediction as variables are added (OK indicates ordinary Kriging).

more computationally expensive and this may outweigh increased accuracy.

3.5.3. *Kriging with noisy data*

In the same way as for an RBF prediction, a Kriging model can be allowed to regress the data by adding a regularization constant to the diagonal of the correlation matrix. The addition of this constant alters the error estimation and we need to use an error estimate in keeping with the origins of the observed data. The modelling error and the errors due to noise are given by

$$s_r^2(\mathbf{x}_{n+1}) = \hat{\sigma}_r^2\left[1 + \lambda - \boldsymbol{\psi}^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\mathbf{I})^{-1}\boldsymbol{\psi} + \frac{1 - \mathbf{1}^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\mathbf{I})^{-1}\boldsymbol{\psi}}{\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\mathbf{I})^{-1}\mathbf{1}}\right], \quad (33)$$

where [40]

$$\hat{\sigma}_r^2 = (\mathbf{y} - \mathbf{1}\hat{\mu}_r)^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\mathbf{I})^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}_r)/n. \quad (34)$$

This error estimate is appropriate when the data contain error from physical experiments.

We can express the modelling error only by using the variance

$$\hat{\sigma}_{ri}^2 = \frac{(\mathbf{y} - \mathbf{1}\hat{\mu})^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\mathbf{I})^{-1}\boldsymbol{\Psi}(\boldsymbol{\Psi} + \lambda\mathbf{I})^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu})}{n} \quad (35)$$

in the interpolating error equation (22) [41]. This error estimate is appropriate when the data contain 'noise' from computer experiments. It is important to use the appropriate error estimate when formulating the infill criteria in Section 4.

3.6. *Support vector regression*

SVR comes from the theory of support vector machines (SVM), which were developed at AT&T Bell Laboratories in the 1990s [42]. In a surrogate-based engineering design optimization context, it is perhaps more appropriate to consider SVR as an extension to RBF methods rather than SVMs, and we will do just that.

The key attribute of SVR is that it allows us to specify or calculate a margin (ε) within which we are willing to accept errors in the sample data without them affecting the surrogate prediction. This may be useful if our sample data have an element of random error due to, for example, finite mesh size, since through a mesh sensitivity study we could calculate a suitable value for ε. If the data are derived from a physical experiment, the accuracy of the measurements taken could be used to specify ε. To demonstrate this we have sampled our noisy one-dimensional test function at 21 evenly spaced points. Since we know the standard deviation of the noise is one, we have chosen $\varepsilon = 1$. The resulting SVR is shown in Fig. 8. The sample points which lie within the $\pm\varepsilon$ band (known as the ε-tube) are ignored, with the predictor being defined entirely by those which lie on or outside this region: the *support vectors*.

The basic form of the SVR prediction is the familiar sum of basis functions $\psi^{(i)}$, with weightings $w^{(i)}$, added to a base term μ; all calculated in different ways to their counterparts in the RBF and Kriging literature, yet contributing to the prediction in the same way:

$$\hat{f}(\mathbf{x}) = \mu + \sum_{i=1}^{n} w^{(i)}\psi(\mathbf{x}, \mathbf{x}^{(i)}). \quad (36)$$

3.6.1. *The support vector predictor*

We have not included derivations of the polynomial, RBF and Kriging models in this review, as these can be found easily throughout the engineering literature. SVR is rather new in this field and so we will devote some space to its derivation.

Following the theme of most SVM texts, we will first consider a linear regression, i.e. $\psi(\cdot) = \mathbf{x}$:

$$\hat{f}(\mathbf{x}) = \mu + \mathbf{w}^{\mathrm{T}}\mathbf{x}. \quad (37)$$

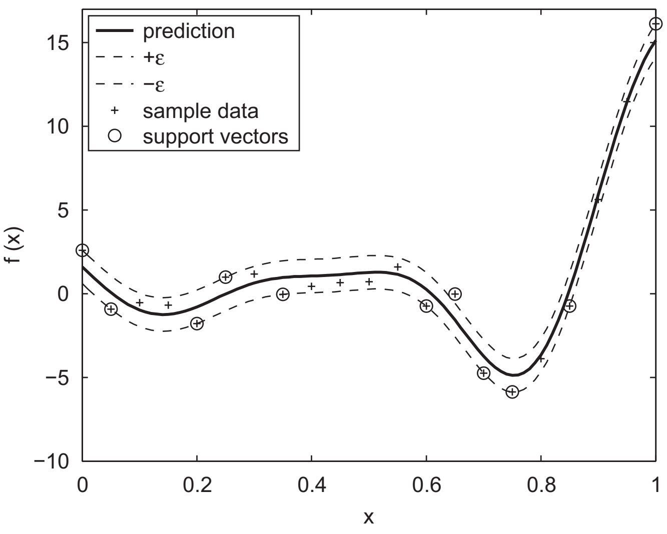

Fig. 8. A SVR prediction using a Gaussian kernel through the one-dimensional test function with added noise.

To produce a prediction which generalizes well, we wish to find a function with at most ε deviation from **y** and at the same time minimize the model complexity.[2] We can minimize the model complexity by minimizing the vector norm $|\mathbf{w}|^2$, that is, the flatter the function the simpler it is, and the more likely to generalize well. Cast as a constrained convex quadratic optimization problem, we wish to

$$\begin{aligned}&\text{minimize} \quad \tfrac{1}{2}|\mathbf{w}|^2\\ &\text{subject to} \quad -\varepsilon \leqslant y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)} - \mu \leqslant \varepsilon.\end{aligned} \tag{38}$$

Note that the constraints on this optimization problem assume that a function $\hat{f}(\mathbf{x})$ exists which approximates all $y^{(i)}$ with precision $\pm\varepsilon$. Such a solution may not actually exist and it is also likely that better predictions will be obtained if we allow for the possibility of outliers. This is achieved by introducing slack variables, ξ^+ for $f(\mathbf{x}^{(i)}) - y(\mathbf{x}^{(i)}) > \varepsilon$ and ξ^- for $y(\mathbf{x}^{(i)}) - f(\mathbf{x}^{(i)}) > \varepsilon$. We now

$$\begin{aligned}&\text{minimize} \quad \frac{1}{2}|\mathbf{w}|^2 + C\frac{1}{n}\sum_{i=1}^{n}(\xi^{+(i)} + \xi^{-(i)})\\ &\text{subject to} \quad \begin{cases} y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)} - \mu \leqslant \varepsilon + \xi^{+(i)},\\ \mathbf{w}\cdot\mathbf{x}^{(i)} + \mu - y^{(i)} \leqslant \varepsilon + \xi^{-(i)},\\ \xi^{+(i)}, \quad \xi^{-(i)} \geqslant 0.\end{cases}\end{aligned} \tag{39}$$

From Eq. (39) we see that the minimization is a tradeoff between model complexity and the degree to which errors larger than ε are tolerated. This tradeoff is governed by the user defined constant $C \geqslant 0$ ($C = 0$ would correspond to a flat function through μ). This method of tolerating errors is known as the ε-insensitive loss function and is shown in Fig. 9. Points which lie inside the ε-tube (the ε-tube is shown in Fig. 8) will have no loss associated with them, while points outside have a loss which increases linearly away from the prediction with the rate determined by C.

[2] The requirement of minimizing model complexity to improve generalization derives from Occam's Razor: *entia non-sunt multiplicanda praeter necessitatem*, which translates to 'entities should not be multiplied beyond necessity' or, in lay terms, 'all things being equal, the simplest solution tends to be the best one'. This principle is attributed to William of Ockham, a 14th century English logician and Franciscan friar.

The constrained optimization problem of Eq. (39) is solved by introducing Lagrange multipliers, $\eta^{+(i)}$, $\eta^{-(i)}$, $\alpha^{+(i)}$ and $\alpha^{-(i)}$, to give the Lagrangian

$$\begin{aligned}L = {}& \frac{1}{2}|\mathbf{w}|^2 + C\frac{1}{n}\sum_{i=1}^{n}(\xi^{+(i)} + \xi^{-(i)}) - \sum_{i=1}^{n}(\eta^{+(i)}\xi^{+(i)} + \eta^{-(i)}\xi^{-(i)})\\ & - \sum_{i=1}^{n}\alpha^{+(i)}(\varepsilon + \xi^{+(i)} - y^{(i)} + \mathbf{w}\cdot\mathbf{x}^{(i)} + \mu)\\ & - \sum_{i=1}^{n}\alpha^{-(i)}(\varepsilon + \xi^{-(i)} + y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)} - \mu),\end{aligned} \tag{40}$$

which must be minimized with respect to $\mathbf{w}$, μ, $\xi^\pm$ (the *primal variables*) and maximized with respect to $\eta^{\pm(i)}$ and $\alpha^{\pm(i)}$ (the *dual variables*), where $\eta^{\pm(i)}, \alpha^{\pm(i)} \geqslant 0$ ($\pm$ refers to both + and − variables). For active constraints, the corresponding $(\alpha^{-(i)} + \alpha^{+(i)})$ will become the support vectors (the circled points in Fig. 8) whereas for inactive constraints, $(\alpha^{-(i)} + \alpha^{+(i)}) = 0$ and the corresponding $y^{(i)}$ will be excluded from the prediction.

The minimization of L with respect to the primal variables and maximization with respect to the dual variables means we are looking for a saddle point, at which the derivatives with respect to the primal variables must vanish:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})\mathbf{x}^{(i)} = 0, \tag{41}$$

$$\frac{\partial L}{\partial \mu} = \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)}) = 0, \tag{42}$$

$$\frac{\partial L}{\partial \xi^+} = \frac{C}{n} - \alpha^{+(i)} - \eta^{-(i)} = 0, \tag{43}$$

$$\frac{\partial L}{\partial \xi^-} = \frac{C}{n} - \alpha^{-(i)} - \eta^{-(i)} = 0. \tag{44}$$

From (41) we obtain

$$\mathbf{w} = \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})\mathbf{x}^{(i)}, \tag{45}$$

and by substituting into (37) the SVR prediction is found to be

$$\hat{f}(\mathbf{x}) = \mu + \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})(\mathbf{x}^{(i)}\cdot\mathbf{x}). \tag{46}$$

The kernel trick: Until now we have considered our data **X** to exist in real coordinate space, which we will denote as $\mathscr{X} \in \mathbb{R}^k$. We wish to extend Eq. (46) beyond linear regression, to basis functions (known in the SV literature as *kernels*) which can capture more complicated landscapes. To do this we say that **x** in (46) is in *feature space*, denoted as $\mathscr{H}$, which may not coincide with $\mathbb{R}^k$. We can define a mapping between these two spaces, $\boldsymbol{\phi}: \mathscr{X} \mapsto \mathscr{H}$. We are only dealing with the inner product $\mathbf{x}\cdot\mathbf{x}$ and $\mathbf{x}\cdot\mathbf{x} = \boldsymbol{\phi}\cdot\boldsymbol{\phi}$. We can actually choose the mapping $\boldsymbol{\phi}$ and employ

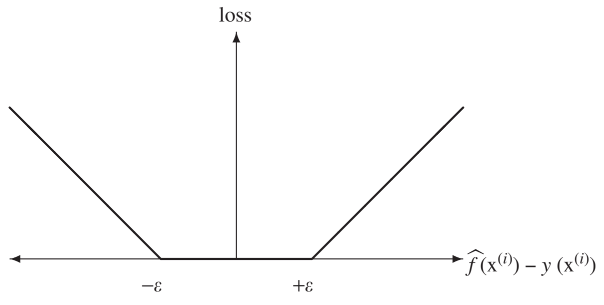

Fig. 9. ε-insensitive loss function.

different basis functions by using $\boldsymbol{\psi} = \boldsymbol{\phi} \cdot \boldsymbol{\phi}$:

$$\widehat{f}(\mathbf{x}) = \mu + \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})\psi^{(i)}. \tag{47}$$

We can do this so long as:

1. ψ is continuous,
2. ψ is symmetric, i.e. $\psi(\mathbf{x}, \mathbf{x}^{(i)}) = \psi(\mathbf{x}^{(i)}, \mathbf{x})$,
3. ψ is positive definite, which means the correlation matrix $\boldsymbol{\Psi} = \boldsymbol{\Psi}^{\mathrm{T}}$ and has non-negative eigenvalues,

that is, ψ must be a Mercer kernel. Popular choices for ψ are

$$\begin{aligned}
&\psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = (\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}) \quad \text{(linear)},\\
&\psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = (\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)})^{d} \quad (d \text{ degree homogeneous polynomial}),\\
&\psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = (\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)} + c)^{d} \quad (d \text{ degree inhomogeneous polynomial}),\\
&\psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(\frac{-|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}|^2}{\sigma^2}\right) \quad \text{(Gaussian)},\\
&\psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp\left(-\sum_{k=1}^{l}\theta_k|\mathbf{x}_k^{(i)} - \mathbf{x}_k^{(j)}|^{p_k}\right) \quad \text{(Kriging)}.
\end{aligned} \tag{48}$$

Whichever form of ψ is chosen, the method of finding the support vectors remains unchanged.

3.6.2. *Finding the support vectors*

With the kernel substitution made, the support vectors are found by substituting (41)–(44) into (40) to eliminate $\eta^{-(i)}$ and $\eta^{+(i)}$, and to finally obtain the dual variable optimization problem:

$$\begin{aligned}
&\text{maximize} \quad \begin{cases} -\dfrac{1}{2}\sum_{i,j=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})(\alpha^{+(i)} - \alpha^{-(i)})\boldsymbol{\Psi}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}), \\ -\varepsilon\dfrac{1}{2}\sum_{i=1}^{n}(\alpha^{+(i)} + \alpha^{-(i)}) + \sum_{i=1}^{n} y^{(i)}(\alpha^{+(i)} - \alpha^{-(i)}) \end{cases}\\
&\text{subject to} \quad \begin{cases} \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)}) = 0, \\ \alpha^{\pm(i)} \in [0, C/n]. \end{cases}
\end{aligned} \tag{49}$$

In order to find $\boldsymbol{\alpha}^{+}$ and $\boldsymbol{\alpha}^{-}$, rather than a combined $(\boldsymbol{\alpha}^{+} - \boldsymbol{\alpha}^{-})$, we must re-write (49) as

$$\begin{aligned}
&\text{minimize} \quad \begin{cases} \dfrac{1}{2}\begin{pmatrix} \boldsymbol{\alpha}^{+} \\ -\boldsymbol{\alpha}^{-} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \boldsymbol{\Psi} & -\boldsymbol{\Psi} \\ -\boldsymbol{\Psi} & \boldsymbol{\Psi} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}^{+} \\ -\boldsymbol{\alpha}^{-} \end{pmatrix} \\ +\begin{pmatrix} \mathbf{1}^{\mathrm{T}}\varepsilon - \mathbf{y} \\ \mathbf{1}^{\mathrm{T}}\varepsilon + \mathbf{y} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \boldsymbol{\alpha}^{+} \\ -\boldsymbol{\alpha}^{-} \end{pmatrix} \end{cases}\\
&\text{subject to} \quad \begin{cases} \mathbf{1}^{\mathrm{T}}\begin{pmatrix} \boldsymbol{\alpha}^{+} \\ -\boldsymbol{\alpha}^{-} \end{pmatrix} = 0, \\ \boldsymbol{\alpha}^{+}, \boldsymbol{\alpha}^{-} \in [\mathbf{0}, \mathbf{C}/n]. \end{cases}
\end{aligned} \tag{50}$$

Note that, as per the convention of most optimization algorithms, we have also transformed the maximization problem into a minimization.

We will not cover the formulation of quadratic programming algorithms used to solve problems such as (50). This subject is covered in detail by Schölkopf and Smola [43]. We have had success using *Matlab*'s `quadprog`.

3.6.3. *Finding μ*

In order to find the constant term μ, know as the bias, we exploit the fact that at the point of the solution of the optimization problem (49) the product between the dual variables and the constraints vanishes and see that

$$\alpha^{+(i)}(\varepsilon + \xi^{+(i)} - y^{(i)} + \mathbf{w}\boldsymbol{\psi}(\mathbf{x}^{(i)}) + \mu) = 0, \tag{51}$$

$$\alpha^{-(i)}(\varepsilon + \xi^{-(i)} + y^{(i)} - \mathbf{w}\boldsymbol{\psi}(\mathbf{x}^{(i)}) - \mu) = 0 \tag{52}$$

and

$$\xi^{+(i)}\left(\frac{C}{n} - \alpha^{+(i)}\right) = 0, \tag{53}$$

$$\xi^{-(i)}\left(\frac{C}{n} - \alpha^{-(i)}\right) = 0. \tag{54}$$

This is one of the Karush–Kuhn–Tucker conditions, which hold at the optimum (see, e.g. [43]).

From (53) and (54) we see that either $(C - \alpha^{\pm(i)}) = 0$ or $\xi^{\pm(i)} = 0$ and so all points outside of the ε-tube (where the slack variable $\xi^{\pm(i)} > 0$) must have a corresponding $\alpha^{\pm(i)} = C$. Along with Eqs. (51) and (52), and noting that $\mathbf{w}\boldsymbol{\psi}(\mathbf{x}^{(i)}) = \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})\boldsymbol{\psi}(\mathbf{x})$ this tells us that either

$$\alpha^{+(i)} = 0 \quad \text{and} \quad \mu = y^{(i)} - \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})\boldsymbol{\psi}(\mathbf{x}) + \varepsilon \quad \text{if } 0 < \alpha^{-(i)} < \frac{C}{n} \tag{55}$$

or

$$\alpha^{-(i)} = 0 \quad \text{and} \quad \mu = y^{(i)} - \sum_{i=1}^{n}(\alpha^{+(i)} - \alpha^{-(i)})\boldsymbol{\psi}(\mathbf{x}) - \varepsilon \quad \text{if } 0 < \alpha^{+(i)} < \frac{C}{n}. \tag{56}$$

Using (55) and (56) we can compute μ from one or more $\alpha^{\pm(i)}$'s which are greater than zero and less than C. More accurate results will be obtained if an $\alpha^{\pm(i)}$ not too close to these bounds is used.

3.6.4. *Choosing C and ε*

Our initial slack variable formulation (39) was a tradeoff between model complexity and the degree to which errors larger than ε are tolerated and is governed by the constant C. A small constant will lead to a flatter prediction (more emphasis on minimizing $\frac{1}{2}|\mathbf{w}|^2$), usually with fewer SVs, while a larger constant will lead to a closer fitting of the data (more emphasis on minimizing $\sum_{i=1}^{n}(\xi^{+(i)} + \xi^{-(i)})$), usually with a greater number of SVs. We wish to choose C which produces the model with the best generalization. The scaling of $\mathbf{y}$ will have an effect on the optimal value of C, so it is good practice to start by normalizing $\mathbf{y}$ to have elements between zero and one. Fig. 10 shows SVRs of the noisy one-dimensional function (this time with noise of standard deviation four), normalized between zero and one, for varying C. The Gaussian kernel variance, σ^2, has been tuned to minimize the RMSE of each prediction using 101 test points. This RMSE is displayed above each plot. It is clear that, although there is an optimum choice for C, the *exact* choice is not overly critical. It is sufficient to try a few C's of varying orders of magnitude and choose that which gives the lowest RMSE for a test data set. For small problems it is possible to obtain a more accurate C by using a simple bounded search algorithm.

Here we have prior knowledge of the amount of noise in the data and so have been able to choose ε as the standard deviation of this noise. There are many situations where we may be able to estimate the degree of noise, e.g. from a mesh dependency and solution convergence study. Situations, however, arise where the noise is an unknown quantity, e.g. a large amount of experimental data with measurements obtained by different researchers. In these situations we can calculate a value of ε which will give the most accurate prediction by using ν-SVR [43].

We have outlined the SVR formulation, but those wishing to delve further into this promising technique will find Schölkopf and Smola [43] a very useful text. A watered down version of the

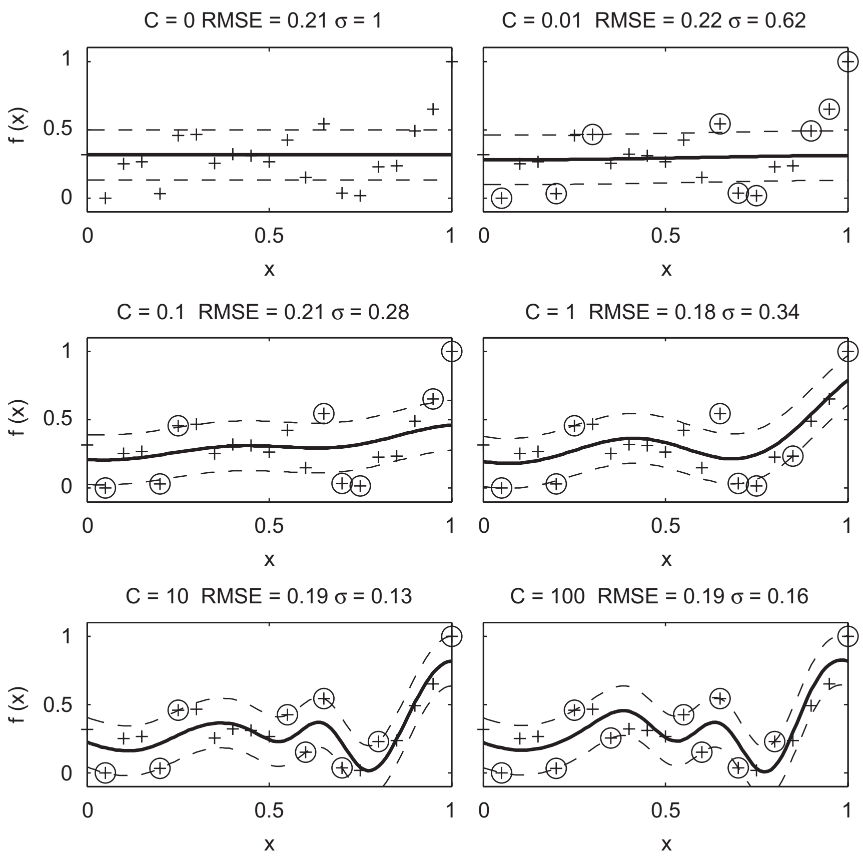

Fig. 10. SVR predictions and corresponding RMSEs for varying C ($\varepsilon = 4/\text{range}(\mathbf{y}) = 0.18(\mathbf{y})$).

SVR related part of their book appears in Smola and Schölkopf [44]. Our work here and in Forrester et al. [8] is inspired by these references. While there is no doubt that SVR is powerful method for prediction, particularly with large, high-dimensional data sets, there is only a slim body of literature detailing its use in engineering design. An example among few is Clarke et al. [45] who compare SVR with other surrogate modelling methods when predicting a variety of engineering problems. No doubt the lack of published material is partly because the method is still young, but also perhaps because the expense of engineering analyses means that we are rarely faced with the problem of very large data sets. More often than not we have a high-dimensional design space, but no possibility of filling it with large amounts of data. In such situations we habitually wish to use all our analysis data and so the SVR process of choosing a subset serves no purpose. SVR is an elegant way of producing predictions from large sets of noisy data and so may have uses in building surrogate models from, for example, extensive archives of historical data. The time required to train an SVR model is longer than other methods we have considered, due to the presence of the additional quadratic programming problem, but the accuracy and speed of prediction make it a good candidate for this scenario. Because of the lengthy training times, SVR is unlikely to find favour with those wishing to create surrogates in an ongoing optimization loop.

3.7. Enhanced modelling with additional design information

3.7.1. Exploiting gradient information

A key benefit of surrogate model based search is that the gradients of the true function(s) are not required. If gradient information is available, the designer may in fact choose to employ a localized gradient descent search of the true function with no surrogate model. However, if a global optimum is sought, the gradient information can be used to enhance the accuracy of a surrogate model of the design landscape, which can then be searched using a global optimizer.

Gradient information should only be used to enhance the surrogate if it is available cheaply, otherwise we are likely to be better off simply making more calls to the true function and building the surrogate using a larger sample. This effectively precludes the use of finite-differencing and the complex step approximation [46], although these methods could be useful for calculating a few derivatives, perhaps of particularly active variables. Most likely to be of use is gradient information found via algorithmic differentiation (AD) [47] (also known as automatic differentiation, though the term 'automatic' can instill false hope, since manual intervention is required in most cases), which requires access to the function source code, and the adjoint approach [48], which requires the creation of a whole new source code. One could also write separate code for the derivatives of a black-box code (see, e.g. [49]).

Howsoever the derivatives of the function have been found, the methods by which we incorporate them into the surrogate model are essentially the same. We mentioned earlier that Kim et al. [19] have presented a gradient enhanced MLS method. van Keulen and Vervenne [50] have presented promising results, albeit for approximating analytical test functions, for a gradient enhanced WLS method. We will examine how the information is incorporated into a Gaussian process based RBF using a form of co-Kriging [51].

RBF models are typically built from the sum of a number of basis functions centred around the sample data. The height of these functions determines the value of the prediction at the sample points (usually such that the model interpolates the data) and the width determines the rate at which the function moves away from this value. If gradient information is available at the sample locations, we can incorporate this into the model, using a second set of basis function.

These additional basis functions determine the gradient of the prediction at the sample points and the rate at which the function moves away from this gradient. The form of the basis function used to incorporate the gradient information is simply the derivative of the first n Gaussian basis functions with respect to the design variables:

$$\frac{\partial\psi^{(i)}}{\partial x_l^{(i)}} = \frac{\partial \exp(-\sum_{l=1}^{k}\theta_l(x_l^{(i)}-x_l)^2)}{\partial x_l^{(i)}} = -2\theta_l(x_l^{(i)}-x_l)\psi^{(i)}. \tag{57}$$

Fig. 11 shows how this function behaves as $x_l^{(i)} - x_l$ varies. Here we are looking at how the prediction will be distorted from the model produced by the first n basis functions. Intuitively no distortion is applied at a sampled point: we can learn no more about the value at this point than the sample data value. As we move away from the point the function pulls the prediction up or down. The θ_l hyper-parameter determines the activity of the function: a higher θ_l leads to a small region of distortion, with the value of $\partial\psi/\partial x_l$ quickly returning to zero, while a low $\partial\psi/\partial x_l$ means that a larger area is influenced by the value of the gradient in the jth direction at $x^{(i)}$.

In a gradient-enhanced RBF the correlation matrix $\mathbf{\Psi}$ must include the correlation between the data and the gradients and the gradients and themselves as well as the correlations between the data, and will be denoted by the $(k+1)n \times (k+1)n$ matrix $\dot{\mathbf{\Psi}}$. The matrix, for a k-dimensional problem ($k=1$), is constructed as follows:

$$\dot{\mathbf{\Psi}} = \begin{pmatrix} \mathbf{\Psi} & \frac{\partial\mathbf{\Psi}}{\partial x_1^{(i)}} & \frac{\partial\mathbf{\Psi}}{\partial x_2^{(i)}} & \cdots & \frac{\partial\mathbf{\Psi}}{\partial x_k^{(i)}} \\ \frac{\partial\mathbf{\Psi}}{\partial x_1^{(j)}} & \frac{\partial^2\mathbf{\Psi}}{\partial x_1^{(i)}\partial x_1^{(j)}} & \frac{\partial^2\mathbf{\Psi}}{\partial x_1^{(i)}\partial x_2^{(j)}} & \cdots & \frac{\partial^2\mathbf{\Psi}}{\partial x_1^{(i)}\partial x_k^{(j)}} \\ \frac{\partial\mathbf{\Psi}}{\partial x_2^{(j)}} & \frac{\partial^2\mathbf{\Psi}}{\partial x_1^{(j)}\partial x_2^{(i)}} & \frac{\partial^2\mathbf{\Psi}}{\partial x_2^{(i)}\partial x_2^{(j)}} & \cdots & \frac{\partial^2\mathbf{\Psi}}{\partial x_2^{(i)}\partial x_k^{(j)}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial\mathbf{\Psi}}{\partial x_k^{(j)}} & \frac{\partial^2\mathbf{\Psi}}{\partial x_1^{(j)}\partial x_k^{(i)}} & \frac{\partial^2\mathbf{\Psi}}{\partial x_2^{(j)}\partial x_k^{(i)}} & \cdots & \frac{\partial^2\mathbf{\Psi}}{\partial x_k^{(i)}\partial x_k^{(j)}} \end{pmatrix}. \tag{58}$$

The superscripts in Eq. (58) refer to which way round the subtraction is being performed when calculating the distance in the correlation ψ. This is not important when we are squaring the result but, after differentiating, sign changes will appear depending upon whether we differentiate with respect to $x^{(i)}$ or $x^{(j)}$. Using the product and chain rules, the following derivatives are obtained:

$$\frac{\partial\mathbf{\Psi}^{(i,j)}}{\partial x^{(i)}} = -2\theta(x^{(i)}-x^{(j)})\mathbf{\Psi}^{(i,j)}, \tag{59}$$

$$\frac{\partial\mathbf{\Psi}^{(i,j)}}{\partial x^{(j)}} = 2\theta(x^{(i)}-x^{(j)})\mathbf{\Psi}^{(i,j)}, \tag{60}$$

$$\frac{\partial^2\mathbf{\Psi}^{(i,j)}}{\partial x^{(i)}\partial x^{(j)}} = [2\theta - 4\theta^2(x^{(i)}-x^{(j)})^2]\mathbf{\Psi}^{(i,j)}, \tag{61}$$

$$\frac{\partial^2\mathbf{\Psi}^{(i,j)}}{\partial x_l^{(i)}\partial x_m^{(i)}} = -4\theta_l\theta_m(x_l^{(i)}-x_l^{(j)})(x_m^{(i)}-x_m^{(j)})\mathbf{\Psi}^{(i,j)}. \tag{62}$$

The $\boldsymbol{\theta}$ parameter is found by maximizing the concentrated ln-likelihood in the same manner as for a standard Gaussian RBF. Other than the above correlations, the only difference in the construction of the gradient-enhanced model is that $\mathbf{1}$ is now a $(k+1)n \times 1$ column vector of n ones followed by nk zeros. The gradient-enhanced predictor is

$$\hat{y}(\mathbf{x}^{(n+1)}) = \hat{\mu} + \dot{\boldsymbol{\psi}}^{\mathrm{T}}\dot{\mathbf{\Psi}}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}), \tag{63}$$

where

$$\dot{\boldsymbol{\psi}} = \left(\boldsymbol{\psi}, \frac{\partial\boldsymbol{\psi}}{\partial x_1}, \ldots, \frac{\partial\boldsymbol{\psi}}{\partial x_k}\right)^{\mathrm{T}}. \tag{64}$$

Fig. 12 shows a contour plot of the Branin function along with a gradient-enhanced Kriging prediction based on nine sample points. True gradients and gradients calculated using a finite difference of the gradient-enhanced Kriging prediction are also shown. The agreement between the functions and gradients is remarkable for this function, however, the method is unlikely to perform quite so well on true engineering functions.

We can take the use of gradients to the next step and include second derivatives in an Hessian-enhanced model. The basis function used to incorporate the second derivative information is the second derivative of the first n Gaussian basis functions with respect to the design variables:

$$\frac{\partial^2\psi^{(i)}}{\partial x_l^{(i)2}} = \frac{\partial^2 \exp(-\sum_{l=1}^{k}\theta_l(x_l^{(i)}-x_l)^2)}{\partial x_l^{(i)2}} = [-2\theta_l + 4\theta_l^2(x_l^{(i)}-x_l)^2]\psi^{(i)}. \tag{65}$$

Fig. 13 shows how the twice differentiated basis function behaves for varying θ.

Fig. 14 shows three predictions of our one variable test function: Kriging, gradient-enhanced Kriging and Hessian-enhanced Kriging. In our other figures showing Kriging predictions of this function based on three points we have cheated a little by constraining the θ parameter to give a good prediction. Here we have opened up the bounds on θ and the MLE actually gives a very poor prediction: because of the sparsity of data, no trend is recognised and the prediction is simply narrow bumps around a mean fit. The extra gradient information significantly improves the prediction. It should be borne in mind though that adding extra observed data instead of the expense of calculating gradients would have improved the prediction too. In high-dimensional problems a few extra observed points will not be as useful as many derivatives and it is here that cheap gradient information (e.g. from adjoint formulations) is of most use.

The nine weighted basis functions and mean used to build the prediction in Fig. 14 are shown in Fig. 15. It is clear from this figure how each type of basis function affects the prediction. The first three (ψ) are simple deviations from the mean and the second

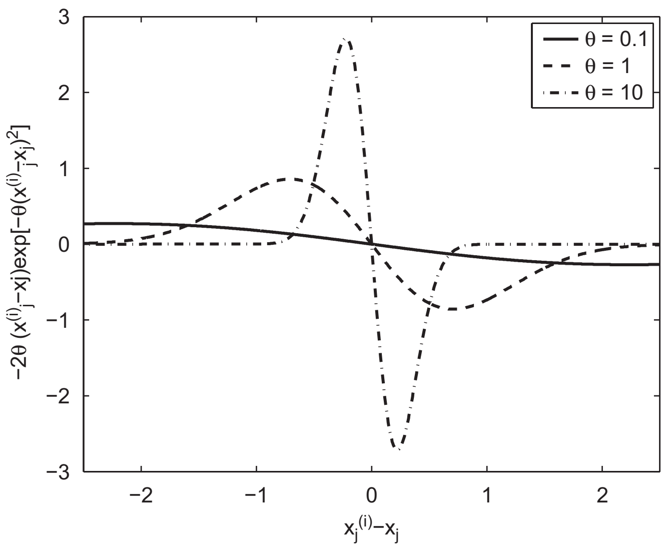

Fig. 11. Differentiated correlations for varying $\boldsymbol{\theta}$.

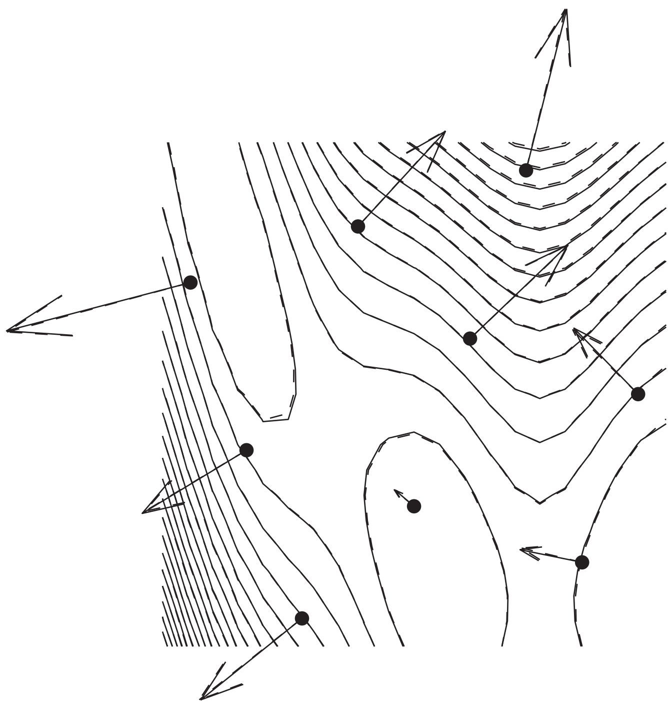

Fig. 12. Contours of the Branin function (solid) and a gradient-enhanced prediction (dashed) based on nine points (dots). True gradients (solid arrows) and gradients calculated using a finite difference of the gradient-enhanced Kriging prediction (dashed arrows) are also shown. Note that the true function and the prediction are so close that the solid contours and arrows almost completely obscure their dashed counterparts.

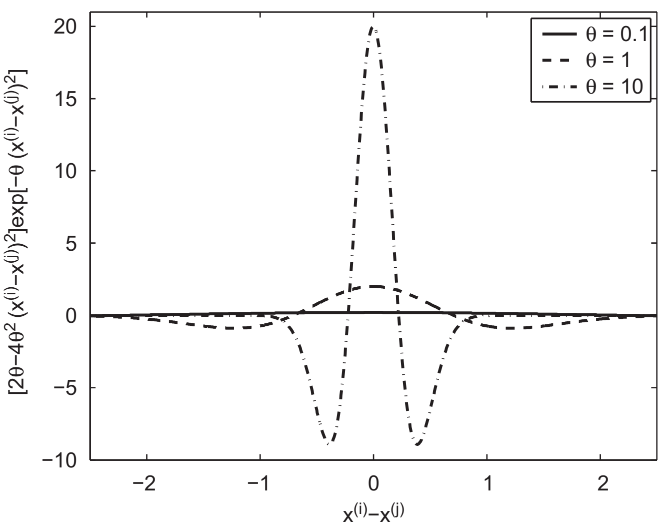

Fig. 13. Twice differentiated correlations for varying $\boldsymbol{\theta}$.

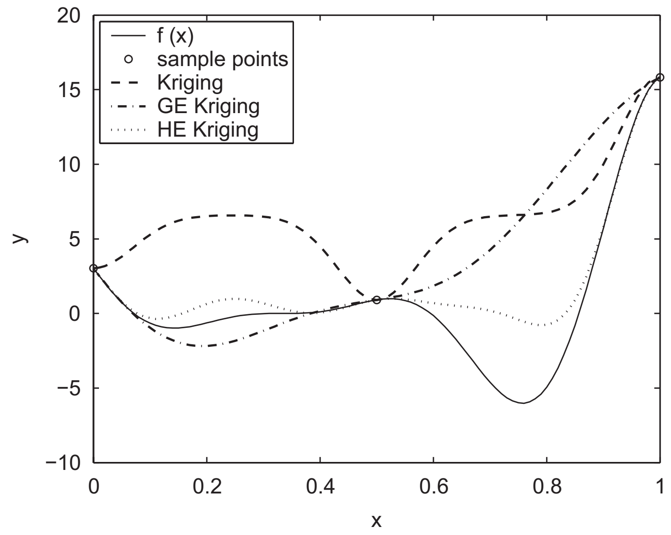

Fig. 14. Kriging, gradient enhanced Kriging, and Hessian-enhanced Kriging predictions of $f(x) = (6x-2)^2 \sin(12x-4)$ using three sample points.

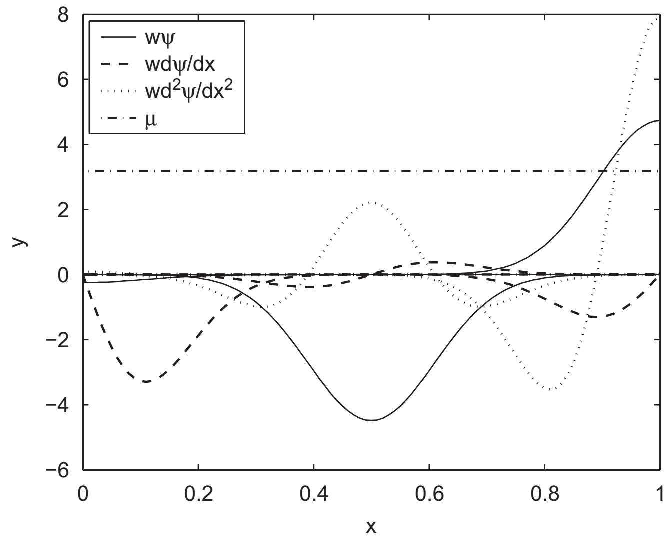

Fig. 15. The nine basis functions used to construct the Hessian-enhanced Kriging prediction, multiplied by their weights, $\mathbf{w} = \ddot{\boldsymbol{\Psi}}^{-1}(\mathbf{y} - \mathbf{1}\mu)$. These are added to the constant μ to produced the prediction in Fig. 14.

three ($\dot{\psi}$) clearly match the gradient at the sample points. Of the final three bases ($\ddot{\psi}$), the first has little effect (the gradient is near constant at this point), the second works against ψ to flatten the function, while the third adds to the curvature, resulting in the steep curve into the global minimum.

The use of derivative information adds considerable complexity to the model and the increased size of the correlation matrix leads to (very much) lengthier parameter estimation, however, it clearly leads to the possibility of building more accurate predictions. Schemes to reduce model parameter estimation times for large correlation matrices are always the target of research effort, though a panacea is yet to reveal itself. Second derivatives are not often available to the designer but, with the increasing use of automatic differentiation tools, models which can take advantage of this information may soon provide significant speed-ups compared to using additional function calls—particularly in very high-dimensional problems where adjoint approaches prove most powerful.

3.7.2. *Multi-fidelity analysis*

When additional information is available, rather than gradients of the function to be approximated, we are perhaps more likely to have available other cheaper approximations of the function. It may be, for example, that as well as using finite element analysis or computational fluid dynamics, a quick calculation can be made using empirical equations, more simple beam theory, or panel methods. In multi-fidelity (also known as variable-fidelity) surrogate-based methods a greater quantity of this cheap data may be coupled with a small amount of expensive data to enhance the accuracy of a surrogate of the expensive function. To make use of the cheap data, we must formulate some form of correction process which models the differences between the cheap and expensive function.

Although we may have many forms of analysis, let us assume for our discussion that we have just two ways of calculating the function (the methods can be extended to multiple levels of analyses). Our most accurate expensive data have values $\mathbf{y}_e$ at points $\mathbf{X}_e$ and the less accurate cheap data have values $\mathbf{y}_c$ at points $\mathbf{X}_c$. The formulation of a correction process is simplified if the expensive function sample locations coincide with a subset of the cheap sample locations ($\mathbf{X}_e \subset \mathbf{X}_c$). The correction process will usually take the form

$$\mathbf{y}_e = Z_\rho \mathbf{y}_c + Z_d. \tag{66}$$

With $Z_d = 0$, Z_ρ can take the form of any approximation model fitted to $\mathbf{y}_e/\mathbf{y}_c(\mathbf{X}_e)$. Likewise, with $Z_\rho = 1$, Z_d can take the form of an approximation fitted to $\mathbf{y}_e - \mathbf{y}_c(\mathbf{X}_e)$. These processes are then used to correct $\mathbf{y}_c$ when making predictions of the expensive function f_e. If the correction process is simpler than f_e, then we can expect predictions based on a large quantity of cheap data with a simple correction to be more accurate than predictions based on a small quantity of expensive data. This simple form of combining multi-fidelity analyses has be used by Leary et al. [52] for finite element analyses using different mesh sizes and by Forrester et al. [53] for combining CFD of varying levels of convergence.

Instead of using Z_c or Z_d, which are *output* correction processes, we can employ an *input* correction, known as *space mapping* [54]. By distorting the locations of $\mathbf{X}_c$ we can attempt to align the contours of the cheap function with those of the expensive function. If the cheap and expensive functions have similar scaling, we hope to find a mapping $\mathfrak{p}(\mathbf{X}_e)$ such that $\mathbf{y}_e(\mathbf{X}_e) \approx \mathbf{y}_c(\mathfrak{p}(\mathbf{X}_e))$. Of course the scaling of the cheap and expensive functions may be quite different and so an additional correction process from Eq. (66) may be required.

A more powerful multi-fidelity method is that of co-Kriging [36]—an enhancement to the geostatistical method of Kriging, but equally applicable as an enhancement to any parametric RBF. Co-Kriging has been used extensively outside of aerospace design. For example Hevesi et al. [55] predict average annual precipitation values near a potential nuclear waste disposal site using a sparse set of precipitation measurements from the region along with the correlated and more easily obtainable elevation map of the region. Kennedy and O'Hagan [56] apply co-Kriging to the correlation of results of computer simulations of varying fidelities and cost. Forrester et al. [57] extend the method from prediction to optimization and present the aerodynamic design of a wing using correlated empirical and panel codes. The following presentation of the co-Kriging method is based on this reference.

Using our two sets of data; cheap and expensive, we begin the co-Kriging formulation by concatenating the sample locations to give the combined set of sample points

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_c \\ \mathbf{X}_e \end{pmatrix} = \begin{pmatrix} \mathbf{x}_c^{(1)} \\ \vdots \\ \mathbf{x}_c^{(n_c)} \\ \mathbf{x}_e^{(1)} \\ \vdots \\ \mathbf{x}_e^{(n_e)} \end{pmatrix}.$$

As with Kriging, the value at a point in $\mathbf{X}$ is treated as if it were the realization of a stochastic process. For co-Kriging we therefore have the random field

$$\mathbf{Y} = \begin{pmatrix} \mathbf{Y}_c(\mathbf{X}_c) \\ \mathbf{Y}_e(\mathbf{X}_e) \end{pmatrix} = \begin{pmatrix} Y_c(\mathbf{x}_c^{(1)}) \\ \vdots \\ Y_c(\mathbf{x}_c^{(n_c)}) \\ Y_e(\mathbf{x}_e^{(1)}) \\ \vdots \\ Y_e(\mathbf{x}_e^{(n_e)}) \end{pmatrix}.$$

Here we use the *auto-regressive* model of Kennedy and O'Hagan [56] which assumes that $\text{cov}\{Y_e(\mathbf{x}^{(i)}), Y_c(\mathbf{x}) | Y_c(\mathbf{x}^{(i)})\} = 0,\ \forall \mathbf{x} \neq \mathbf{x}^{(i)}$. This means that no more can be learnt about $Y_e(\mathbf{x}^{(i)})$ from the cheaper code if the value of the expensive function at $\mathbf{x}^{(i)}$ is known (this is known as a Markov property which, in essence, says we assume that the expensive simulation is correct and any inaccuracies lie wholly in the cheaper simulation).

Gaussian processes $Z_c(\cdot)$ and $Z_e(\cdot)$ represent the local features of the cheap and expensive codes. Using the auto-regressive model we are essentially approximating the expensive code as the cheap code multiplied by a constant scaling factor ρ plus a Gaussian process $Z_d(\cdot)$ which represents the difference between $\rho Z_c(\cdot)$ and $Z_e(\cdot)$:

$$Z_e(\mathbf{x}) = \rho Z_c(\mathbf{x}) + Z_d(\mathbf{x}). \tag{67}$$

Where in Kriging we have a covariance matrix $\text{cov}\{\mathbf{Y}(\mathbf{X}), \mathbf{Y}(\mathbf{X})\} = \sigma^2 \mathbf{\Psi}(\mathbf{X}, \mathbf{X})$, we now have a covariance matrix:

$$\mathbf{C} = \begin{pmatrix} \sigma_c^2 \mathbf{\Psi}_c(\mathbf{X}_c, \mathbf{X}_c) & \rho \sigma_c^2 \mathbf{\Psi}_c(\mathbf{X}_c, \mathbf{X}_e) \\ \rho \sigma_c^2 \mathbf{\Psi}_c(\mathbf{X}_e, \mathbf{X}_c) & \rho^2 \sigma_c^2 \mathbf{\Psi}_c(\mathbf{X}_e, \mathbf{X}_e) + \sigma_d^2 \mathbf{\Psi}_d(\mathbf{X}_e, \mathbf{X}_e) \end{pmatrix}. \tag{68}$$

The notation $\mathbf{\Psi}_c(\mathbf{X}_e, \mathbf{X}_c)$, for example, denotes a matrix of correlations of the form ψ_c between the data $\mathbf{X}_e$ and $\mathbf{X}_c$.

The correlations are of the same form as Eq. (17), but there are two correlations, ψ_c and ψ_d and we therefore have more parameters to estimate: $\boldsymbol{\theta}_c$, $\boldsymbol{\theta}_d$, $\mathbf{p}_c$, $\mathbf{p}_d$ and the scaling parameter ρ. Our cheap data are considered to be independent of the expensive data and we can find MLEs for μ_c, σ_c^2, $\boldsymbol{\theta}_c$ and $\mathbf{p}_c$ by maximizing the concentrated ln-likelihood:

$$-\frac{n_c}{2} \ln(\widehat{\sigma}_c^2) - \frac{1}{2} \ln |\det(\mathbf{\Psi}_c(\mathbf{X}_c, \mathbf{X}_c))|, \tag{69}$$

where

$$\widehat{\sigma}_c^2 = (\mathbf{y}_c - \mathbf{1}\widehat{\mu}_c)^{\mathrm{T}} \mathbf{\Psi}_c(\mathbf{X}_c, \mathbf{X}_c)^{-1} (\mathbf{y}_c - \mathbf{1}\widehat{\mu}_c)/n_c. \tag{70}$$

To estimate μ_d, σ_d^2, $\boldsymbol{\theta}_d$, $\mathbf{p}_d$ and ρ, we first define

$$\mathbf{d} = \mathbf{y}_e - \rho \mathbf{y}_c(\mathbf{X}_e), \tag{71}$$

where $\mathbf{y}_c(\mathbf{X}_e)$ are the values of $\mathbf{y}_c$ at locations common to those of $\mathbf{X}_e$ (the Markov property implies that we only need to consider this data). If $\mathbf{y}_c$ is not available at $\mathbf{X}_e$ we may estimate ρ at little additional cost by using Kriging estimates $\widehat{\mathbf{y}}_c(\mathbf{X}_e)$ found from Eq. (20) using the already determined parameters $\widehat{\boldsymbol{\theta}}_c$ and $\widehat{\mathbf{p}}_c$. The concentrated ln-likelihood of the expensive data is now

$$-\frac{n_e}{2} \ln(\widehat{\sigma}_d^2) - \frac{1}{2} \ln |\det(\mathbf{\Psi}_d(\mathbf{X}_e, \mathbf{X}_e))|, \tag{72}$$

where

$$\widehat{\sigma}_d^2 = (\mathbf{d} - \mathbf{1}\widehat{\mu}_d)^{\mathrm{T}} \mathbf{\Psi}_d(\mathbf{X}_e, \mathbf{X}_e)^{-1} (\mathbf{d} - \mathbf{1}\widehat{\mu}_d)/n_e. \tag{73}$$

As with Kriging, Eqs. (69) and (72) must be maximized numerically using a suitable global search routine. Depending upon the cost of evaluating the cheap and expensive functions f_c and f_e, for very high-dimensional problems the multiple matrix inversions involved in the likelihood maximization may render the use of the co-Kriging model impractical (the size of the matrices depends directly on the quantities of data available, and the number of search steps needed in the MLE process is linked to the number of parameters being tuned). Typically a statistical model used as a surrogate will be tuned many fewer times than the number of evaluations of f_e required by a direct search. The cost of tuning the model can therefore be allowed to exceed the cost of computing f_e and still provide significant speed-up. For large k and n the time required to find MLEs can be reduced by using a constant $\theta_{c,j}$ and $\theta_{d,j}$ for all elements of $\boldsymbol{\theta}_c$ and $\boldsymbol{\theta}_d$ to simplify the maximization, though this may affect the accuracy of the approximation.

Co-Kriging predictions are given by

$$\widehat{y}_e(\mathbf{x}) = \widehat{\mu} + \mathbf{c}^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{y} - \mathbf{1}\widehat{\mu}), \tag{74}$$

where

$$\widehat{\mu} = \mathbf{1}^{\mathrm{T}}\mathbf{C}(\mathbf{X},\mathbf{X})^{-1}\mathbf{y}/\mathbf{1}^{\mathrm{T}}\mathbf{C}(\mathbf{X}_e,\mathbf{X}_e)^{-1}\mathbf{1} \tag{75}$$

and $\mathbf{c}$ is a column vector of the covariance between $\mathbf{X}$ and $\mathbf{x}$ (see [57] for a derivation).

If we make a prediction at one of our expensive points, $\mathbf{x}^{(n+1)} = \mathbf{x}_e^{(i)}$ and $\mathbf{c}$ is the $n_c + i$th column of $\mathbf{C}$, then $\mathbf{c}^{\mathrm{T}}\mathbf{C}^{-1}$ is the $n_c + i$th unit vector and $\widehat{y}_e(\mathbf{x}_e^{(i)}) = \widehat{\mu} + \mathbf{y}^{(n_c+i)} - \widehat{\mu} = y_e^{(i)}$. We see, therefore, that Eq. (74) is an interpolator of the expensive data (just like ordinary Kriging), but will in some sense regresses the cheap data unless it coincides with $\mathbf{y}_e$.

The estimated MSE in this prediction is similar to the Kriging error, and is calculated as

$$s^2(\mathbf{x}) \approx \rho^2\widehat{\sigma}_c^2 + \widehat{\sigma}_d^2 - \mathbf{c}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{c} + \frac{1 - \mathbf{1}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{c}}{\mathbf{1}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{1}}. \tag{76}$$

For $\mathbf{x}^{(n_e+1)} = \mathbf{x}_e^{(i)}$, $\mathbf{c}^{\mathrm{T}}\mathbf{C}^{-1}$ is the $n_c + i$th unit vector, $\mathbf{c}^{\mathrm{T}}\mathbf{C}^{-1}\mathbf{c} = \mathbf{c}^{(n_c+i)} = \rho_c^2\sigma_c^2 + \sigma_d^2$ and so $s^2(\mathbf{x})$ is zero (just like ordinary Kriging). For $\mathbf{X}_c \backslash \mathbf{X}_e$, $s^2(\mathbf{x}) \neq 0$ unless $\mathbf{y}_e = \mathbf{y}_c(\mathbf{X}_e)$. The error at these points is determined by the character of Z_d. If this difference between $\rho Z_c(\mathbf{X}_e)$ and $Z_e(\mathbf{X}_e)$ is simple (characterized by low $\theta_{d,j}$'s) the error will be low, whereas a more complex difference (high $\theta_{d,j}$'s) will lead to high error estimates.

As shown for Kriging in Section 3.5.3, a regression parameter can be added to the leading diagonal of the correlation matrix when noise is present. In fact, two parameters may be used: one for the cheap data and one for the expensive data.

We will recycle our simple one variable test function to demonstrate co-Kriging. Imagine that our expensive to compute data are calculated by the original function $f_e(x) = (6x - 2)^2 \sin(12x - 4)$, $x \in [0, 1]$, and a cheaper estimate of this data is given by $f_c(x) = Af_e + B(x - 0.5) + C$. We sample the design space extensively using the cheap function at $\mathbf{X}_c = \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$, but only run the expensive function at four of these points, $\mathbf{X}_e = \{0, 0.4, 0.6, 1\}$.

Fig. 16 shows the functions f_e and f_c with $A = 0.5$, $B = 10$, and $C = -5$. A Kriging prediction through $\mathbf{y}_e$ gives a poor approximation to the deliberately deceptive function, but the co-Kriging prediction lies very close to f_e, being better than both the standard Kriging model and the cheap data. Despite the considerable differences between f_e and f_c, a simple relationship has been found between the expensive and cheap data and the estimated error reduces almost to zero at $\mathbf{X}_c$ (see Fig. 17).

While we are not considering sampling techniques in this paper, the problem of choosing the n_e-element subset $\mathbf{X}_e$ of $\mathbf{X}_c$ is an unusual one and so in this case we will make an exception. As with an initial sample, we wish to cover the parameter space evenly so we turn to the Morris–Mitchell criterion [7], but this time we are dealing with a limited, discreet parameter space and thus the problem becomes a combinatorial one. Since selecting the subset that satisfies this is an *NP-complete* problem and an exhaustive search would have to examine ${}_{n_c}C_{n_e} = n_c!/n_e!(n_c - n_e)!$

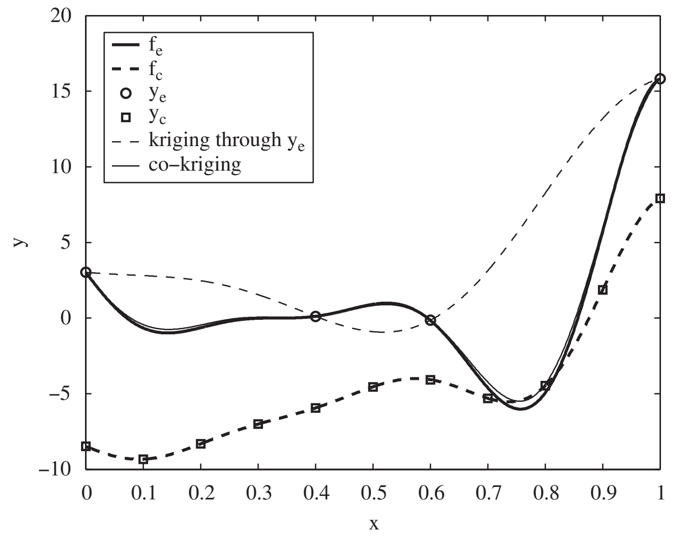

Fig. 16. A one variable co-Kriging example. The Kriging approximation using four expensive data points ($\mathbf{y}_e$) has been significantly improved using extensive sampling from the cheap function ($\mathbf{y}_c$).

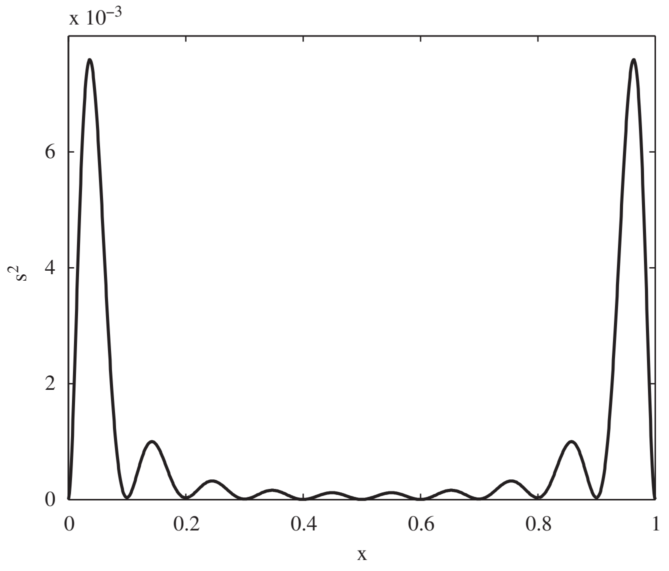

Fig. 17. Estimated error in the co-Kriging prediction in Fig. 16. The simple relationship between the data results in low error estimates at $\mathbf{X}_c$ as well as $\mathbf{X}_e$.

subsets (clearly infeasible for all but very moderate cardinalities), here we use an exchange algorithm to select $\mathbf{X}_e$ (see, e.g. [28]).

We start from a randomly selected subset $\mathbf{X}_e$ and calculate the Morris–Mitchell criterion. We then exchange the first point $\mathbf{x}_e^{(1)}$ with each of the remaining points in $\mathbf{X}_c \backslash \mathbf{X}_e$ and retain the exchange which gives the best Morris–Mitchell criterion. This process is repeated for each remaining point $\mathbf{x}_e^{(2)} \ldots \mathbf{x}_e^{(n_e)}$. A number of restarts from different initial subsets can be employed to avoid local optima. Fig. 18 shows a Morris–Mitchell optimal LH with a subset chosen using this exchange algorithm.

A rule of thumb for the number of points which should be used in the sampling plan is $n = 10k$. When using a particularly cheap analysis n_c may be rather greater than this, allowing us to build a more accurate model, and if the relationship between f_c and f_e is simple, n_e may be somewhat fewer—the advantage of the co-Kriging method.

Our choice of cheap function for the above example is somewhat contrived. For our test function the correction process

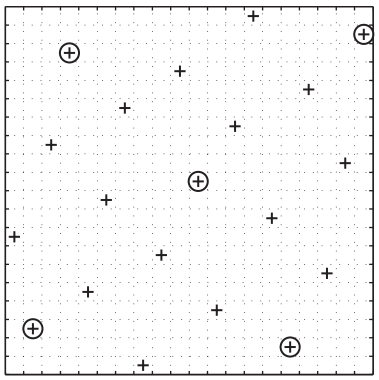

Fig. 18. A 20 point Morris–Mitchell optimal Latin hypercube (+) with a five point subset found using the exchange algorithm (∘).

$Z_d(\cdot)$ is linear. Co-Kriging will work effectively for more complex correction processes with the proviso that $Z_d(\cdot)$ must be simpler to model than $Z_e(\cdot)$. Although we have only considered combining two levels of analysis, the co-Kriging method can be extended to multiple levels by using additional ρ's and $\mathbf{d}$'s (see [56] for more details).

Although multi-level modelling can be achieved simply by combining independent surrogates of the ratios or differences between data, the co-Kriging method is more powerful, both in terms of the complexity of relationships it can handle, and its ability to provide error estimates which can be used to formulate infill criteria.

4. Infill criteria

While our surrogate is built upon assumption, our designs, of course, cannot be—the ensuing lawsuits would be too costly! Results from the surrogate must be confirmed with calls to the true function. Indeed, at any stage we take our optimum design to be the best result of the true function, not that from the surrogate. Additional calls to the true function are not only used to validate the surrogate, but also to enhance its accuracy. It is the judicious selection of new points at which to call the true function, so-called *infill points*, which represents the heart of the surrogate-based optimization process. Applying a series of infill points, based on some infill criteria, is also known as *adaptive sampling* (or *updating*), that is we are sampling the objective function in promising areas based on a constantly changing surrogate.

The success or failure of a surrogate-based optimization rests on the correct choice of model and infill criteria. Just as when choosing the model, when selecting the infill criteria we can also take short-cuts by making certain assumptions. While offering quick solutions, such short-cuts are naturally susceptible to failure. Jones [32] does an admirable job in highlighting possible avenues of failure and points towards the correct route to the global optimum. In the following sections we will give an overview of his work coupled with our own experience, and point towards some new methods which try to guarantee the eventual location of the global optimum. We should note at this point, however, that in much practical design work our aim is design improvement and often starts from a locally optimized design. Also that it is always possible to design pathological functions that will fool any optimization process except exhaustive search.

Before embarking on an infill process, we can try a simple 'trick' to improve the accuracy of the surrogate. Consider the problem of predicting stress vs. cross-sectional area. The problem, albeit already simple, could be further simplified to a linear relationship by predicting the negative of the reciprocal of the stress. Of course relationships will rarely be so straightforward, but, nonetheless, it is worth trying a few transformations (e.g., negative reciprocal, logarithm) and re-calculating a generalization error estimate to see if the accuracy of the surrogate can be improved [58].

4.1. Exploitation

4.1.1. Minimizing the predictor

The most basic assumption we can make is that the surrogate model is globally accurate and all we need to do is validate the optimum of the surrogate, having found it with large numbers of calls to a suitably robust global optimizer, by running a single true function evaluation at this point. Hopefully our assumption of global accuracy is based on some form of validation metric. Cross-validation or, ideally, tests using a separate set of data used to compute a MSE or correlation coefficient can be used to indicate the global validity of a surrogate [9]. It is, however, unlikely that the surrogate will, initially, be sufficiently accurate in the region of the optimum and so it is usual practice to apply a succession of infill points at the predicted optimum. After each infill point the surrogate is re-fitted to the data such that an interpolating model will descend into the optimum location. This method is illustrated in Figs. 19 and 20, where a function, $f(x) = (6x-2)^2\sin(12-4)$, $x \in [0,1]$, is sampled with three initial points followed by five infill points at the minimum of the prediction. The method quickly descend into a local minimum. The vague region of this minimum was indicated by the initial prediction and the infill points isolate the precise position.

4.1.2. The trust-region method

The above method will find at least a local minimum of the surrogate, given the mild assumptions that the objective function is smooth and continuous. Convergence may be rather lengthy depending upon the function. Alexandrov et al. [59] shows rigorous proofs of convergence to a local optimum from an arbitrary point for a trust-region based method which can be used if the surrogate interpolates the observed data and also matches the gradient of the objective function at the observed points. Of the surrogates we have considered, a gradient enhanced MLS and gradient enhanced Kriging are permissible. The trust-region method can also be employed by using the first order scaling

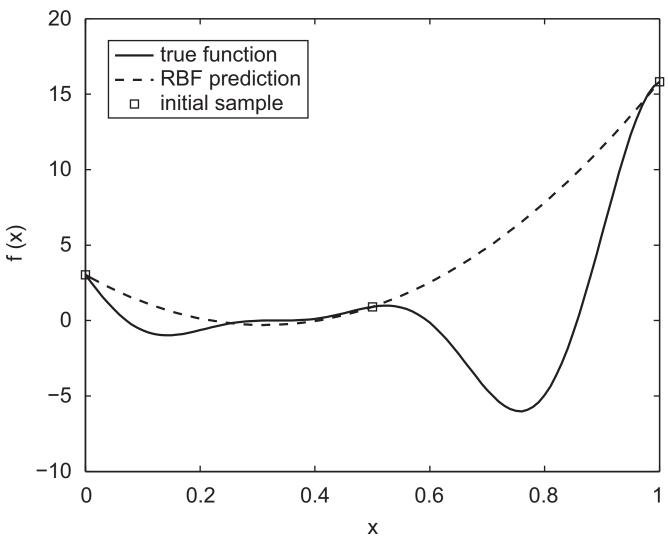

Fig. 19. An initial prediction of our one variable function using a Gaussian process model based on three points.

algorithm of Haftka [60] to match the gradient of the function. Eldred et al. [61] have extended this to second-order scaling.

In this approach we start at an arbitrary $\mathbf{x}_0$ and search a surrogate $\widehat{y}(\mathbf{x})$ in the interval $\mathbf{x}_0 \pm \delta_0$. The *trust-region* δ is initialized at some user defined value. The first plot in Fig. 21 shows a gradient enhance Kriging model of our one variable test function through $\mathbf{x}_0 = 0.5$. $\delta_0 = 0.25$ and the second plot shows an infill point at the minimum of the trust-region with a new gradient enhanced Kriging model through this point ($\mathbf{x}_1$) and the initial point. Based on this first iteration at $m = 0$, δ_m is updated as follows. We first evaluate how well the prediction performed as

$$r = \frac{f(\mathbf{x}_{m-1}) - f(\mathbf{x}_m)}{f(\mathbf{x}_{m-1}) - \widehat{y}(\mathbf{x}_m)}, \tag{77}$$

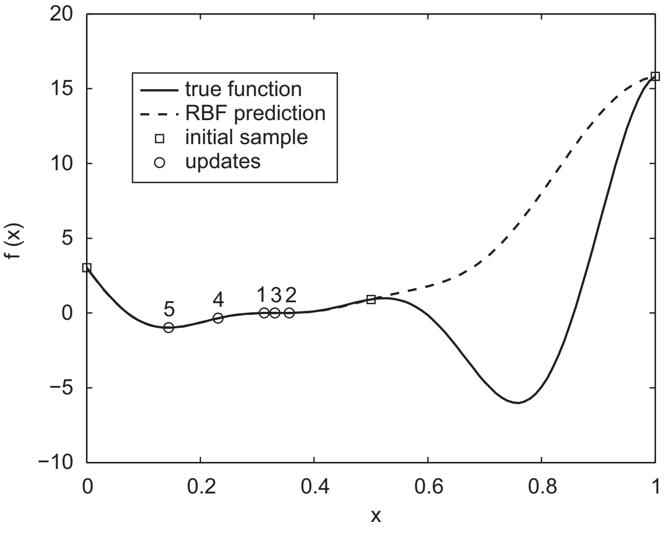

Fig. 20. Minimum prediction based infill points, starting from the prediction in Fig. 19, converging towards a local optimum.

then calculate the new trust-region as

$$\delta_m = \begin{cases} c_1\|\mathbf{x}_m - \mathbf{x}_{m-1}\| & \text{if } r < r_1, \\ \min\{c_2\|\mathbf{x}_m - \mathbf{x}_{m-1}\|, \Delta\} & \text{if } r > r_2, \\ \|\mathbf{x}_m - \mathbf{x}_{m-1}\| & \text{otherwise}. \end{cases} \tag{78}$$

$c_1 < 1$ and $c_2 > 1$ are factors affecting the degree to which the trust-region shrinks and expands depending on how well the surrogate performs (here we have used $c_1 = 0.75$ and $c_2 = 1.25$). r_1 and r_2 determine how poorly we allow the surrogate to perform before reducing the trust-region and how well it must perform before increasing the trust-region. Typical values are $r_1 = 0.10$ and $r_2 = 0.75$.

We now find $\mathbf{x}_2$ by minimizing $\widehat{y}(\mathbf{x})$ in the region $\mathbf{x}_1 \pm \delta_1$. The process is repeated until a stopping criterion is met (see [62] for information on stopping criteria). The remaining plots in Fig. 21 show a further two infill points converging towards a local minimum of the function. The above description outlines the core of the trust-region approach to the use of surrogate models. More details can be found in Alexandrov et al. [59], with a multi-fidelity implementation in Alexandrov et al. [63].

Although the above exploitation-based infill criteria are attractive methods for local optimization, it is clear from Fig. 20 that a prediction-based infill criterion may not find the global optimum of a deceptive objective function. Likewise, although the trust-region approach will find a local optimum from an arbitrary starting point, it may not find the global optimum if $\mathbf{x}_0$ is not in the global basin of attraction. To locate the true global optimum, we clearly need an element of *exploration* in our infill criterion.

4.2. Exploration

Pure design space exploration can essentially be viewed as filling in the gaps between existing sample points. Perhaps the simplest way of doing this is to use a sequentially space filling sampling plan such as a Sobol sequence or LPτ array [64,65], although such sample methods exhibit rather poor space filling characteristics for small samples. New points could also be

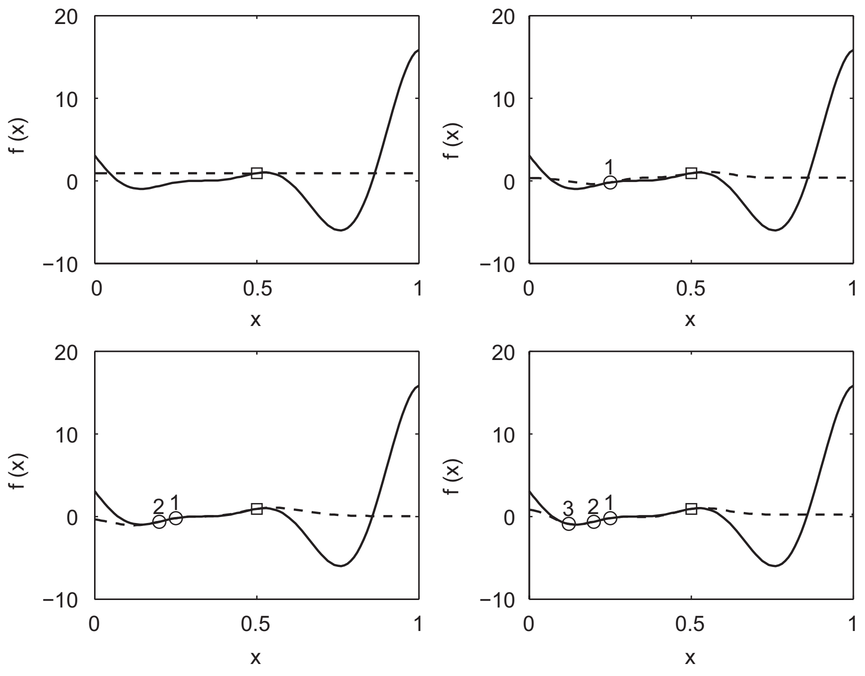

Fig. 21. Trust-region based infill points to a gradient enhanced Kriging prediction.

positioned using a maxi min criterion [7]. If error estimates are available for the surrogate, infill points can be positioned at points of maximum estimated error. Error estimates from regressing models, with the exception of those with the modified Gaussian process variance (Eq. (35)), are of dubious merit here. For the exploration of a design space populated by computer experiments we require that the estimated error returns to zero at all sample locations. Otherwise we run the risk of the maximum error occurring at a previously visited point. While this is a valid outcome in the world of physical experiments with their random errors, re-running a deterministic computer experiment as an infill point is useless.

The Gaussian process based models considered in this paper assume a stationary covariance structure, that is the basis function variance is constant across the design space. The model does not account for some areas of the design space having more activity than others, e.g. flat spots may not be modelled effectively. This is unlikely to be a serious problem for optimization, but may be for building a model which accurately predicts the underlying function in *all* areas. For such stationary covariance models a maximum error based infill will indeed just fill in the gaps between sample points. It is possible to build a surrogate with a non-stationary covariance [66]. Maximum error infill points based on such a model may well perform better at improving generalization than simply using a space filling sample with more points.

Pure exploration is of dubious merit in an optimization context. Time spent accurately modelling suboptimal regions is time wasted when all we require is the global optimum itself. Exploration-based infill has its niche in design space visualization and comprehension where the object is to build an accurate approximation of the entire design landscape to help the designer visualize and understand the design environment they are working in. We will not dwell on visualization issues, but those interested in design space visualization might consult Holden [67]. Exploration also has a role in producing a globally accurate model when the final surrogate is to be used in a realtime control system or in a more complex overarching calculation, such as aeroelasticity.

4.3. *Balanced exploration/exploitation*

We know that to exploit the surrogate before the design space has been explored sufficiently may lead to the global optimum lying undiscovered, while over exploration is a waste of resources. Thus the Holy Grail of global optimization is finding the correct balance between exploitation and exploration. Concurring with Jones [32], we will split the following discussion into two breeds of infill criteria: one- and two-stage methods. In a two-stage method the surrogate is fitted to the data and the infill criterion calculated based upon this model. In a one-stage approach the surrogate is not fixed when calculating the infill criterion, rather the infill criterion is used to calculate the surrogate. We will begin with the simpler and more common two-stage methods.

4.3.1. *Two-stage approaches*

Statistical lower bound: The simplest way of balancing exploitation of the prediction $\hat{y}(\mathbf{x})$ and exploration using $s^2(\mathbf{x})$ (e.g., Eq. (22)) is to minimize a *statistical lower bound*

$$LB(\mathbf{x}) = \hat{y}(\mathbf{x}) - As(\mathbf{x}), \tag{79}$$

where A is a constant that controls the exploitation/exploration balance. As $A \to 0$, $LB(\mathbf{x}) \to \hat{y}(\mathbf{x})$ (pure exploitation) and as $A \to \infty$, the effect of $\hat{y}(\mathbf{x})$ becomes negligible and minimizing $LB(\mathbf{x})$ is equivalent to maximizing $s(\mathbf{x})$ (pure exploration). A key problem is that, as the method stands, it is difficult to choose a value for A. For example, a suitable choice of A for one function might lead to over exploitation of another. In Section 4.3.2 we will look at how a one-stage approach can solve this problem. A possible solution for a two-stage implementation is to try a number of values of A and position infill points where there are clusters of minima of Eq. (79).

Probability of improvement: By considering $\hat{y}(\mathbf{x})$ as the realization of a random variable we can calculate the probability of an improvement $I = y_{\min} - Y(\mathbf{x})$ upon the best observed objective value so far, $y_{\min}$:

$$P[I(\mathbf{x})] = \frac{1}{s\sqrt{2\pi}} \int_{-\infty}^{0} e^{(-(I-\hat{y}(\mathbf{x}))^2)/2s^2}\, dI. \tag{80}$$

This equation is interpreted graphically in Fig. 22. The figure shows the prediction in Fig. 20 along with a vertical Gaussian distribution with variance $s^2(\mathbf{x})$ centred around $\hat{y}(\mathbf{x})$. This Gaussian distribution represents our uncertainty in the prediction $\hat{y}(\mathbf{x})$ and the part of the distribution below the horizontal dotted line indicates the possibility of improving on the best observed value (the quantity we are integrating in Eq. (80)). The probability of improvement is the area enclosed by the Gaussian distribution below the best observed value so far (the value of the integral in Eq. (80)).

Expected improvement: Instead of simply finding the probability that there will be some improvement, we can calculate the amount of improvement we expect, given the mean $\hat{y}(\mathbf{x})$ and variance $s^2(\mathbf{x})$. This *expected improvement* is given by

$$E[I(\mathbf{x})] = \begin{cases} (y_{\min} - \hat{y}(\mathbf{x}))\Phi\left(\dfrac{y_{\min} - \hat{y}(\mathbf{x})}{s(\mathbf{x})}\right) + s\phi\left(\dfrac{y_{\min} - \hat{y}(\mathbf{x})}{s(\mathbf{x})}\right) & \text{if } s>0, \\ 0 & \text{if } s=0, \end{cases} \tag{81}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the normal cumulative distribution function and probability density function, respectively. This equation can be interpreted graphically from Fig. 22 as the first moment of the area enclosed by the Gaussian distribution below the best observed value so far.

The progress of maximum $E[I(\mathbf{x})]$ updates to our one-variable test function is shown in Fig. 23. Clearly the $E[I(\mathbf{x})]$ has escaped the local minimum to the left and succeeded in locating the global optimum.

The infill criteria we have reviewed so far represent the current tools of choice for surrogate-based optimization in the aerospace

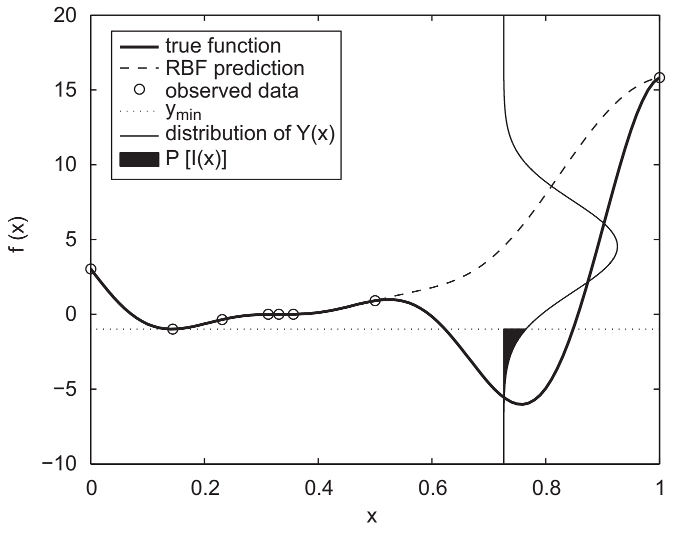

Fig. 22. A graphical interpretation of the probability of improvement.

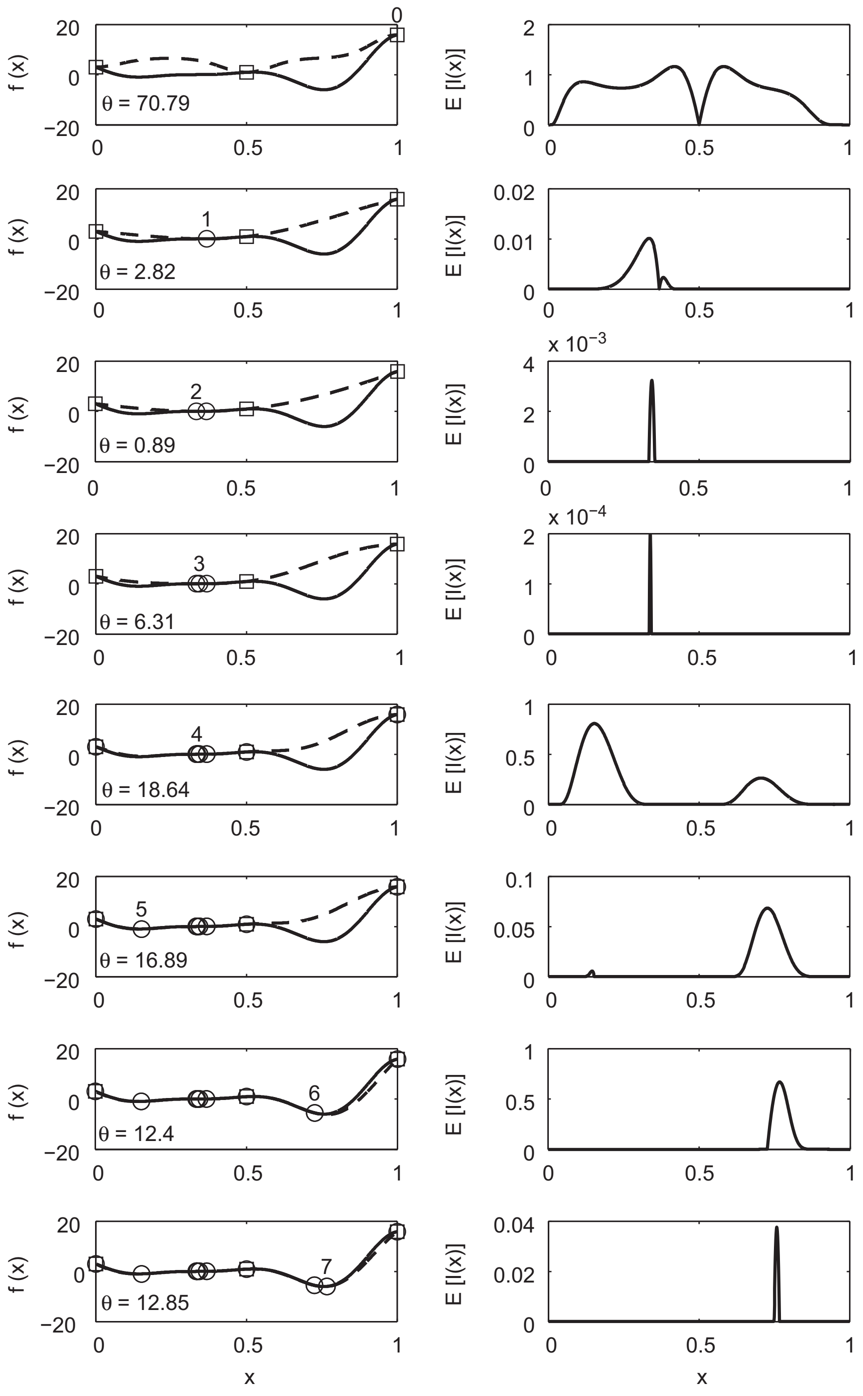

Fig. 23. The progress of a search of the one variable test function using a maximum $E[I(\mathbf{x})]$ infill strategy.

industry. We will now look at a more recently developed breed of infill criteria, which attempt to address some of the problems with those we have covered so far. In many situations maximizing $E[I(\mathbf{x})]$ will prove to be the best route to finding the global optimum and as such has become very popular as a tool for global optimization, evident from the number of citations to the seminal paper by Jones et al. [58]. Should the assumptions through which we base our confidence in this method prove to be false, maximizing $E[I(\mathbf{x})]$ (and $P[I(\mathbf{x})]$) may converge very slowly or not at all. The assumption upon which $E[I(\mathbf{x})]$ and $P[I(\mathbf{x})]$ trip up is that they assume that the model parameters have been estimated accurately based on the observed data. Note that, although the search in Fig. 23 does locate the optimum, it dwells in a region which does not even contain a local optimum until θ is estimated correctly. That is, until there is sufficient data to estimate θ correctly. In situations where data are sparse and/or the true function is deceptive, we may wish to consider a breed of infill criteria which can alleviate this pitfall.

4.3.2. One-stage approaches

All the above infill criteria could *possibly* be mislead by a particularly poor or unlucky initial sample and a very deceptively positioned optimum. Consider the function shown in Fig. 24 which, although on face value looks rather contrived, represents the worst case scenario of a type of situation that can occur in surrogate-based optimization. We have been unlucky enough to sample the function at three points with the same function value. An error-based infill criterion cannot cope with the prediction in Fig. 24 because the estimated error is zero for all values of x and so $P[I(\mathbf{x})]$ or $E[I(\mathbf{x})]$ would also be zero. The error does not have to be zero in all areas for problems to arise. Slow convergence of error-based infill criteria can occur whenever there is a significant underestimation of the error.

In situations like that in Fig. 24 we need to employ an infill criterion which takes into account the possibility that a deceptive sample may have resulted in significant error in the model parameters. The criteria we will consider do not use the surrogate to find the minimum, but rather use the minimum to find the surrogate. Or, in a sound bite (paraphrased from [68])—ask not what the surrogate implies about the minimum—ask what the minimum implies about the surrogate.

Goal seeking: We may be able to estimate a suitable value for the global optimum or perhaps we would just like to search for a specific improvement, even if it is not known if that improvement is possible. In such cases we can use a method which does not search for expectations or probabilities of improvement, but assesses the likelihood that an objective function value *could* exist at a given point [32].

The Kriging predictor can be considered as a maximum likelihood estimate of the sample data augmented with the point to be predicted. Instead of estimating the value $\hat{y}(\mathbf{x})$ for a given $\mathbf{x}$, we can assume the predictor passes through a goal y^g as well as the sample data and find the value of $\mathbf{x}^g$ which best fits this assumption. To do this we maximize the *conditional ln-likelihood*

$$-\frac{n}{2}\ln(2\pi)-\frac{n}{2}\ln(\hat{\sigma}^2)-\frac{1}{2}\ln|\mathbf{C}|-\frac{(\mathbf{y}-\mathbf{m})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{y}-\mathbf{m})^{\mathrm{T}}}{2\hat{\sigma}^2}, \tag{82}$$

where

$$\mathbf{m}=\mathbf{1}\mu+\boldsymbol{\psi}(\hat{y}^g-\mu) \tag{83}$$

and

$$\mathbf{C}=\boldsymbol{\Psi}-\boldsymbol{\psi}\boldsymbol{\psi}^{\mathrm{T}}, \tag{84}$$

by varying $\mathbf{x}^g$ *and* the model parameters (at this stage we may wish to widen any upper and lower bounds on $\boldsymbol{\theta}$). The position of the goal, $\mathbf{x}^g$, appears in Eq. (82) via its vector of correlations with the observed data, $\boldsymbol{\psi}$. We must maximize the conditional ln-likelihood numerically in the same way as for tuning the model parameters. We can first make a substitution for the MLE $\hat{\sigma}^2$

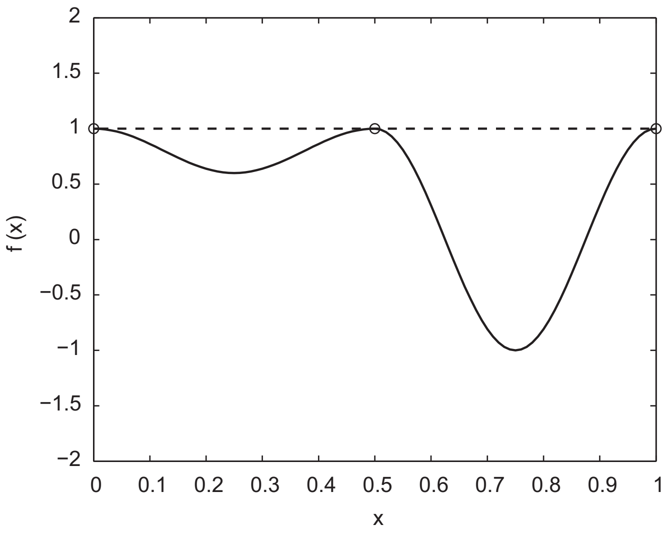

Fig. 24. A deceptive function with a particularly unlucky sampling plan.

(Eq. (31)) to give the concentrated conditional ln-likelihood:

$$-\frac{n}{2}\ln(\hat{\sigma}^2)-\frac{1}{2}\ln|\mathbf{C}|. \tag{85}$$

To see how effective this method can be we will consider the search of our one-dimensional test function. We begin with three sample points and set an objective function goal of -7 (a little less than the true optimum, but let us assume we do not know what that is). Fig. 25 shows the progress of infill points positioned at locations which maximize the conditional likelihood of the goal. Despite its deceptive location, the goal seeking method quickly finds the global optimum. We cannot choose a purely arbitrary goal. An overly optimistic goal will lead to too much exploration, since there will be an equally low likelihood in many areas. A pessimistic goal will result in a local search, but the goal will quickly be obtained, which may well be an acceptable outcome. Gutmann [69] suggests, and has had success, trying a range of goals and positioning infill points where there are clusters of optimal infill locations.[3] A more elegant method, when a suitable estimate for a goal cannot be made, is to calculate a lower bound based on the conditional likelihood.

The conditional lower bound: In many cases we will not be able to specify a goal for the optimization, but we can still use a conditional likelihood approach. Instead of finding the $\mathbf{x}$ which gives the highest likelihood conditional upon $\hat{y}(\mathbf{x})$ passing through a goal, we find the $\mathbf{x}$ which minimizes $\hat{y}(\mathbf{x})$ subject to the conditional likelihood not being too low [68].

Again, consider the prediction of our deceptive one variable test function based on an initial sample of three points. This is shown in Fig. 26, along with the statistical lower bound found by subtracting the estimated RMSE ($s(\mathbf{x})$). At $x=0.7572$, which we know is the minimum of the function, a point with $y^h=\hat{y}(\mathbf{x})$ has been imputed (i.e. we have hypothesized that this point is part of the sample data, even though it has not actually been observed). The likelihood conditional upon the prediction passing through this point is shown. Subsequently, we have imputed lower and lower values at $x=0.7572$ and re-optimized $\hat{\theta}$ to produce a prediction through these points. These values fall well below our statistical lower bound, but still have a conditional likelihood and so represent possible values at $x=0.7572$. As the imputed value reduces the conditional likelihood becomes extremely low and we clearly need a systematic method of dismissing imputations which are very unlikely. We achieve this using a likelihood ratio test.

By calculating the ratio of the conditional likelihood of $\hat{y}$ (using the maximum likelihood estimate of θ L_0, to the conditional likelihood, L_{cond}, of the prediction passing through the imputed point and comparing to the χ^2 distribution, we can make a decision as to whether to accept the value of the imputed point. To be accepted

$$2\ln\frac{L_0}{L_{\text{cond}}}<\chi^2_{\text{critical}}(\text{limit},\text{dof}) \tag{86}$$

must be satisfied. The value of the critical χ^2 value will depend upon the confidence limit we wish to obtain and the number of degrees of freedom (the number of model parameters). For the example in Fig. 26, if we wish to obtain a confidence limit of 0.95, we use limit $=0.975$ (we are only considering the lower bound) and dof $=1$ to obtain $\chi^2_{\text{critical}}=5.0239$ (from tables or, e.g., *Matlab*). Fig. 26 shows the likelihood ratio for each hypothesized

[3] This is similar to trying a range of weightings between local and global search. Sóbester et al. [70] used a weighted expected improvement formulation as part of a two-stage approach to achieve similar ends, while Forrester [71] used a weighted statistical lower bound with reinforcement learning to chose the weighting.

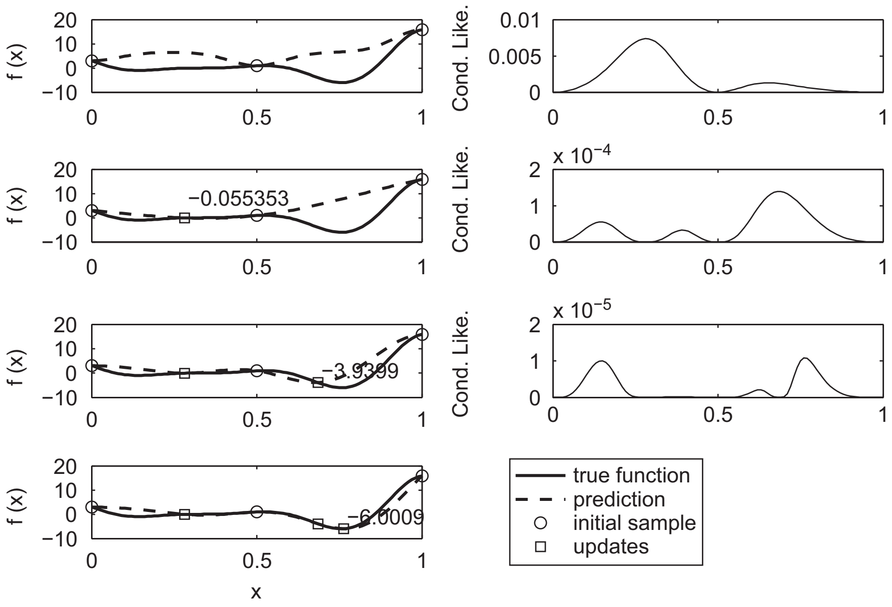

Fig. 25. The progress of a search of the one variable test function in the range {0, 1} using a goal seeking infill strategy.

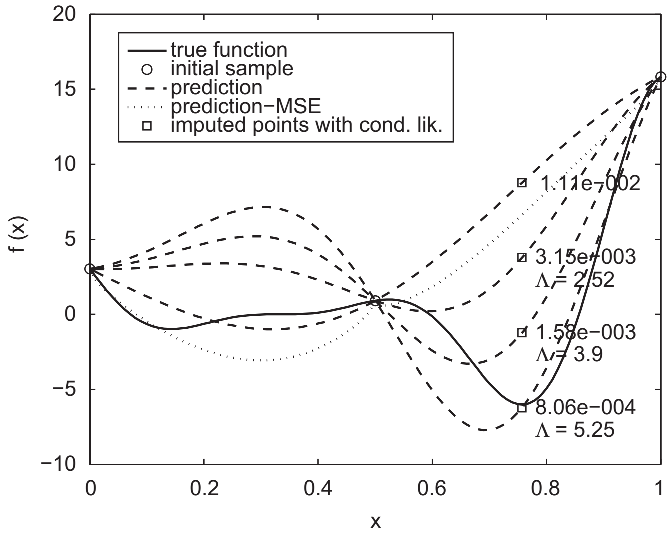

Fig. 26. The conditional likelihood and likelihood ratio for hypothesized points with increasingly lower objective function values.

point which has been imputed, calculated using the conditional likelihoods shown. The lowest value would be rejected based on χ^2_{critical}.

Using this likelihood ratio test we can systematically compute a lower confidence bound for the prediction. The minimum of this lower bound can then be used as an infill criterion. To choose a new infill point we must minimize y^h by varying y^h, $\mathbf{x}$, and the model parameters, subject to the constraint defined by (86). Fig. 27 shows the progress of a search of the deceptive one variable test problem using this infill criterion, starting from the same three point initial sample. A 95% confidence interval has been chosen. Despite us not specifying a goal *a priori*, the infill strategy has quickly found the global optimum. We are still left with a rather annoying control parameter—we must choose the confidence interval and it is not entirely clear what is the best method of doing this. In a similar vein to Gutmann's goal seeking method, a number of confidence intervals could be tried.

Kriging is known to give inaccurate error estimates, particularly with sparse sampling [72] and the conditional lower bound can, in fact, be used to calculate what may be a more reliable Gaussian process based model error estimate by setting the confidence interval to give one standard deviation. By using the conditional lower bound approach to calculate error estimates, the two-stage approaches of probability of improvement and expected improvement can be transformed into one-stage methods, allowing problems such as that shown in Fig. 24 to be solved with the added benefits of $E[I(\mathbf{x})]$ over a lower bound criterion. The benefits are that, assuming the function to be searched is smooth and continuous, $E[I(\mathbf{x})]$ can be proved to converge to the global optimum. It can also readily be modified to account for constraints and multiple objectives, which we will consider in the next two sections. Forrester and Jones [73] show the formulation of an expected improvement criterion using conditional lower bound based error estimates.

These one-stage approaches seem to be the panacea we have been looking for, but in many situations they could prove to be intractable. In the conditional bound approach, for example, $\mathbf{x}$, y^h, σ and $\mathbf{p}$ (for a Kriging model) need to be varied to minimize the lower bound. This search of up to $3k+1$ parameters is naturally far more computationally intensive than a standard two-stage method, particularly when we have a significant number of sample points. Recall that at each step in this $3k+1$ dimensional search we must invert a matrix whose dimensions are the size of the data set. If, however, the data set is rather limited because the underlying function is extremely expensive, this may nonetheless be worthwhile. This will sometimes be the case in high fidelity CFD or non-linear FEA based optimization.

4.4. *Parallel infill*

We have so far assumed that infill points will be applied in a serial process, but it is often possible to apply parallel updates. Many of the infill criteria we have reviewed exhibit multi-modal behaviour. In particular, the $E[I(\mathbf{x})]$ and conditional likelihood plots in Figs. 23 and 25. A search which locates a number of minima, e.g. multi-start hill-climbers or a genetic algorithm with clustering, can be employed to obtain a number of infill points [74]. The

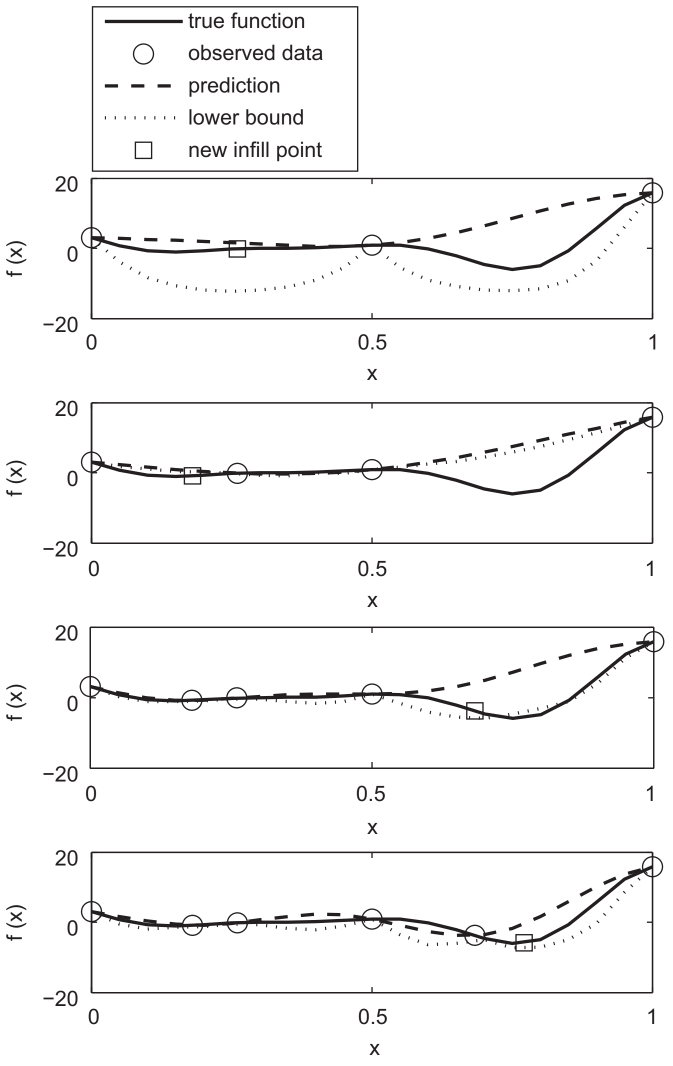

Fig. 27. The progress of a search of the one variable test function in the range [0, 1] using a conditional lower bound infill strategy.

y can then be evaluated simultaneously to take advantage of parallel computing capabilities. We cannot guarantee how many infill points will be obtained and so the process may not take advantage of all available resources.

A method of obtaining a specified number of infill points has been suggested by Schonlau [75]. We search the infill criterion to find its global optimum and then temporarily add the surrogate model predicted value at this point, i.e. assume the model is correct at this location and impute its value. The surrogate is then constructed with this temporary new point (with no need to re-estimate model parameters) and the infill criterion searched again. For $P[I(\mathbf{x})]$ and $E[I(\mathbf{x})]$ we do not change $f_{\min}$, should the imputed prediction be lower than this. The process is continued until the desired number of infill points has been obtained. These infill points are then evaluated and added to the data set in place of the temporary predictions. This method makes effective use of parallel computing resources, though the sampling may not be as well placed as a sequential scheme.

We note in passing that setting up and searching a surrogate of any kind can be a bottleneck in a heavily parallel computing environment. If we have sufficient processors, then evaluating all the points in our initial sampling plan can occur simultaneously. We must then pull all these results together to build and study the surrogate before we can return to our parallel calculation of sets of infill points. This fact will always limit the amount of time we can dedicate to the building and searching of surrogates.

5. Constraints

Traditional constrained optimization approaches can be applied to a surrogate-based optimization process. Of note is the use of the augmented Lagrangian method for constrained optimization in a surrogate-based trust region search [63]. More simple to implement is the application of penalty functions [76]. Whether the constraint is cheap, and evaluated directly, or expensive, and a surrogate model of the constraint is employed, in most cases penalty functions can be applied to surrogate-based search in the usual manner. When one or more constraints are violated, a suitable penalty is applied to the value obtained from the surrogate of the objective function. Thus a search of the surrogate is deviated away from regions of violation. For a $\max\{E[I(\mathbf{x})]\}$ or $\max\{P[I(\mathbf{x})]\}$ based search, $y_{\min}$ should be replaced with the minimum observed function value which satisfies the constraint.

We may not be able to model a constraint function, for example when a region of infeasibility is defined purely by objective function calculations failing. In such situations we can penalize the surrogate in regions of failures by imputing large objective function values at failed points. Forrester et al. [77] used $\hat{y}(\mathbf{x}_{\text{failed}}) + s^2(\mathbf{x}_{\text{failed}})$ for imputed points and showed this to work well for an aerofoil design problem.

Assuming we can model the constraint function(s), a fully probabilistic approach can be taken to their inclusion. We shall concentrate on this surrogate model specific method. Before delving into the mathematics, it is useful to set out what we might expect when using Gaussian process (e.g. Kriging) models for both the objective and constraint functions. If, at a given point in the design space, the predicted errors in a constraint model are low and the surrogate shows a constraint violation, then the expectation of improvement will also be low, but not zero, since there is a finite possibility that a full evaluation of the constraints may actually reveal a feasible design. Conversely, if the errors in the constraints are large then there will be a significant chance that the constraint predictions are wrong and that a new point will, in fact, be feasible. Thus the expectation of improvement will be greater. Clearly, for a fully probabilistic approach we must factor these ideas into the calculation of the expectation. It turns out that this is relatively simple to do, although it is rarely mentioned in the literature (a formulation can be found in the thesis of Schonlau [75]). Provided that we assume that the constraints and objective are all uncorrelated a closed form solution can readily be derived. If not, and if the correlations can be defined, then numerical integration in probability space is required. Since such data is almost never available this idea is not pursued further here.

We have already discussed the probability of improvement infill criterion. Now consider a situation when we have a constraint function, also modelled by a Gaussian process, based on sample data in exactly the same way. Rather than calculating $P[I(\mathbf{x})]$, we could use this model to calculate the probability of the prediction being greater than the constraint limit, i.e. the probability that the constraint is met, $P[F(\mathbf{x})>g_{\min}]$. The probability that a design is feasible can be calculated following the same logic as for an improvement, only now instead of using the current best design as the dividing point in probability space we use the constraint limit value, i.e.

$$P[F(\mathbf{x})>g_{\min}] = \frac{1}{s\sqrt{2\pi}} \int_0^{\infty} \mathrm{e}^{-(F-\hat{g}(\mathbf{x}))^2/2s^2}\,\mathrm{d}G, \tag{87}$$

where g is the constraint function, $g_{\min}$ is the limit value, F is the measure of feasibility $G(\mathbf{x}) - g_{\min}$, $G(\mathbf{x})$ is a random variable, and s is the variance of the Kriging model of the constraint. We can couple this result to the probability of improvement from a Kriging model of the objective and the probability that a new infill point both improves on the current best point and is also feasible is then just

$$P[I(\mathbf{x}) \cap F(\mathbf{x}) > g_{\min}] = P[I(\mathbf{x})]P[F(\mathbf{x}) > g_{\min}], \tag{88}$$

since these are independent models.

We can also use the probability that a point will be feasible to formulate a constrained expected improvement. We simply multiply $E[I(\mathbf{x})]$ (Eq. (81)) by $P[F(\mathbf{x}) > g_{\min}]$:

$$E[I(\mathbf{x}) \cap F(\mathbf{x}) > g_{\min}] = E[I(\mathbf{x})]P[F(\mathbf{x}) > g_{\min}]. \tag{89}$$

As an example, the first plot of Fig. 28 shows our one variable function along with a constraint function (simply the negative of the objective minus a constant). The second plot shows $E[I(\mathbf{x})]$, the third plot shows the probability of satisfying the constraint (Eq. (88)), and the fourth plot shows the product of these—our constrained expected improvement (Eq. (89)). Note how multiplying by the probability of satisfying the constraint forces the expectation away from the region where the constraint is violated

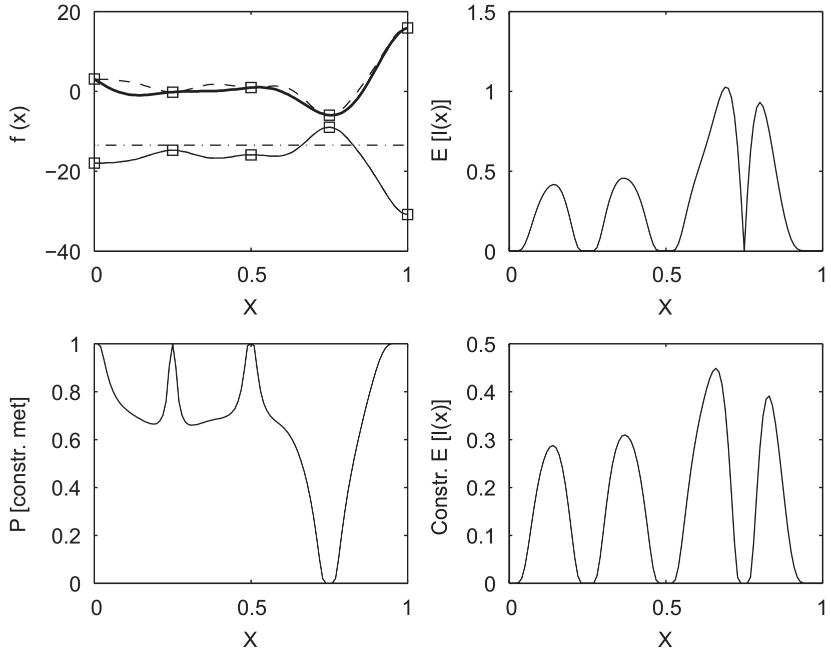

Fig. 28. Predictions of the objective (dash) and constraint functions (thin solid) based on four sample points, with the constraint limit (here $g_{\max}$, which we wish to be below) shown as a dash-dot line (first plot), the unconstrained $E[I(\mathbf{x})]$ (second plot), the probability of meeting the constraint (third plot) and the constrained expected improvement (final plot).

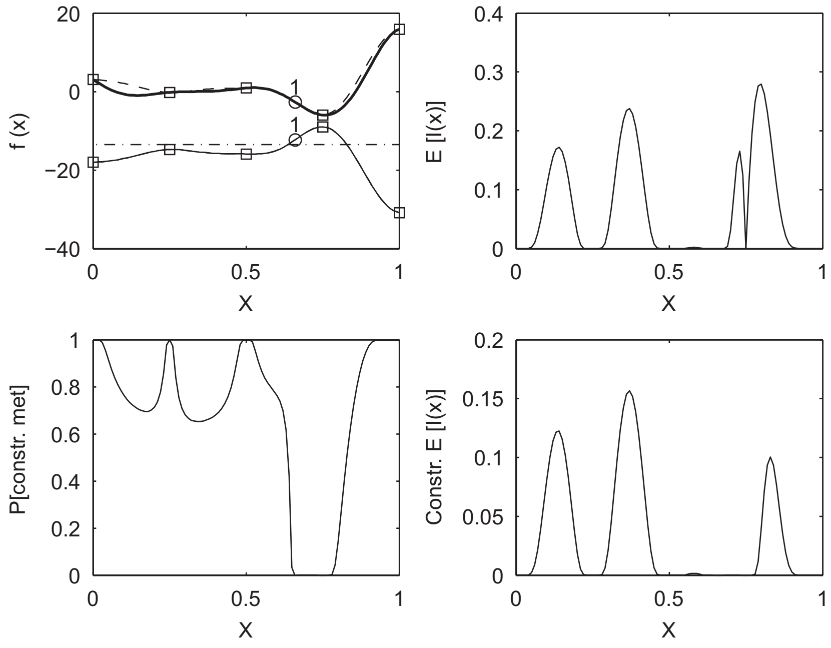

Fig. 29. The build up of the constrained expected improvement after an infill point has been applied at the maximum constrained $E[I(\mathbf{x})]$ in Fig. 28.

and the next infill will be on the constraint boundary (see Fig. 29, which shows the situation after this infill point has been applied).

6. Multiple objectives

In aerospace design it is common to be aiming for light weight, low cost, robust, high performance systems. These aspirations are clearly in tension with each other and so compromise solutions have to be sought. The final selection between such compromises inevitably involves deciding on some form of weighting between the goals. However, before this stage is reached it is possible to study design problems from the perspective of Pareto sets. A Pareto set of designs is one whose members are all optimal in some sense, but where the relative weighting between the competing goals is yet to be finally fixed (see for example [78]). More formally, a Pareto set of designs contains systems that are sufficiently optimized that, to improve the performance of any set member in any one goal function, its performance in at least one of the other functions must be made worse. The designs in the set are said to be *non-dominated* in that no other set member exceeds a given design's performance in all goals. It is customary to illustrate a Pareto set by plotting the performance of its members against each goal function, see Fig. 30, where the two axes are for two competing goal functions that must both be minimized. The series of horizontal and vertical lines joining the set members is referred to as the *Pareto front*—any design lying above and to the right of this line is dominated by members of the set.

There are a number of technical difficulties associated with constructing Pareto sets. Firstly, the set members need to be optimal in some sense—since it is desirable to have a good range of designs in the set this means that an order of magnitude more optimization effort is usually required to produce such a set than to find a single design that is optimal against just one goal. Secondly, it is usually necessary to provide a wide and even coverage in the set in terms of the goal function space—since the mapping between design parameters and goal functions is usually highly non-linear, gaining such coverage is far from simple. Finally, and in common with single objective design, many problems of practical interest involve the use of expensive computer simulations to evaluate the performance of each candidate, and this means that only a limited number of such simulations can usually be afforded.

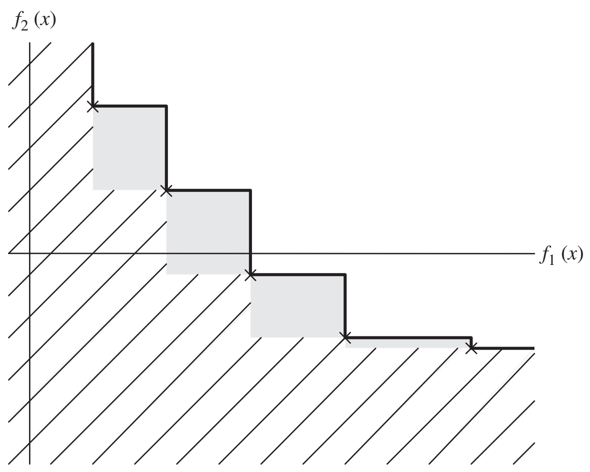

Fig. 30. A Pareto set of five non-dominated points (×) for a problem with two objectives. The solid line is the Pareto front. The shaded area shows where new points would augment the Pareto front, while the hatched area is where new points would dominate and replace the existing set of non-dominated points.

Currently, there appear to be two popular ways of constructing Pareto sets. First, and most simply, one chooses a (possibly non-linear) weighting function to combine all the goals in the problem of interest into a single quantity and carries out a single objective optimization. The weighting function is then changed and the process repeated. By slowly working through a range of weightings it is possible to build up a Pareto set of designs. In a similar vein, one can also search a single objective at a time, while constraining the other objectives. Slowly working through a range of constraint values, a Pareto set can be populated. However, it is by no means clear what weighting function or constraint values to use and how to alter them so as to be able to reach all parts of the potential design space (and thus to have a wide ranging Pareto set). In particular, the weighted single objective method will miss Pareto optimal points if the front is not convex and the weighting in linear. If it is non-linear this can be avoided, but then the form of the function to use must be decided upon.

In an attempt to address this limitation designers have turned to a second way of constructing Pareto sets via the use of population-based search schemes. In such schemes a set of designs is worked on concurrently and evolved towards the final Pareto set in one process. In doing this, designs are compared to each other and progressed if they are of high quality and if they are widely spaced apart from other competing designs. Moreover such schemes usually avoid the need for an explicit weighting function to combine the goals being studied. Perhaps the most well known of these schemes is the NSGA-II method introduced by Deb et al. [79].

To overcome the problem of long run-times a number of workers have advocated the use of surrogate modelling approaches within Pareto front frameworks [80,81]. It is also possible to combine tools like NSGA-II with surrogates [82]. In such schemes an initial sampling plan is evaluated and surrogate models built as per the single objective case, but now there is one surrogate for each goal function. In the NSGA-II approach the search is simply applied to the resulting surrogates and used to produce a Pareto set of designs. These designs are then used to form an infill point set and, after running full computations, the surrogates are refined and the approach continued. Although sometimes quite successful this approach suffers from an inability to balance explicitly exploration and exploitation in the surrogate model construction, in just the same way as when using a prediction-based infill criterion in single objective search, although the crowding or niching measures normally used help mitigate these problems to some extent. Here we consider statistically based operators for use in surrogate model based multi-objective search so as to explicitly tackle this problem.

6.1. Multi-objective expected improvement

To begin with consider a problem where we wish to minimize two objective functions $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$, which we can sample to find observed outputs y_1 and y_2. For simplicity, assume that $\mathbf{x}$ consists of just one design variable x ($k=1$). By evaluating a sampling plan, X, we can obtain observed responses $\mathbf{y}_1$ and $\mathbf{y}_2$. This will allow us to identify the initial Pareto set of m designs that dominate all the others in the training set:

$$\mathbf{y}^*_{1,2} = \{[y_1^{(1)*}(x^{(1)*}), y_2^{(1)*}(x^{(1)*})], [y_1^{(2)*}(x^{(2)*}), y_2^{(2)*}(x^{(2)*})], \\ \ldots, [y_1^{(m)*}(x^{(m)*}), y_2^{(m)*}(x^{(m)*})]\}.$$

In this set the superscript * indicates that the designs are non-dominated. We may plot these results on the Pareto front axes as per Fig. 30. In that figure the solid line is the Pareto front and the hatched and shaded areas represent locations where new designs would need to lie if they are to become members of the Pareto set.

Note that if new designs lie in the shaded area they augment the set and that if they lie in the hatched area they will replace at least one member of the set (since they will then dominate some members of the old set). It is possible to set up our new metric such that an improvement is achieved if we can augment the set or, alternatively, only if we can dominate at least one set member—here we consider the latter metric.

Given the training set it is possible to build a pair of Gaussian process based models (e.g. Kriging models). As when dealing with constrained surrogates, it is assumed that these models are independent (though it is also possible to build correlated models by using co-Kriging, as per Section 3.7.2). The Gaussian processes have means $\widehat{y}_1(x)$ and $\widehat{y}_2(x)$ (from Eq. (20)), and variances $s_1^2(x)$ and $s_2^2(x)$ (from Eq. (22)). These values may then be used to construct a two-dimensional Gaussian probability density function for the predicted responses of the form

$$\phi(Y_1, Y_2) = \frac{1}{s_1(x)\sqrt{2\pi}}\exp\left[-\frac{(Y_1(x)-\widehat{y}_1(x))^2}{2s_1^2(x)}\right] \times \frac{1}{s_2(x)\sqrt{2\pi}}\exp\left[-\frac{(Y_2(x)-\widehat{y}_2(x))^2}{2s_2^2(x)}\right], \quad (90)$$

where it is made explicitly clear that $\widehat{y}_1(x)$, $s_1^2(x)$, $\widehat{y}_2(x)$ and $s_2^2(x)$ are all functions of the location at which an estimate is being sought. Clearly this joint pdf accords with the predicted mean and errors coming from the two Kriging models at x. When seeking to add a new point to the training data we wish to know the likelihood that any newly calculated point will be good enough to become a member of the current Pareto set and, when comparing competing potential designs, which will improve the Pareto set most.

We first considering the probability that a new design at x will dominate a single member of the existing Pareto set, say $[y_1^{(1)*}, y_2^{(1)*}]$. For a two-objective problem this may arise in one of three ways: either the new point improves over the existing set member in goal one, or in goal two, or in both (see Fig. 31). The probability of the new design being an improvement is simply $P[Y_1(x)<y_1^{(i)} \cap Y_2(x)<y_2^{(i)}]$, which is given by integrating the volume under the joint probability density function, i.e. by integrating over the hatched area in Fig. 31.

Next consider the probability that the new point is an improvement given all the points in the Pareto set. Now we must integrate over the hatched (and possibly the shaded) area in Fig. 30. We can distinguish whether we want the new point to augment the existing Pareto set or dominate at least one set member by changing the area over which the integration takes place. Here we will consider only points which dominate the Pareto set (for formulations which deal with Pareto set augmentation see Keane [83]). Carrying out the desired integral is best done by considering the various rectangles that comprise the hatched area in Fig. 30 and this gives

$$\begin{aligned} P[Y_1(x)<\mathbf{y}_1^* \cap Y_2(x)<\mathbf{y}_2^*] = {} & \int_{-\infty}^{y_1^{*(1)}}\int_{-\infty}^{\infty} Y_1\phi(Y_1,Y_2)\,\mathrm{d}Y_2\,\mathrm{d}Y_1 \\ & + \sum_{i=1}^{m-1}\int_{y_1^{*(i)}}^{y_1^{*(i+1)}}\int_{-\infty}^{y_2^{*(i+1)}} Y_1\phi(Y_1,Y_2)\,\mathrm{d}Y_2\,\mathrm{d}Y_1 \\ & + \int_{y_1^{*(m)}}^{\infty}\int_{-\infty}^{y_2^{*(m)}} Y_1\phi(Y_1,Y_2)\,\mathrm{d}Y_2\,\mathrm{d}Y_1. \quad (91) \end{aligned}$$

This is the multi-objective equivalent of the $P[I(\mathbf{x})]$ formulation in Section 4.3.1. It will work irrespective of the relative scaling of the objectives being dealt with. When used as an infill criterion it will not, however, necessarily encourage very wide ranging exploration since it is not biased by the degree of improvement being achieved. To do this we must consider the first moment of the integral, as before when dealing with single objective problems.

The equivalent improvement metric we require for the two objective case will be the first moment of the joint probability density function integral taken over the area where improvements occur, calculated about the current Pareto front. Now, while it is simple to understand the region over which the integral is to be taken (it is just the same as in Eq. (91)) the moment arm about the current Pareto front is a less obvious concept. To understand what is involved, it is useful to return to the geometrical interpretation of $E[I(\mathbf{x})]$ (shown in Fig. 22 for the single objective case). $P[I(\mathbf{x})]$ represents integration over the probability density function in the area below and to the left of the Pareto front where improvements can occur. $E[I(\mathbf{x}^*)]$ (we will use the * superscript to denote the multi-objective formulation) is the first moment of the integral over this area, about the Pareto front. Now the distance the centroid of the $E[I(\mathbf{x}^*)]$ integral lies from the front is simply $E[I(\mathbf{x}^*)]$ divided by $P[I(\mathbf{x}^*)]$, see Fig. 32. Given this position and $P[I(\mathbf{x}^*)]$ it is simple to calculate $E[I(\mathbf{x}^*)]$ based on any location along the front. Hence we first calculate $P[I(\mathbf{x}^*)]$ and the location of the centroid of its integral, $(\bar{Y}_1, \bar{Y}_2)$ (by integration with respect to the origin and division by $P[I(\mathbf{x}^*)]$). It is then possible to establish the Euclidean distance the centroid lies from each member of the Pareto set D. The expected improvement criterion is subsequently calculated using the set member closest to the centroid, $(y_1^*(x^*), y_2^*(x^*))$, by taking the product of the volume under the probability density function with the Euclidean distance between this member and

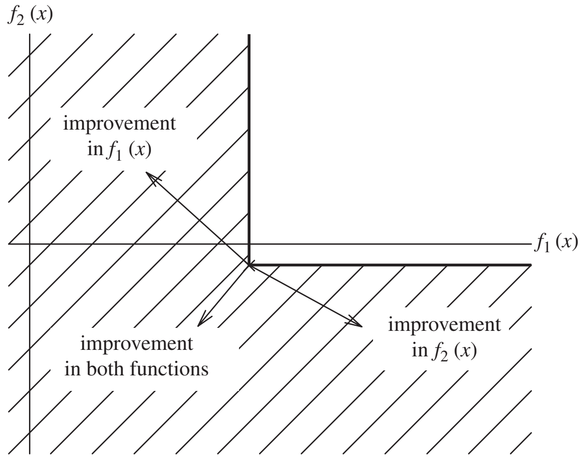
$f_2\,(x)$
improvement in $f_1\,(x)$
$f_1\,(x)$
improvement in both functions
improvement in $f_2\,(x)$

Fig. 31. Improvements possible from a single point in the Pareto set.

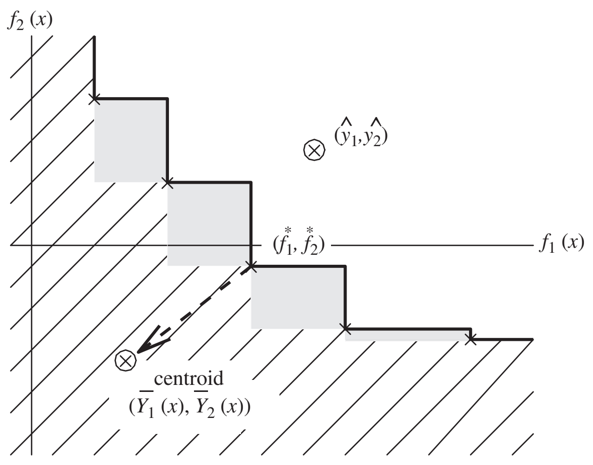

Fig. 32. Centroid of the probability integral and moment arm used in calculating $E[I(\mathbf{x}^*)]$, also showing predicted position of currently postulated update.

the centroid, shown by the arrow in Fig. 32. This leads to the following definition of $E[I(\mathbf{x}^*)]$:

$$E[I(\mathbf{x}^*)] = P[I(\mathbf{x}^*)]\sqrt{(\bar{Y}_1(x) - y_1^*(x^*))^2 + (\bar{Y}_2(x) - y_2^*(x^*))^2}, \qquad (92)$$

where

$$\bar{Y}_1(x) = \left\{ \begin{array}{l} \int_{-\infty}^{y_1^{*(1)}} \int_{-\infty}^{\infty} Y_1 \phi(Y_1, Y_2)\,\mathrm{d}Y_2\,\mathrm{d}Y_1 \\ \quad + \sum_{i=1}^{m-1} \int_{y_1^{*(i)}}^{y_1^{*(i+1)}} \int_{-\infty}^{y_2^{*(i+1)}} Y_1 \phi(Y_1, Y_2)\,\mathrm{d}Y_2\,\mathrm{d}Y_1 \\ \quad + \int_{y_1^{*(m)}}^{\infty} \int_{-\infty}^{y_2^{*(m)}} Y_1 \phi(Y_1, Y_2)\,\mathrm{d}Y_2\,\mathrm{d}Y_1 \end{array} \right\} \Bigg/ P[I(\mathbf{x}^*)], \qquad (93)$$

and $\bar{Y}_2(x)$ is defined similarly.

When defined in this way $E[I(\mathbf{x}^*)]$ varies with the location of the predicted position of the currently postulated update $(\hat{y}_1, \hat{y}_2)$—also shown in Fig. 32, and also with the estimated errors in this prediction, s_1 and s_2, since it is these quantities that define the probability density function being integrated.

The further the predicted update location lies below and to the left of the current Pareto front the further the centroid will lie from the front. Moreover, the further the prediction lies in this direction the closer the integral becomes to unity (since the greater the probability of the update offering an improvement). Both tendencies will drive updates to be improved with regard to the design objectives. Note that if there is a significant gap in the points forming the existing Pareto front, then centroidal positions lying in or near such a gap will score proportionately higher values of $E[I(\mathbf{x}^*)]$, since the Euclidean distances to the nearest point will then be greater. This pressure will tend to encourage an even spacing in the front as it is updated. Also, when the data points used to construct the Gaussian process model (i.e., all points available and not just those in the Pareto set) are widely spaced, the error terms will be larger and this tends to further increase exploration. Thus this $E[I(\mathbf{x}^*)]$ definition balances exploration and exploitation in just the same way as its one-dimensional equivalent.

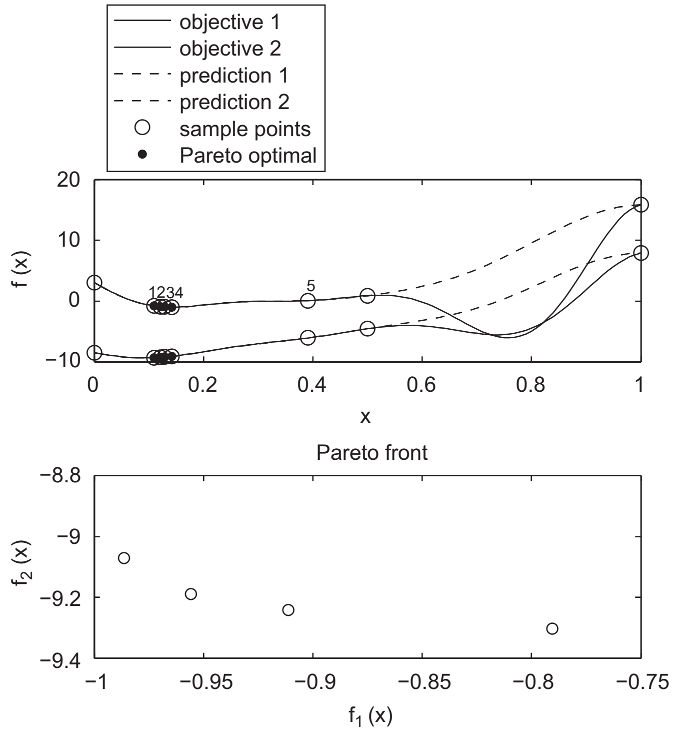

Fig. 33. The first four infill points positioned at the maximum expectation of improving on the Pareto front are all Pareto optimal.

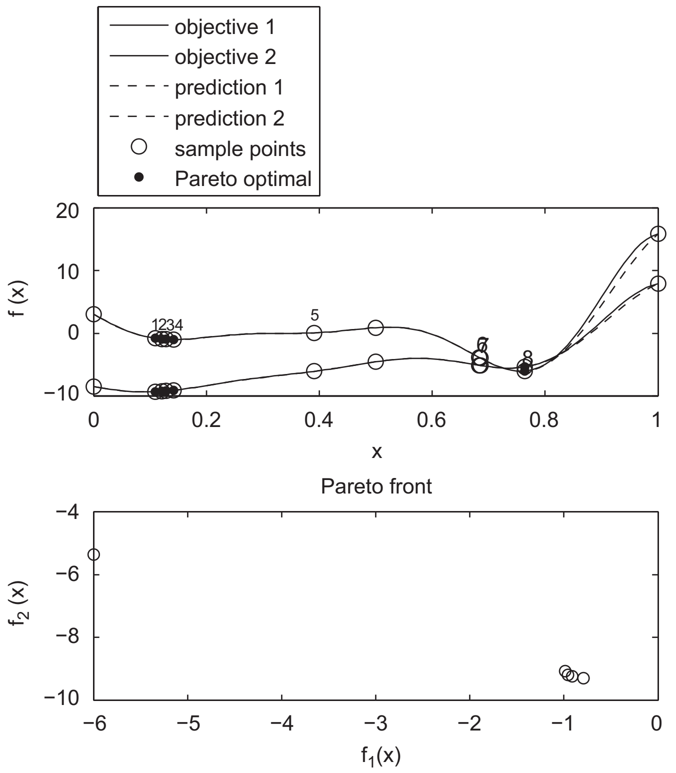

Fig. 34. Further updates locate the global optimum of objective one, which is also, naturally, Pareto optimal.

When calculating the location of the centroid there is still no requirement to scale the objectives being studied but, when deciding which member of the current Pareto set lies closest to the centroid, relative scaling will be important (i.e., when calculating the Euclidean distance). This is an unavoidable and difficult issue that arises whenever explicitly attempting to space out points along the Pareto front, whatever method is used to do this.

Again we will use our one variable test function for illustration. The first plot in Fig. 33 shows two objective functions, the first of which is that used in the previous examples. Starting from a three point initial sample, the first four infill points, based on maximizing the dual-objective expected improvement (Eq. (92)), are all Pareto optimal. Further updates lead to the location of the global optimum of objective one, which represents another part of the Pareto front, as shown in Fig. 34.

It is worth noting that there is no fundamental difficulty in extending this form of analysis to problems with more than two goal functions. This does, of course, increase the dimensionality of the Pareto surfaces being dealt with, and so inevitably complicates further the expressions needed to calculate the improvement metrics. Nonetheless, they always remain expressible in closed form; it always being possible to define the metrics in terms of summations over known integrable functions.

7. Discussion and recommendations

We have covered a range of surrogate modelling methods and infill criteria and have noted the pros and cons of each method along the way. We will now provide some more general thoughts on the applicability of the methods we have discussed. The

Table 1
A taxonomy of surrogate methods.

Sample plan: infill points ratio				$\leqslant\infty$ Comprehension		$>2:1$ Optimization	$\approx 1:2$	$<1:2$	
				Simple landscape	Complex landscape	Local search	$P[I(\mathbf{x})]$, $E[I(\mathbf{x})]$	Goal seeking	Conditional lower bound
$k>20$	$n>500$	SVR		√	√	√			
		Fixed bases e.g. cubic, thin plate		√		√		√	
		Polynomials		√					
$k<20$	$n<500$	MLS, parametric bases	E.g. multi-quadric	√	√	√		√	
			Gaussian bases e.g. Kriging	√	√	√	√	√	√

suitability of each method for various types of problem is shown in Table 1, which we have taken from Forrester et al. [8]. Naturally there are exceptions to every rule and it is risky to make such generalizations on the applicability of methods. The table does, however, give a concise view of the context in which we see each method.

Working through Fig. 1, while referring to Table 1, after any preliminary experiments we may wish to conduct to reduce the dimensionality of the problem, we must choose the number of points which our initial sampling plan will comprise. Assuming there is a maximum budget of function evaluations, we will define the number of points as a fraction of this budget. If our aim is purely to create an accurate model for visualization and design space comprehension, our sampling plan *could* contain all of our budgeted points. However, it is likely to be beneficial to position some points where it is believed that the error in the surrogate is high. Error estimates which reflect the possibility of varying function activity across the design space will be of most use here, e.g. from non-stationary Kriging [66]. If we are using the surrogate as the basis of an infill criteria, we must save some points for that process. For an exploitation-based criterion, most of the points, i.e. more than one half, should be in the initial sample because only a small amount of surrogate model enhancement is possible during the infill process. A notable exception is the trust-region example in Fig. 21 where we started from just one. If a two-stage balanced exploitation/exploration infill criterion is to be employed, Sóbester et al. [84] have shown that approximately one third of the points should be in the initial sample, with the majority saved for the infill stage. The one-stage methods rely less on the initial prediction and so fewer points are required.

The choice of which surrogate to use should be based on the problem size, that is k, the expected complexity, the cost of the analyses the surrogate is to be used in lieu of, and the form of infill strategy that will follow. As discussed at the end of Section 3.2, polynomial models make good surrogates of cheap analyses, following simple trends, in low dimensions. MLS (see Section 3.3) can model more complex landscapes, but is still limited to lower dimensions (for the same reasons as polynomials), and its added expense means that it may not be cheaper than some quick analyses. Fixed bases RBFs (see Section 3.4) are suitable for higher-dimensional, but simple landscapes and can be used in lieu of cheap analyses. SVR (see Section 3.6) with a fixed kernel also fits somewhere in this category, though the initial cost of training the model is higher than for RBFs. Our most complex surrogates—Kriging (see Section 3.5) and parametric RBFs, including parametric SVRs—can only be used for relatively low-dimensional problems due to the expense of training the model, but these methods have the potential to provide more accurate predictions.

Often the choice of surrogate modelling method will be dictated by the infill criteria we wish to apply. When this is not the case, although we can pigeon-hole which surrogate is likely to perform best for a given problem (as we have in Table 1), a more educated choice can be made using various model selection and validation criteria. The accuracy of a number of surrogates could be compared by assessing their ability to predict a validation data set. Such a strategy will require a portion of observed data to be set aside for validation purposes only, making this impractical when observations are expensive. More likely is that cross-validation or bootstrapping errors [85] will be compared when selecting the most accurate surrogate. These methods rely only on the observed data used to construct the surrogate being assessed.

Recently, rather than selecting one surrogate which appears to have better generalization properties, as determined by some validation metric, Goel et al. [86] have tried using a weighted average of an ensemble of surrogates. While Kriging, for example, might accurately predict some non-linear aspect of a function, a polynomial may better capture the underlying trend. By combining these two methods (and maybe others) in a weighted average, better generalization could be achieved. We would argue though that blind Kriging (see Section 3.5.2) can do this in a more mathematically rigorous manner. Ensembles or committees (as they are known in the machine learning literature [87]) are a powerful concept, indeed blind Kriging could be viewed as form of committee model. These 'Jack-of-all-trade' methods seem likely to find increasing favour in problems where the nature of the design landscapes is unknown.

We have already taken an in depth look at the various infill criteria and Table 1 shows which surrogate types these marry to. Essentially, for a surrogate to be suited to some form of search-infill process, the surrogate must have the capacity to modify its shape to fit any complex local behaviour the true function may exhibit. Thus, polynomials must be excluded, since, for practical purposes, there is a limitation on the order polynomial which can be used. From Figs. 19 and 20 we see how an interpolating surrogate converges on an optimum. We stop short of saying that the surrogate *must* interpolate the data, since SVR and regressing Kriging and RBFs will converge towards the optimum of a noisy function to an accuracy determined by the noise, not by deficiencies in the surrogate. For a global search we need some form of error estimate for predictions made by the surrogate (coupled with the above requirements). Thus, of the methods reviewed in this paper, we are limited to the Gaussian process based methods, although Gutmann [69] has employed a one-stage goal seeking approach for a variety of RBFs.

We have not yet looked at when to stop the iterative process in Fig. 1. Choosing a suitable convergence criterion to determine when to stop the surrogate infill process is rather subjective. Goal seeking is the obvious winner in terms of convergence criteria and nothing need be added to the method itself. When choosing infill points based on minimizing the prediction (exploitation), the convergence criterion is simple: we stop when the change in a number of successive infill point objective values is small. Maximum error based infill (exploration) is likely to be stopped when some generalization error metric, e.g. cross-validation, drops below a certain threshold.

When using the probability or expectation of improvement, we can simply stop when the probability is very low or the expectation is smaller than a percentage of the range of observed objective function values. Care should, however, be taken since the estimated MSE of Gaussian process based models is often an under estimator and the search may be stopped prematurely. It is wise to set an overly stringent threshold and wait for a consistently low $P[I(\mathbf{x})]$, $E[I(\mathbf{x})]$ or $E[I(\mathbf{x}^*)]$.

When minimizing a lower bound there is no quantitative indicator of convergence and we are limited to the convergence criteria used for exploitation. Unfortunately, an infill strategy may dwell in the region of a local minima before jumping to another so we cannot guarantee that a series of similar objective values means that the global optimum has been found.

In many real engineering problems we actually stop when we run out of available time or resources, dictated by design cycle scheduling or costs.

Final thoughts: The above discussion gives no definitive answers, and deliberately so. This is because a method which is universally better than all others is yet to present itself. While we wait for it to do so, we must choose our surrogate-based optimization methodology carefully. Although exact choice of methodology may be problem dependent, one underlying trait that any surrogate-based optimization must include is some form of repetitive search and infill process to ensure the surrogate is accurate in regions of interest. Other considerations in terms of model selection, validation and infill criteria are secondary to this key requirement

Acknowledgements

We are grateful for the input and advice of Danie Krige, Donald Jones, András Sóbester, Prasanth Nair and Rafi Haftka.

References

[1] Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Tucker PK. Surrogate-based analysis and optimization. Progress in Aerospace Sciences 2005;41:1–28.

[2] Simpson TW, Toropov V, Balabanov V, Viana FAC. Design and analysis of computer experiments in multidisciplinary design optimization: a review of how we have come—or not. In: 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, British Colombia, 10–12 September, 2008.

[3] Sacks J, Welch WJ, Mitchell TJ, Wynn H. Design and analysis of computer experiments. Statistical Science 1989;4(4):409–23.

[4] Morris MD. Factorial sampling plans for preliminary computational experiments. Technometrics 1991;33(2):161–74.

[5] Johnson ME, Moore LM, Ylvisaker D. Minimax and maximin distance designs. Journal of Statistical Planning and Inference 1990;26:131–48.

[6] McKay MD, Beckman RJ, Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 1979;21(2):239–45.

[7] Morris MD, Mitchell TJ. Exploratory designs for computational experiments. Journal of Statistical Planning and Inference 1995;43:381–402.

[8] Forrester AIJ, Sóbester A, Keane AJ. Engineering design via surrogate modelling: a practical guide. Chichester: Wiley; 2008.

[9] Hastie T, Tibshirani R, Friedman J. The elements of statistical learning. New York: Springer; 2001.

[10] Box EP, Draper NR. Empirical model building and response surfaces. New York: Wiley; 1987.

[11] Cherkassky V, Mulier F. Learning from data—concepts, theory, and methods. New York: Wiley; 1998.

[12] Ralston A, Rabinowitz P. A first course in numerical analysis. New York: McGraw-Hill; 1978.

[13] Myers RH, Montgomery DC. Response surface methodology: process and product optimization using designed experiments. New York: Wiley; 1995.

[14] Goel T, Haftka R. Comparing error estimation measures for polynomial and Kriging approximation of noise-free functions. Structural and Multidisciplinary Optimization 2008; in press, doi:10.1007/s00158-008-0290-z.

[15] Lancaster P, Salkauskas K. Surfaces generated by moving least squares methods. Mathematics of Computation 1981;37(155):141–58.

[16] Levin D. The approximation power of moving least-squares. Mathematics of Computation 1998;67(224):1517–31.

[17] Aitken AC. On least squares and linear combinations of observations. Proceedings of the Royal Society of Edinburgh 1935;55:42–8.

[18] Toropov VV, Schramm U, Sahai A, Jones RD, Zeguer T. Design optimization and stochastic analysis based on the moving least squares method. In: 6th World congress of structural and multidisciplinary optimization, Rio de Janeiro, 30th May–3rd June, 2005.

[19] Kim C, Wang S, Choi KK. Efficient response surface modeling by using moving least-squares method and sensitivity. AIAA Journal 2005;43(11):2404–11.

[20] Ho SL, Yang S, Ni P, Wong HC. Developments of an efficient global optimal design technique—a combined approach of MLS and SA algorithm. COMPEL 2002;21(4):604–14.

[21] Broomhead DS, Loewe D. Multivariate functional interpolation and adaptive networks. Complex Systems 1988;2:321–55.

[22] Sóbester A. Enhancements to global optimisation. PhD thesis, University of Southampton, Southampton, UK; October 2003.

[23] Vapnik V. Statistical learning theory. New York: Wiley; 1998.

[24] Keane AJ, Nair PB. Computational approaches to aerospace design: the pursuit of excellence. Chichester: Wiley; 2005.

[25] Gibbs MN. Bayesian Gaussian processes for regression and classification. Dphil dissertation, University of Cambridge; 1997.

[26] Poggio T, Girosi F. Regularization algorithms for learning that are equivalent to multilayer networks. Science 1990;247:978–82.

[27] Orr M. Regularisation in the selection of RBF centres. Neural Computation 1995;7(3):606–23.

[28] Cook RD, Nachtsheim CJ. A comparison of algorithms for constructing exact D-optimal designs. Technometrics 1980;22(3):315.

[29] Keane AJ. Design search and optimisation using radial basis functions with regression capabilities. In: Parmee IC, editor. Proceedings of the conference on adaptive computing in design and manufacture, vol. VI. Berlin: Springer; 2004. p. 39–49.

[30] Matheron G. Principles of geostatistics. Economic Geology 1963;58:1246–66.

[31] Krige DG. A statistical approach to some basic mine valuation problems on the Witwatersrand. Journal of the Chemical, Metallurgical and Mining Engineering Society of South Africa 1951;52(6):119–39.

[32] Jones DR. A taxonomy of global optimization methods based on response surfaces. Journal of Global Optimisation 2001;21:345–83.

[33] Toal DJJ, Bressloff NW, Keane AJ. Kriging hyperparameter tuning strategies. AIAA Journal 2008;46(5):1240–52.

[34] Zhang Y, Leithead WE. Exploiting hessian matrix and trust-region algorithm in hyperparameters estimation of gaussian process. Applied Mathematics and Computation 2005;171:1264–81.

[35] Keane AJ. Wing optimization using design of experiment, response surface, and data fusion methods. Journal of Aircraft 2003;40(4):741–50.

[36] Cressie NAC. Statistics for spatial data, probability and mathematical statistics. revised ed. New York: Wiley; 1993.

[37] Joseph VR, Hung Y, Sudjianto A. Blind Kriging: a new method for developing metamodels. ASME Journal of Mechanical Design 2008;130.

[38] Joseph VR. A Bayesian approach to the design and analysis of fractional experiments. Technometrics 2006;48:219–29.

[39] Wu CFJ, Hamada M. Experiments: planning, analysis, and parameter design optimization. New York: Wiley; 2000.

[40] Hoyle N. Automated multi-stage geometry parameterization of internal fluid flow applications. PhD thesis, University of Southampton, Southampton, UK; 2006.

[41] Forrester AIJ, Keane AJ, Bressloff NW. Design and analysis of 'noisy' computer experiments. AIAA Journal 2006;44(10):2331–9.

[42] Vapnik V. The nature of statistical learning theory. New York: Springer; 1995.

[43] Schölkopf B, Smola AJ. Learning with kernels. Cambridge, MA: MIT; 2002.

[44] Smola AJ, Schölkopf B. A tutorial on support vector regression. Statistics and Computing 2004;14:199–222.

[45] Clarke SM, Griebsch JH, Simpson TW. Analysis of support vector regression for approximation of complex engineering analyses. Journal of Mechanical Design 2005;127.

[46] Squire W, Trapp G. Using complex variables to estimate derivatives of real functions. SIAM Review 1998;40:110–2.

[47] Griewank A. Evaluating derivatives: principles and techniques of algorithmic differentiation. In: Frontiers in applied mathematics. Philadelphia: SIAM; 2000.

[48] Giles MB, Pierce NA. An introduction to the adjoint approach to design. Flow, Turbulence and Combustion 2000;65:393–2000.

[49] Barthelemy BM, Haftka RT, Cohen GA. Physically based sensitivity derivatives for structural analysis programs. Computational Mechanics 1989;4(6): 465–76.

[50] van Keulen F, Vervenne K. Gradient-enhanced response surface building. Structural and Multidisciplinary Optimization 2004;27:337–51.

[51] Santner TJ, Williams BJ, Notz WI. Design and analysis of computer experiments. In: Springer series in statistics. Berlin: Springer; 2003.

[52] Leary SJ, Bhaskar A, Keane AJ. A knowledge-based approach to response surface modelling in multifidelity optimization. Journal of Global Optimization 2003;26(3):297–319.

[53] Forrester AIJ, Bressloff NW, Keane AJ. Optimization using surrogate models and partially converged computational fluid dynamics simulations. Proceedings of the Royal Society A 2006;462(2071):2177–204.
[54] Bandler J, Cheng Q, Dakroury S, Mohamed A, Bakr M, Madsen K, et al. Space mapping: the state of the art. IEEE Transactions on Microwave Theory and Techniques 2004;52:337–61.
[55] Hevesi J, Flint A, Istok J. Precipitation estimation in mountainous terrain using multivariate geostatistics. Part II: isohyetal maps. Journal of Applied Meteorology 1992;31:677–88.
[56] Kennedy MC, O'Hagan A. Predicting the output from complex computer code when fast approximations are available. Biometrika 2000;87(1):1–13.
[57] Forrester AIJ, Sóbester A, Keane AJ. Multi-fidelity optimization via surrogate modelling. Proceedings of the Royal Society A 2007;463(2088):3251–69.
[58] Jones DR, Schlonlau M, Welch WJ. Efficient global optimisation of expensive black-box functions. Journal of Global Optimisation 1998;13:455–92.
[59] Alexandrov N, Dennis JE, Lewis RM, Torczon V. A trust region framework for managing the use of approximation models in optimization. Structural Optimization 1998;15:16–23.
[60] Haftka RT. Combining global and local approximations. AIAA Journal 1991;29(9):1523–5.
[61] Eldred MS, Giunta AA, Collis SS. Second-order corrections for surrogate-based optimization with model hierarchies. In: 10th AIAA/ISSMO multidisciplinary analysis and optimization conference, Albany, New York, 30–31 August 2004.
[62] Dennis JE, Schnabel RB. Numerical methods for unconstrained optimization and nonlinear equations. Englewood Cliffs, NJ: Prentice-Hall; 1983.
[63] Alexandrov NM, Lewis RM, Gumbert CR, Green LL, Newman PA. Approximation and model management in aerodynamic optimization with variable-fidelity models. Journal of Aircraft 2001;38(6):1093–101.
[64] Sobol IM. On the systematic search in a hypercube. SIAM Journal of Numerical Analysis 1979;16:790–3.
[65] Statnikov RB, Matusov JB. Multicriteria optimization and engineering: theory and practice. New York: Chapman & Hall; 1995.
[66] Xiong Y, Chen W, Apley D, Ding X. A non-stationary covariance-based Kriging method for metamodelling in engineering design. International Journal for Numerical Methods in Engineering 2007;71:733–56.
[67] Holden C. Visualization methodologies in aircraft design optimization. PhD thesis, University of Southampton, Southampton, UK; January 2004.
[68] Jones DR, Welch WJ. Global optimization using response surfaces. In: Fifth SIAM conference on optimization, Victoria, Canada, 20–22 May, 1996.
[69] Gutmann HM. A radial basis function method for global optimization. Journal of Global Optimization 2001;19(3):201–27.
[70] Sóbester A, Leary SJ, Keane AJ. On the design of optimization strategies based on global response surface approximation models. Journal of Global Optimization 2005;33:31–59.
[71] Forrester AIJ. Efficient global optimisation using expensive CFD simulations. PhD thesis, University of Southampton, Southampton, UK; November 2004.
[72] den Hertog D, Kleijnen JPC, Siem AYD. The correct Kriging variance estimated by bootstrapping. Journal of the Operational Research Society 2006; 57(4):400–9.
[73] Forrester AIJ, Jones DR. Global optimization of deceptive functions with sparse sampling. In: 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, British Colombia, 10–12 September 2008.
[74] Sóbester A, Leary SJ, Keane AJ. A parallel updating scheme for approximating and optimizing high fidelity computer simulations. Structural and multidisciplinary optimization 2004;27:371–83.
[75] Schonlau M. Computer experiments and global optimization. PhD thesis, University of Waterloo, Waterloo, Ontario, Canada; 1997.
[76] Siddall JN. Optimal engineering design: principles and applications. New York: Marcel Dekker; 1982.
[77] Forrester AIJ, Sóbester A, Keane AJ. Optimization with missing data. Proceedings of the Royal Society A 2006;462(2067):935–45.
[78] Fonseca CM, Fleming PJ. An overview of evolutionary algorithms in multiobjective optimization. IEEE Transactions on Evolutionary Computation 1995;3(1):1–16.
[79] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 2002;6(2):182–97.
[80] Wilson B, Cappelleri D, Simpson W, Frecker M. Efficient Pareto frontier exploration using surrogate approximations. Optimization and Engineering 2001;2:31–50.
[81] Knowles J, Hughes EJ. Multiobjective optimization on a budget of 250 evaluations. In: Coello C, et al., editor. Evolutionary multi-criterion optimization (EMO-2005). Lecture notes in computer science, vol. 3410. Berlin: Springer; 2005.
[82] Voutchkov II, Keane AJ. Multi-objective optimization using surrogates. In: Proceedings of the 7th international conference on adaptive computing in design and manufacture, Bristol, 2006. p. 167–75 (ACDM 2006, ISBN 0-9552885-0-9).
[83] Keane AJ. Statistical improvement criteria for use in multiobjective design optimization. AIAA Journal 2006;44(4):879–91.
[84] Sóbester A, Leary SJ, Keane AJ. A parallel updating scheme for approximating and optimizing high fidelity computer simulations. In: 3rd ISSMO/AIAA internet conference on approximations in optimization, 2002.
[85] Efron B. Estimating the error rate of a prediction rule: improvement on cross-validation. Journal of the American Statistical Association 1983;78(382): 316–31.
[86] Goel T, Haftka R, Shyy W, Queipo NV. Ensemble of surrogates. Structural and Multidisciplinary Optimization 2007;33:199–216.
[87] Tresp V. A Bayesian committee machine. Neural Computation 2000;12: 2719–41.