Engineering with Computers (2007) 23:155–174
DOI 10.1007/s00366-006-0051-9

ORIGINAL ARTICLE

Multidimensional sequential sampling for NURBs-based metamodel development

Cameron J. Turner · Richard H. Crawford · Matthew I. Campbell

Received: 18 October 2005 / Accepted: 17 August 2006 / Published online: 24 November 2006

Abstract Adaptive design of experiments approaches are intended to overcome the limitations of a priori experimental design by adapting to the results of prior runs so that subsequent runs yield more significant information. Such approaches are valuable in engineering applications with metamodels, where efficiently collecting a dataset to define an unknown function is important. While a variety of approaches have been proposed, most techniques are limited to sampling for only one phenomenon at a time. We propose a multicriteria optimization approach that effectively simultaneously samples for multiple phenomena. In addition to determining the next sequential sampling point, such an algorithm also can be formulated to support conclusions about the adequacy of the experiment through the use of convergence criteria. A multicriteria adaptive sequential sampling algorithm, along with convergence metrics, is defined and demonstrated on five trial problems of engineering interest. The results of these five problems demonstrate that a multicriteria sequential sampling approach is a useful engineering tool for modeling engineering design spaces using NURBs-based metamodels.

Keywords Non uniform rational B-splines · Metamodels · Sequential sampling · Sampling criteria · Cooling schedule · Multicriteria sampling

This paper is approved for release by Los Alamos National Laboratory under LA-UR-06-6651 and is based upon research originally presented at the 2003 and 2004 ASME IDETC/CIE Conferences. Funding provided by Los Alamos National Laboratory and the National Science Foundation under Grant No. DMI-0323838. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Los Alamos National Laboratory.

C. J. Turner (✉)
Plutonium Manufacturing and Technology Division,
Los Alamos National Laboratory, P.O. Box 1663,
MS J580, Los Alamos, NM 87545, USA
e-mail: cturner@alumni.utexas.net

R. H. Crawford · M. I. Campbell
Department of Mechanical Engineering,
The University of Texas at Austin, 1 University Station,
C2200, Austin, TX 78712-0292, USA

R. H. Crawford
e-mail: rhc@mail.utexas.edu

M. I. Campbell
e-mail: mc1@mail.utexas.edu

1 Introduction

Modern engineering design usually involves the use of simulations and experiments during the product realization process. The overall goal of a set of experiments or simulations is to develop information about the underlying design space behavior. Ideally, a sufficient number of simulations or experiments will be performed to clearly define the relationships among the design variables and performance indices that define the design space. But in many cases, the cost of experiments and simulations precludes extensive design space surveys. This cost is particularly acute in high dimensional problems where even 2^k searches are costly.

Experimental design approaches are intended to make the most out of a limited number of trials, determined by the budget for the experiment [1–3]. Traditional a priori experimental designs do not adapt to information gained from previous experimental trials. Consequently, adaptive sequential sampling approaches have been developed to replace or supplement classical experimental designs. Many of these approaches are applicable to both physical experiments or to computer simulations.

The results of a set of experimental or simulation trials are often reduced to a mathematical model known as a metamodel. The metamodel provides a convenient representation of the design space with a single mathematical expression. Adaptive sequential sampling techniques commonly incorporate metamodels. Many sequential sampling techniques are limited by consideration of only one aspect of the metamodel under construction, and do not provide feedback on the metamodel quality as it is created.

1.1 Motivation

The motivation for this research is to develop an adaptive sequential sampling algorithm for a metamodel defined with Non Uniform Rational B-splines (NURBs). Our NURBs-based metamodel is referred to as a Hyperdimensional Performance Model or HyPerModel. The design and application of HyPerModels is presented in detail in Turner [4]. One of the seminal challenges in defining a HyPerModel (and other types of metamodels) is obtaining sufficient data to accurately represent the design space. Adaptive sequential sampling techniques are often used to obtain the necessary data.

In developing a suitable algorithm for HyPerModels, called Hyperdimensional Performance Sample (HyPerSample), we sought to include several features, including:

- the incorporation of multiple criteria for simultaneous consideration through the use of a cooling schedule;
- the use of information contained within the HyPerModel representation to improve criteria evaluation efficiency; and
- the incorporation of convergence metrics intended to terminate the search when a HyPerModel representation of user-defined accuracy is obtained or to provide justification for additional experiments or simulations when the data budget is exhausted.

In this paper we provide a short description of HyPerModels and a review of sequential sampling approaches. The underlying theory of the multicriteria approach follows, with results from several different trial problems used to demonstrate the effectiveness of this approach.

1.2 HyPerModels: NURBs-based metamodels

Many Computer-Aided Design/Engineering (CAD/CAE) software systems use NURBs-based representations to describe geometric objects. Thus, there are many excellent resources with thorough descriptions of the behavior and implementation of NURBs, including Rogers and Adams [5], Piegl and Tiller [6] and Cohen et al. [7]. Most references focus on the NURBs curve and surface representations, such as those commonly employed in CAD/CAE applications. Higher dimensional forms are discussed in Cohen et al. [7], but a metamodeling implementation does not appear in the literature until the work of Turner et al. [8, 9].

Since the mathematics of HyPerModels underlies the formulations of the criteria used in HyPerSample, the fundamentals of HyPerModels are defined in this section. Additional detail is available in Turner [4]. For simplicity, the necessary functions are defined as 1D input 1D output curves, which can be readily extended to higher dimensions via tensor products. Eq. 1 defines a 1D input 1D output HyPerModel with a planar NURBs curve, $\mathbf{p}(u)$:

$$\mathbf{p}(u) = \frac{\sum_{i=1}^{n_C} \mathbf{b}_i w_i N_{i,k}(u)}{\sum_{i=1}^{n_C} w_i N_{i,k}(u)} \quad \text{for } a \leqslant u \leqslant b \tag{1}$$

where $\mathbf{b}_i$ is the ith of n_C control points, w_i is a positive scalar defining the weight of the ith control point, and $N_{i,k}(u)$ is the ith B-spline basis function of order k given as a function of u (the HyPerModel input dimension). The parameter u defines a position along the curve length, which is equivalent to a point on the curve defined by the vector $\mathbf{p}(u)$ (the HyPerModel output dimension). The B-spline basis function, $N_{i,k}(u)$, is a recursive function defined by Eqs. 2 and 3,

$$\begin{aligned} N_{i,k}(u) = {} & \left(\frac{u - \mathbf{x}_i}{\mathbf{x}_{i+k-1} - \mathbf{x}_i}\right) N_{i,k-1}(u) \\ & + \left(\frac{\mathbf{x}_{i+k} - u}{\mathbf{x}_{i+k} - \mathbf{x}_{i+1}}\right) N_{i+1,k-1}(u), \end{aligned} \tag{2}$$

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } \mathbf{x}_i \leqslant u < \mathbf{x}_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where $\mathbf{x}_i$ is the ith element in the knot vector, a sequence of parameter values defining the regions of control point influence within the HyPerModel. For the ith control point, that region of influence is defined by the knot vector, $\mathbf{x}$, and the metamodel order, k. The B-spline basis function exhibits the behaviors defined by Eqs. 4, 5, 6, and 7.

$$\sum_{i=1}^{n+1} N_{i,k}(u) \equiv 1 \quad \forall k \text{ and } u \tag{4}$$

subject to the constraints $2 \leqslant k \leqslant n_C$ (5)

$$0 < w_{\min} \leqslant w_i \leqslant w_{\max} \leqslant 1 \tag{6}$$

and to the definition $\frac{0}{0} \equiv 0$ (7)

HyPerModels use the control point locations, control point weights (effectively a homogeneous coordinate of the control point), open knot vectors, and the curve order, k, to produce a highly flexible curve definition [10].

Control points are a key factor in understanding HyPerModel behavior. Control points collectively bound the HyPerModel in space, and each control point attracts the HyPerModel towards its location. The control point region of influence is derived through the B-spline basis function, which is ultimately governed by the knot vector $\mathbf{x}$ and curve order k. Finally, the control point weight, w, regulates the strength of control point influence on the HyPerModel. These parameters are determined with the fitting techniques described in Turner [4] and Turner et al. [9].

2 Adaptive sequential sampling techniques

Rather than establish a rigid experimental design a priori, adaptive sequential sampling techniques alter the experimental design between trials. An experimental design method, such as full and partial factorial searches or latin hypercubes, may still be used as the initial foundation for a sequential sampling method, but subsequent sampling locations are determined adaptively. The fundamental concept behind an adaptive sequential sampling technique is to use results from prior experiments to determine subsequent experiments.

Many sequential sampling techniques involve a metamodel, either as an iterative result or as an ultimate output of the experiment or simulation. For instance, Farhang-Mehr and Azarm [11], Osio and Amon [12], Martin and Simpson [13], Wang and Simpson [14], Sasena et al. [15] and Sasena [16], all use kriging metamodels in their sequential sampling algorithms. Perez et al. [17] uses response surface models (RSM), while Gutmann [18] and Jin et al. [19] apply sequential sampling to radial basis function (RBF) models. While there are distinct differences between these approaches, the fundamental method is quite similar:

1. determine what data to collect;
2. collect new data;
3. revise the metamodel as necessary; and
4. repeat until the experimental data budget is exhausted.

In addition, these algorithms all employ various nonlinear optimization algorithms to determine the next points to sample in the design space. The most significant distinctions between different algorithms are in the criteria employed to determine the design space location of the next samples, and whether one or more data points are sampled in a given metamodel iteration.

2.1 Prior work in sequential sampling

Various authors have identified a variety of criteria. For instance, Kushner's criterion [20] applies a Gaussian cumulative distribution function to identify the point with the greatest probability of improving upon the best point in the model. Other authors, including Locatelli [21], Mockus [22] and Schonlau [23], subsequently proposed variations of Kushner's criterion. Cox and John [24] proposed an alternative criterion based on lower confidence bounding, Osio and Amon [12] developed a data adaptive optimal sampling criterion, and Farhang-Mehr and Azarm [25] proposed a maximum entropy criterion, while Watson and Barnes [26] developed a set of thresholding criteria for geological contamination studies. Many criteria originally developed for geologic exploration are associated with kriging metamodels commonly employed in geostatistics. These criteria can be classified as either global or local criteria, based on whether they focus on improving the global metamodel fit to the entire dataset or improving the metamodel fit in a particular region of interest.

Sasena et al. [15] and Sasena [16] reviewed a number of criteria in conjunction with a kriging metamodel, although many are adaptable to other metamodel types. Sasena found that the best and most consistent technique is a switching criterion (Switch). Switch initially searches globally, attempting to reduce the maximum variance of the metamodel. Switch then becomes a local search in the neighborhood of the minimum point until the search stalls, and then reverts back to a global search. Switch alternates between two different criteria. Thus, Switch is similar to a cooling schedule approach, a concept adapted from the stochastic optimization method Simulated Annealing [27, 28].

In simulated annealing, a cooling schedule is applied during each optimization run to guide the optimization algorithm to a local optimum. The most successful cooling schedules applied in simulated annealing [29] are nonmonotonic, with alternating periods of heating and cooling, as suggested by Lundy and Mees [30] and Osman [31]. The cooling schedule approach studied by Sasena et al. [15] and Sasena [16] is a monotonic cooling schedule, applied not to the individual optimizations but instead applied to the entire data collection process. Switch represents a nonmonotonic cooling schedule that instantaneously changes between sampling goals. A cooling schedule approach that smoothly transitions between sequential sampling objective function goals as the samples are collected was not tested.

Also note that finding minima or maxima is not always the sequential sampling goal. There could be other areas of interest, such as regions of rapid change or regions where a discontinuity exists that may be the focus of the local search. For instance, Friedman [32] suggested a sequential sampling approach that focused on regions of rapid change, such as those associated with a discontinuity. Similarly, Farhang-Mehr and Azarm [11] suggested a modified form of a maximum entropy criterion that is biased towards regions of rapid change (often associated with regions with multiple local optima) as an improvement over their maximum entropy criterion [25].

Comparable sequential sampling approaches have also been developed for other metamodel types. Sasena et al. [15] and Sasena [16] uses a kriging model and an algorithm called SuperEGO to solve an automotive design problem. SuperEGO is a Bayesian optimization algorithm developed from the efficient global optimization (EGO) algorithm [33]. SuperEGO and EGO adaptively refine the dataset used to define the metamodel by identifying additional points for data collection using a criterion to model and eliminate regions of uncertainty in an associated metamodel. EGO and SuperEGO sample optimal metamodel regions so that the optimum location is sampled directly from original data sources. Jones [34] identified several interpolating kriging models that support the EGO algorithm. Martin and Simpson [13] and Wang and Simpson [14] propose similar adaptive metamodeling techniques for kriging models to achieve the same end using mean square error and fuzzy clustering respectively. Gutmann [18] suggested a similar approach using a utility function for RBF metamodels. Jin et al. [19] proposes the Maximum Scaled Distance and Cross-Validation approaches for both kriging and RBF metamodels.

In some of these studies, (for instance Farhang-Mehr and Azarm [11] and Osio and Amon [12]) sets of sampled points were collected between metamodel updates. This makes considerable sense in an experimental setting where it may be easier to run several experimental trials in close succession. However, in other studies (notably Sasena et al. [15] and Sasena [16]), data points were collected individually and the metamodel was updated at each step. Sasena [16] discusses the collection of a single point per iteration versus multiple points per iteration as a current area of research interest within the adaptive sequential sampling community.

2.2 Multicriteria sequential sampling

Adaptive sequential sampling techniques for metamodels may need to pursue multiple sampling goals simultaneously. Samples should be collected so as to provide an accurate survey of the breadth of the entire design space such that local features are not missed and the metamodel error is reduced, particularly in regions of interest. These regions of interest may include extrema, or regions that delineate critical changes in the behavior of the underlying function. For instance, regions of rapid change suggest that the behavior of the underlying function should be characterized carefully. Regions with little curvature may be of interest for design problems, denoting robust solution regions. Other regions of the design space may play significant roles in the definition of the metamodel and should be defined carefully.

The chief issue with formulating the sequential sampling problem as a multicriteria optimization problem is deciding how to combine the criteria into an objective function. For two criteria, either averaging the criteria (as in a cooling schedule) or switching between criteria, as in Switch, are reasonable approaches and have been used successfully [16]. As the number of criteria increases, the cooling schedule approach becomes more attractive. A cooling schedule, similar to that defined for simulated annealing optimization problems, allows the sequential sampling algorithm to smoothly transition between different sequential sampling priorities. To do this, a set of barycentric criteria weights (that always sum to 1) within the range defined by the schedule are defined. However, this is only effective if the criteria are normalized; otherwise, the relative magnitudes of the criteria values will compromise the blending provided by the cooling schedule. One option to achieve a barycentric set of criteria weights in the cooling schedule is to use Bernstein basis functions, defined in Eqs. 8 and 9:

$$J_i(t) = \frac{(n-1)!}{i!(n-i-1)!} t^i (1-t)^{n-i-1} \quad \forall\, 0 \leqslant t \leqslant 1 \tag{8}$$

such that:

$$\sum_{i=0}^{n-1} J_i(t) \equiv 1 \quad \forall\, t \tag{9}$$

where $J_i(t)$ is the ith criteria weight, n is the number of criteria defined in the cooling schedule, and t is the parameterized sample number defined by Eq. 10 as:

$$t = \frac{S_n - 1}{CS_p - 1} \tag{10}$$

where S_n is the current sample number and CS_p is the cooling schedule period, a user defined parameter denoting the cooling schedule duration. A cooling schedule may be cycled through several times during a sequential sampling experiment (creating nonmonotonic behavior). With this formulation, the criteria weights defined from the Bernstein basis functions exhibit a Partition of Unity property (resulting in a normalized objective function when the criteria also are normalized), with symmetric functions that have equally spaced maximum locations [5]. As a result each criterion is dominant over an equal period of the cooling schedule. Finally, this selection takes advantage of the mathematical relationship of Bernstein basis function to the B-spline basis function to introduce programming efficiencies into the HyPerSample algorithm. Note that other basis functions also are feasible selections and may offer advantages over the Bernstein basis. Further research is necessary to determine the implications of alternative selections.

Consider a four criteria implementation. The cooling schedule defines a set of criteria weights, J_0, J_1, J_2, and J_3 that are combined with four criteria, in this case the proximity (P_C), weight (W_C), slope magnitude (S_C), and a model (M_C) criteria respectively (which are defined in the following section), to define the objective function, *f(x)*:

$$\begin{aligned} f(x,t) &= J_0(t)P_C(x) + J_1(t)W_C(x) \\ &\quad + J_2(t)S_C(x) + J_3(t)M_C(x). \end{aligned} \tag{11}$$

The resulting cooling schedule is shown in Fig. 1. Note that the result of this approach is an objective function that remains constant while each optimization problem is solved, but is modified with each sample collected. Since the cooling schedule repeats after completing a cycle, its behavior is nonmonotonic.

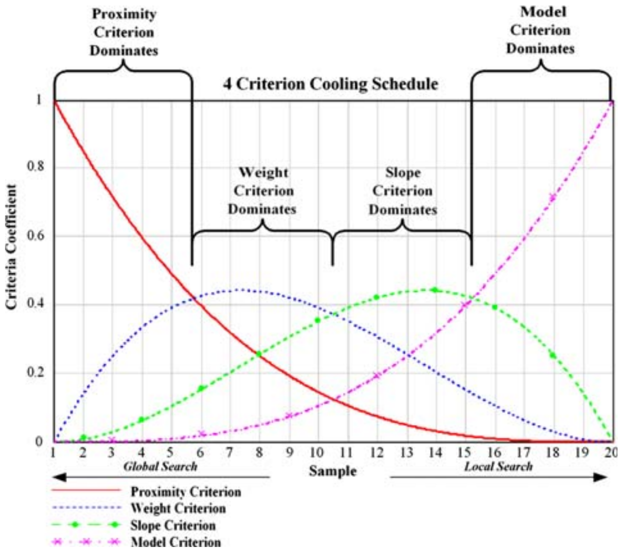


Fig. 1 A four criteria cooling schedule example

2.3 Multidimensional sampling criteria

It is possible to use variations of the criteria described in Sasena et al. [15] and Sasena [16] to define a multicriteria sequential sampling optimization problem with a cooling schedule. We have carefully defined four criteria (proximity, P_C, weight, W_C, slope magnitude, S_C, and model, M_C) that are evaluated using the unique properties of HyPerModels to reduce computational costs.

In general, as the HyPerModel is constructed, control points are located in proximity to existing data points. Therefore, it is reasonable to use the proximity of control points to determine if a region has been previously sampled. The proximity criterion defines a parabolic span between control point locations, much like a cable between towers on a suspension bridge, with the parabola depth determined by the relative control point spacing. The coefficients for each local polynomial segment can be calculated in closed form, leading to a computationally efficient criterion despite its piecewise construction. For instance, a simple 1D input 1D output HyPerModel with three control points and an order of k = 3, consists of two parabolic spans. Thus, the value of the proximity criterion at a point u, is defined by Eqs. 12, 13, 14, 15, and 16. Given a value of u such that

$$\mathrm{CP_j} \leqslant u < \mathrm{CP_{j+1}} \tag{12}$$

$$P_C = 1.0 - \left(1.0 - Au^2 - Bu - C\right)/\mathrm{CP}_{\max} \quad (13)$$

$$A = -4.0/\left(\mathrm{CP_{j+1}} - \mathrm{CP_j}\right) \quad (14)$$

$$B = 4.0\left(\mathrm{CP_j} + \mathrm{CP_{j+1}}\right)/\left(\mathrm{CP_{j+1}} - \mathrm{CP_j}\right) \quad (15)$$

$$C = 1.0 - 4.0\left(\mathrm{CP_j}\right)\left(\mathrm{CP_{j+1}}\right)/\left(\mathrm{CP_{j+1}} - \mathrm{CP_j}\right) \quad (16)$$

where $\mathrm{CP_j}$ is the control point u-coordinate for the next smallest control point with respect to the position u, $\mathrm{CP_{j+1}}$ is the control point u-coordinate for the next larger control point with respect to the position u, and $\mathrm{CP_{max}}$ is the maximum difference between neighboring control points in the u-axis direction. By definition, the criterion is normalized. Because control points are arranged in an N-dimensional grid, the proximity criterion can be defined with a tensor product of the criteria for each dimension.

The weight criterion evaluates the confidence in control point locations. In a NURBs HyPerModel, the location of each control point is defined with respect to the nearest data point based only on the input coordinates of the two points. The more distance between a control point and its nearest neighboring data points, the less confidence there is in the location of the control point, and thus we reduce the control point weight (and therefore the impact of the control point on the HyPerModel). This is accomplished with Eq. 17.

$$w = w_{\min} + \left(w_{\max} - w_{\min}\right)\left(\mathbf{r}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{r}\right) \quad (17)$$

Recalling that the input dimensions are normalized and equal to the HyPerModel input coordinates,

$$u = x \in [0,1] \quad (18)$$

$$\text{and thus} \quad 0 \leqslant \left(\mathbf{r}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{r}\right) \leqslant 1 \quad (19)$$

$$\text{so therefore} \quad 0 < w_{\min} \leqslant w \leqslant w_{\max} \quad (20)$$

where w is the weight of the control point, $w_{\min}$ is the minimum weight value, $w_{\max}$ is the maximum weight value, $\mathbf{r}$ is a vector derived from the spatial correlation function,

$$R(u,x) = \mathrm{e}^{-\theta|u-x|^p} \quad (21)$$

where θ defines the range of influence of the data ($\theta > 0$), and p defines the smoothness of the model ($0 < p < 2$, where increasing values of p lead to a smoother model), and u and x represent the current point and the sampled data points [8]. In this case, $\mathbf{r}$ is derived from $R(u_i, x_{CP})$, relating the parametric control point location (x_{CP}) to the location of the ith nearby neighbor data points (u_i). The matrix $\mathbf{R}$ in Eq. 17 is derived from the spatial correlation function, $R(u_i, u_j)$, defined in Eq. 21, relating the location of data point u_i to the location of data point u_j. In this case, to limit the computational cost of inversion, $\mathbf{R}$ is based on the 10 nearest neighbors to each control point rather than the entire dataset. Based on a parametric study of the affect of θ and p on the calculated weights, it was determined that reasonable parameter values for θ and p can be obtained from $w_{\min}$ and the number of control points (in each input dimension direction), n_C, respectively, according to the relations defined in Eqs. 22 and 23:

$$\theta = \ln\left(w_{\min}\right) \quad (22)$$

$$p = \frac{\ln\left(\ln\left(C\right)\right)}{\ln\left(1/n_C\right)} \quad (23)$$

where C is a coefficient ($C > 1.0$) defining the minimum weight of influence at the nearby neighborhood boundary. Values of $w_{\min} = 0.1, w_{\max} = 1.0$, and $C = 2$ have yielded good results, and result in weights that range from 0.1 for a control point with little data near its location, to a value of 1.0 for a control point with many nearby and even coincident neighbors. The control point weight represents the proximity of a control point to existing data points and thus estimates the level of confidence that can be placed in a control point location. Control points with data in close proximity are well defined and exhibit weights near $w_{\max}$, while control points without nearby data points will exhibit weights near $w_{\min}$. This conclusion is also based upon the same parametric study used to establish the relations defined in Eqs. 22 and 23. Since a control point weight value of zero leads to numerical difficulties in Eq. 1, $w_{\min}$ was established a small nonzero value. Since the weight criterion, W_C, is defined to be normalized, the $w_{\max}$ was thus set at a value of 1. Consequently, the value of C determines the rate at which the proximity of data to a control point location reduces the control point weight, while Eqs. 22 and 23 affect the shape of that function. The weight criterion is calculated throughout the design space by treating the weights as the output coordinate for a linear B-spline object.

The model and slope criteria are derived directly from the HyPerModel formed from the current dataset iteration. Since a HyPerModel obeys the Convex Hull Property of NURBs, the control point network bounds the HyPerModel. Therefore, normalizing the control point locations also normalizes the HyPerModel.

Normalization is a perspective transformation, and thus, the HyPerMdoel with normalized control points is invariant (i.e., the optimum will not move due to the transformation) with respect to the transformation [5–7]. This transformation can be done for all of the possible output variables in the HyPerModel simultaneously. The resulting criterion may exhibit a reduced magnitude because the control point network varies from 0 to 1, while the curve will lie within those bounds. The model criterion is used to search for minima, maxima or extrema in the HyPerModel. As these locations are often of significance in design space modeling applications, additional effort to assure that these regions are well defined is reasonable. This is the physical goal of the model criterion. Since these goals are mutually exclusive, three variations of the model criterion have been defined to search for minima, maxima, or extrema, as in Eqs 24, 25, and 26:

$$\text{for minima:} \quad M_C(x) = m(x) \tag{24}$$

$$\text{for maxima:} \quad M_C(x) = 1 - m(x) \tag{25}$$

$$\text{for extrema:} \quad M_C(x) = m(x)[1 - m(x)] \tag{26}$$

where *m(x)* represents the normalized current HyPerModel of the unknown function. Each variation requires only a single evaluation of the HyPerModel.

The control point network also provides the key to normalizing the slope magnitude criterion. A HyPerModel will follow the shape of the control polygon because it is a barycentric expression of control point locations, with the slope between neighboring control points bounding the metamodel slope. It is very easy to find the maximum and minimum control point network slopes with a simple search of the control net. Consequently, both the model and slope magnitude criteria can be calculated and normalized with knowledge of the control point locations, control point weights, and metamodel curve order. Like the results from the model criterion, the normalization does not produce a perfect variation between 0 and 1.

Minima in the slope magnitude criterion indicate regions of rapid change within the HyPerModel. This could be due to the presence of multiple local optimal in close proximity, or due to local dramatic changes in the design space. Either may be of interest to a designer. The slope criterion is defined as:

$$S_C(x) = 1 - (m'(x))^2 \tag{27}$$

where $m'(x)$ is the normalized derivative of the HyPerModel. In an N-dimensional application, there are N slopes in each of the N input directions. The slope magnitude criterion is the tensor product of each of these slope magnitudes (which are individually normalized).

2.4 Sampling convergence

Adaptive sequential sampling approaches generally converge when the available experimental budget to collect data points has been exhausted or when the predicted cost of acquiring additional data exceeds the predicted value of that new data [15, 16]. This is because most metamodel approaches use the entire available dataset to define the metamodel. HyPerModels only use data points to define the HyPerModel if the data points exceed a specified root mean square (RMS) error threshold (normalized with respect to the full scale of the HyPerModel), i.e., data points that are not well represented by the current metamodel. This is only possible if an approximating HyPerModel is acceptable. In a HyPerModel, two RMS error thresholds are used. The minimum RMS error threshold (RMS min) defines when a HyPerModel prediction is close enough to a data point value to approximate that data point within a user-specified tolerance. The maximum RMS error threshold (RMS max) provides a user-specified tolerance above which the local errors in the HyPerModel outweigh the global accuracy of the HyPerModel.

The segregation of data into used and unused datasets represents an important development for adaptive sequential sampling algorithms. The unused data provides a mechanism to statistically evaluate the current HyPerModel against several convergence criteria including the maximum local RMS error and the global correlation coefficient, facilitating simultaneous metamodel validation and sampling. In other adaptive sequential sampling algorithms, model validation occurs after metamodel convergence is achieved by collecting additional data at additional cost if it occurs at all. Given a sufficient unused dataset size, an experiment can be determined to be complete before the experimental budget is exhausted or if the experimental budget is even adequate to estimate and validate the HyPerModel. The HyPerSample algorithm for determining sampling convergence is shown in Fig. 2.

Sampling convergence is determined by several user-defined parameters including the global correlation coefficient target threshold (Target), RMS min, RMS max, the cooling schedule duration (CS Limit), the minimum dataset size (minimum), and the experimental data budget (Data Budget). The global correlation coefficient measures how well the HyPerModel explains the variations in the unused dataset, but can

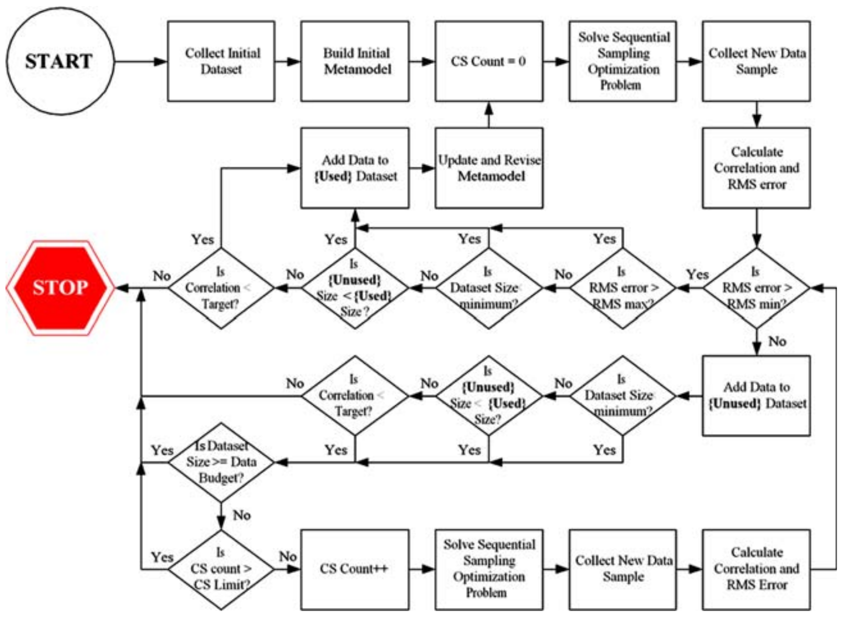

START
Collect Initial Dataset
Build Initial Metamodel
CS Count = 0
Solve Sequential Sampling Optimization Problem
Collect New Data Sample
Calculate Correlation and RMS error
Add Data to {Used} Dataset
Update and Revise Metamodel
STOP
Is Correlation < Target?
Is {Unused} Size < {Used} Size?
Is Dataset Size minimum?
Is RMS error > RMS max?
Is RMS error > RMS min?
Add Data to {Unused} Dataset
Is Dataset Size >= Data Budget?
Is CS count > CS Limit?
CS Count++
Calculate Correlation and RMS Error
Yes
No

Fig. 2 The HyPerSample algorithm

be ignored if the maximum RMS error threshold is exceeded at any point in the unused dataset. The remaining parameters terminate the algorithm if the experimental budget is exceeded (Data Budget), set a threshold for the minimum number of experiments (minimum) to be conducted before the statistical analysis is considered valid, and define the number of sampling attempts than can be made without updating the HyPerModel (CS Limit). Typical values for each parameter are shown in Table 1. With these parameters, convergence occurs due to:

1. *Convergence due to sampling limit*. The algorithm can exhaust its sampling budget without achieving its accuracy goals. When this happens, it can be argued that additional experimentation is justified to further refine and validate the model.
2. *Convergence due to cooling schedule limit*. If the number of consecutive samples collected without requiring a HyPerModel revision and refit equals the cooling schedule cycle duration, (20 samples in Fig. 1), the sequential sampling algorithm will begin to repeat previous results. In this case, random sampling, modifications to the sequential sampling criteria, or changes to the cooling schedule are warranted. In short, a different sampling approach is needed to continue the search.
3. *Convergence due to model correlation*. The algorithm can also converge by achieving its global correlation target with an RMS error less than the RMS max as calculated with the unused dataset.

2.5 Number of samples per iterations

HyPerSample can provide multiple sampling points per iteration, or only a single point. In general, the objective function resulting from the cooling schedule contains multiple local optima, which can be sampled simultaneously. However, in the current implementation, the best optimum location is selected as a "parent" point. In addition to the parent, multiple "child" points also are defined. The dimensionality of the

Table 1 Sequential sampling parameter values

Problem dimension	Target correlation	Min RMS limit	Max RMS limit	Cooling schedule reset limit	Min data sample	Experimental data budget	Initial dataset	
							Small	Large
1	99%	0.5% full scale	2.5% full scale	15	20	50	2	11
2				20	8 × 8	20 × 20	4	25
3				20	6 × 6 × 6	16 × 16 × 16	8	125

problem and the density of the control point grid determine the exact number of children in a given iteration. Furthermore, some of the child points may be eliminated from the final sampling set if they are in close proximity (determined with a user-defined threshold) to points that have been previously sampled. An example of this parent–child relationship is shown in Fig. 3.

Any of the above approaches can be used to generate data for a HyPerModel as a model is not dependent upon any particular data structure. However, because a HyPerModel uses a grid-based control point network, the use of a grid-based sampling scheme seems to reduce the number of iterations required to achieve convergence. However, further research also is needed on this topic.

3 Experimental trials

Two experimental questions are addressed through a set of experimental trials. The first question is whether HyPerSample does a better job representing unknown functions than a random search resulting in an equal number of sample points. The second question is if there is any benefit to a more extensive initial search to define the initial HyPerModel before proceeding with HyPerSample. To address these questions, the HyPerSample algorithm was implemented into the HyPerMaps software package developed to define HyPerModels and a set of five nonlinear functions was selected for testing.

Initially, two initial search sizes were undertaken. The smaller search was a 2^k search, which establishes the corners of the design space for each problem. A second larger search, consisting of equally spaced points (11 points for the 1D input problems, 25 points for the 2D input problems, and 125 points for the 3D problem) also was conducted. These initial searches provided data to define the initial HyPerModel enabling HyPerSample to define the initial search criteria. For each of the five functions, both a small and a large initial dataset were used to define the initial HyPerModel and then HyPerSample was used to collect additional data until one of the three convergence results introduced in Sect. 2.4 was achieved. Table 1 defines the parameters by which these trials were conducted.

These trials determined the number of data points needed by HyPerSample to model each function. Using these dataset sizes, five equally sized but randomly generated datasets were defined for each function. This produced 10 datasets for each function (5 for the HyPerModels generated by the small initial dataset and 5 for the HyPerModels generated from the large initial dataset). HyPerModels were then fit to each randomly generated dataset. The HyPerModels generated with HyPerSample and from randomly generated datasets were then compared to the actual functions to determine their global correlation coefficient (i.e. how similar are the HyPerModels to the actual function globally) and their local RMS error, expressed as a percentage of the full scale of the model (i.e. what is the largest local deviation between the HyPerModel and the actual function locally) using 101^k samples for the 1D and 2D functions and 41^k samples for the 3D function. The results of the five randomly generated datasets for each function and size were averaged. The use of replicates is intended to reduce the effect of a randomly good or randomly bad set of data points.

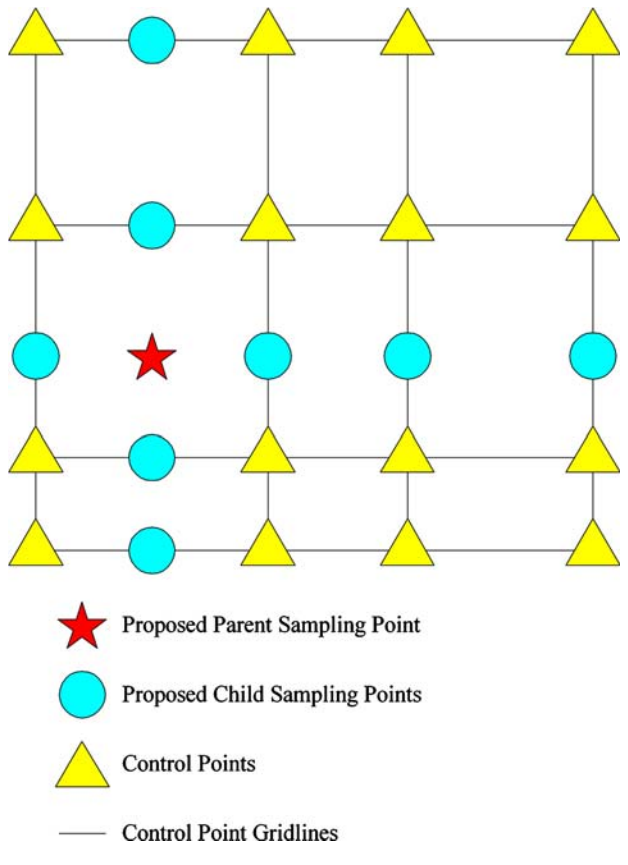

Proposed Parent Sampling Point
Proposed Child Sampling Points
Control Points
Control Point Gridlines

Fig. 3 A parent–child relationship determines the set of possible sampling points from a single optimum (parent) sampling location in HyPerSample

3.1 Algorithm implementation

Several details are notable in the implementation of HyPerSample within HyPerMaps. The sequential sampling optimization problem is solved with a sequential quadratic programming algorithm, using the control point locations as starting locations. To differentiate between local optima with similar objective function values, possible optimal solutions are compared to existing data point locations in the dataset and points that have already been sampled are eliminated from consideration.

In addition, the experimental data budget is treated as a ‘‘soft’’ limit. That is to say, when the dataset size equals or exceeds the experimental point budget, the search is terminated. However, the experimental point budget may be exceeded to complete sampling of the current point set during the current sampling iteration. This was done in order to simplify the implementation of HyPerSample. Thus it is possible to sample slightly more than 100% of the experimental budget. Future versions will implement a capability for a ‘‘hard’’ limit.

Finally, the 2D and 3D trials incorporate four sampling criteria, while the 1D trials used only three criteria (the weight criterion was not used in the 1D trials). These criteria are ordered as in Fig. 1 with the extrema form of the model criterion (Eq. 26). This order places globally oriented search criteria in the early stages of the cooling schedule and locally oriented search criteria in the later stages. Within the current implementation of HyPerSample, 1 to 4 criteria can be selected (although only one version of the model criterion at a time) and the user can specify their order in the cooling schedule. Further research into the impact of the order in which the criteria are considered and the number of criteria used is still needed.

3.2 Trial problems

The five test functions are discussed in detail in Sect. 4, including two 1D input–1D output functions, two 2D input–1D output functions, and a 3D input–6D output function. Two of these functions are directly related to physical engineering applications. The first function is based on the Jacobian Condition Number (JCN) of a planar 3 degree-of-freedom robot [35] shown in Fig. 4. Murphy et al. [36] proposed the Local Variability function shown in Fig. 5, defined with a sparse dataset, and represented with a quadratic polynomial blend defined by subsequent sets of three data points. Two more functions are nonlinear optimization test functions that are defined by Eq. 28, the 6-hump camel back function and by Eq. 29, the Hansen function [37].

$$f_{25}(x_0,x_1) = 4x_0^2 - 2.1x_0^4 + \frac{x_0^6}{3} + x_0x_1 - 4x_1^2 + 4x_1^4 \quad \text{for } -2 \leqslant x_0 \leqslant 2 \quad \text{for } -1 \leqslant x_1 \leqslant 1 \tag{28}$$

$$\begin{aligned} f_{26}(x_0,x_1) &= \begin{pmatrix} \cos(1) + 2\cos(x_0+2) + 3\cos(2x_0+3) \\ +4\cos(3x_0+4) + 5\cos(4x_0+5) \end{pmatrix} \\ &\times \begin{pmatrix} \cos(2x_1+1) + 2\cos(3x_1+2) + 3\cos(4x_1+3) \\ +4\cos(5x_1+4) + 5\cos(6x_1+5) \end{pmatrix} \\ &\text{for} \quad -3 \leqslant x_{0,1} \leqslant 3 \end{aligned} \tag{29}$$

The final example function is also based on a physical engineering application modeling the problem of locating and sizing a crane at a construction site. Conceptually, this problem is very similar to many

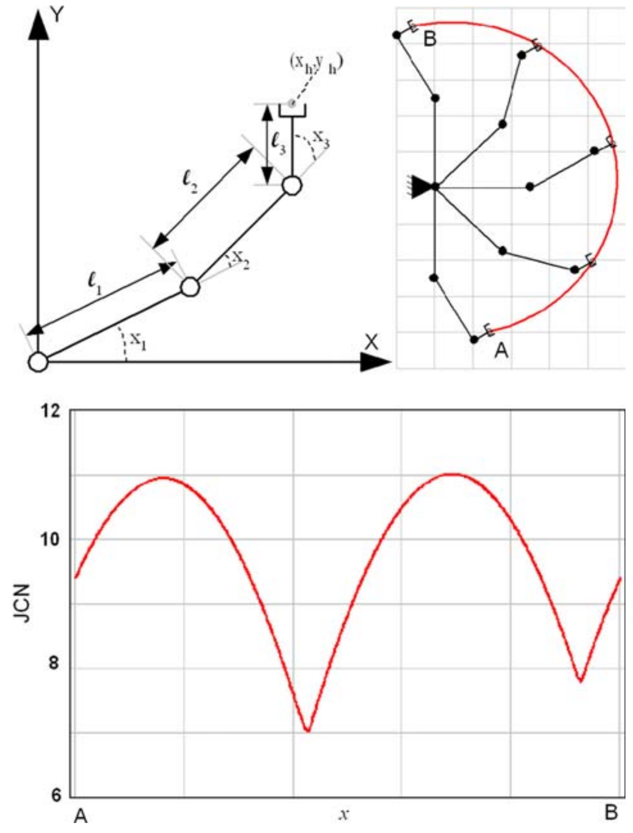


Fig. 4 An example problem constructed from a three degree-of-freedom planar robot, similar to those used in wafer handling, with a proposed path generated by the joint conditions that x_2 = 30° and $x_1 = -x_3$ so that the robot tool maintains a constant orientation of 30° to the horizontal. Small values of the JCN indicate that the robot is approaching undesirable singular configurations along its path from A to B [4]

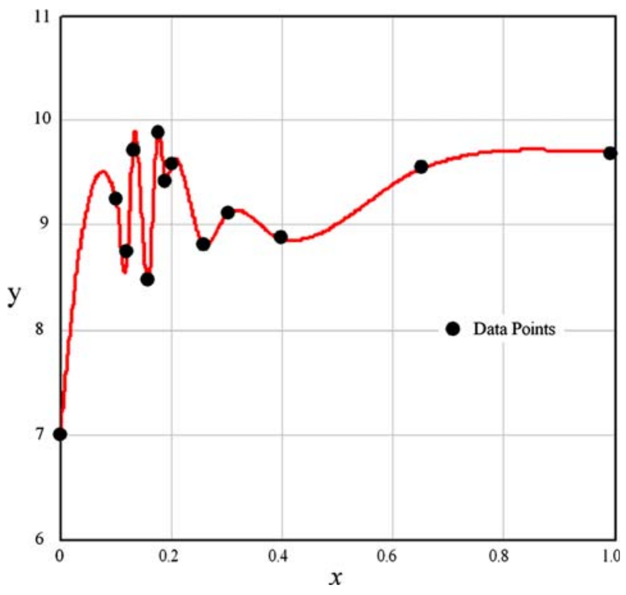
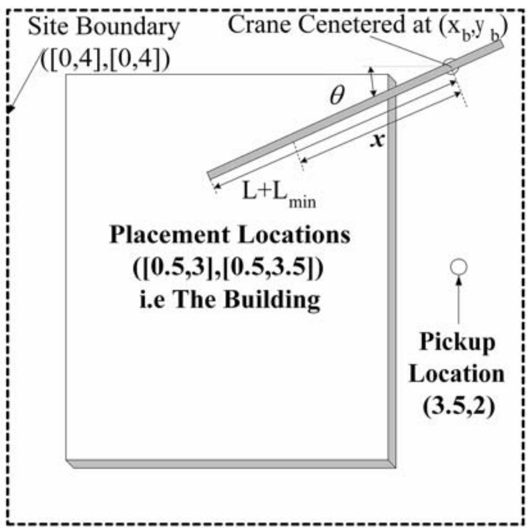


Fig. 5 The Local Variability function uses a dataset obtained from Murphy et al. [36] to define a parabolic blend as shown in [4]



Site Boundary ([0,4],[0,4])
Crane Cenetered at (x_b,y_b)
θ
x
L+L_{min}
Placement Locations ([0.5,3],[0.5,3.5]) i.e The Building
Pickup Location (3.5,2)

Fig. 6 Locating a crane at a construction site is a tradeoff between the size of the crane required, the range of motion required for the crane to reach all locations of interest at the site (which is related to the efficiency with which the crane can accomplish its work), and the need to avoid interfering with construction due to the crane location. The mathematics of the objective function combining these conflicting needs is defined in [4]

‘‘pick-and-place’’ robotics problems such as those common in 2D assembly operations (wafer handling, circuit board assembly, etc). This problem includes six performance indices, defined by three design variables. Figure 6 defines the problem design variables.

4 Performance comparisons

Performance of the multicriteria sequential sampling algorithm HyPerSample is demonstrated with the resulting correlation and RMS error metrics for the trial problems studied. For the five test problems presented in detail, the number of data points collected and used in the construction of the metamodel, the number of extrema modeled in comparison with the actual number of extrema, and the position error and value error of the optimal location are also evaluated. The reason for convergence is indicated for each trial. The performance for all five test problems, with respect to the initial dataset size and the randomly generated datasets is summarized in Sect. 4.6.

4.1 JCN test problem

The Jacobian condition number (JCN) test problem displays nonlinear behavior, with three distinct local minima, two of which are defined by sharp cusps. For this problem, intermediate results have been included, demonstrating the progress of the algorithm in developing a suitable metamodel. The search begins with an initial dataset of two points, defining a linear model, obtained from a 2^k search and shown in Fig. 7.

Subsequent iterations add additional data to the dataset, which is incorporated into the model, leading to an evolution of the metamodel representation. After eight iterations, 11 points have been defined, and the model has evolved to that shown in Fig. 8.

Further iterations continue to add data to the model, but result in small changes to the HyPerModel since the model criterion is becoming increasingly important in the calculation of the objective function. The majority of points added are near the initially defined optimum located near $x = 0.4$, as shown in Fig. 9.

At this point the cooling schedule resets, having reached its reset limit. Sampling reverts to a global search, adding points into regions that were previously sparsely sampled. Some of these points result in model refinements and others confirm that the model already represents the underlying function and are added to the unused dataset. After 23 iterations, the resulting model is shown in Fig. 10.

The model does not yet have sufficient data points in the unused dataset to achieve convergence, therefore, sampling continues. Eventually 39 points of the 50 points budgeted are collected, with 19 points actually used to define the HyPerModel. When a larger dataset of 11 points is initially provided, comparable results are obtained with a total of 41 points collected and 25 used

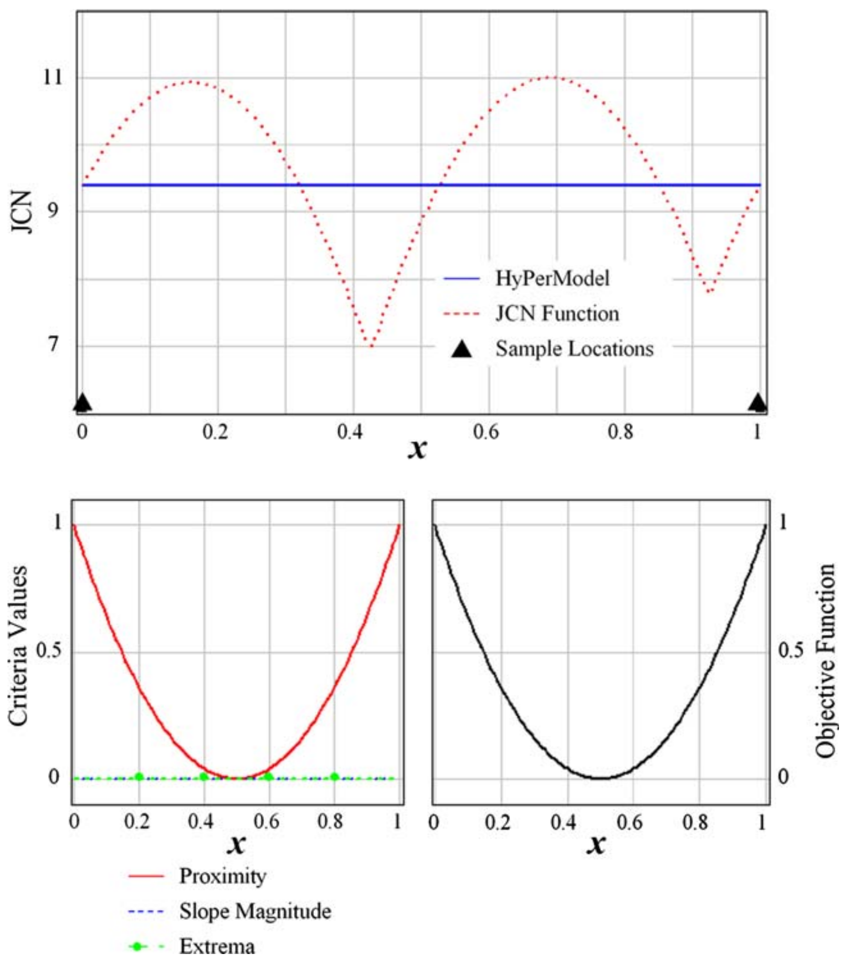


Fig. 7 The initial model is a horizontal straight line with zero slope. As this is the first iteration, the cooling schedule defines the objective function as the proximity criterion only. The model correlation to the actual function is initially 0%

to define the HyPerModel. The resulting models are shown in Fig. 11.

The larger initial dataset does have a slightly higher correlation than the 2^k approach (99.8 vs. 99.4%) and a smaller RMS error (10.3 vs. 15.5%). Both HyPerModels converge by achieving the correlation target while maintaining an RMS error less than the RMS error threshold as calculated with the unused dataset.

4.2 Local variability test problem

Like the JCN test problem, the nonlinear nature of the Local Variability test function presents a challenge to sequential sampling algorithms. The high variability of the function for $0.1 < x < 0.4$ requires additional samples in this particular region. The existence of the global optimum at $x = 0$ means that this point is sampled immediately for a 2^k search, but also tends to draw the model towards small values. Similarly, since the points in the region of greatest variability will generally be included in the next metamodel iteration, the proximity criterion will tend to favor other regions in the HyPerModel. Only the slope criterion will be drawn to this region. The resulting HyPerModels are shown in Fig. 12.

While both models achieve correlations of better than 90%, adequate for many applications, both miss the local optimum near $x = 0.15$. As was true for the JCN test problem, the larger initial sample did result in a slightly higher model correlation (93.3 vs. 90.9%) as well as a lower maximum RMS error (38.0 vs. 48.0%) in comparison to the 2^k initial dataset. The larger initial dataset also converges after adding 15 consecutive points without triggering a model update, the cooling schedule limit. Thus, convergence is due to reaching the cooling schedule limit, whereas the previous model converged due to model correlation.

4.3 Six-hump camel back problem

The six-hump camel back problem, defined by Eq. 28, is a 2D nonlinear optimization trial problem to which sequential sampling is often applied. Both initial sampling sizes produced comparable correlations with respect to the actual function (99+%). The larger initial sample of 25 points did produce a model that exhibited

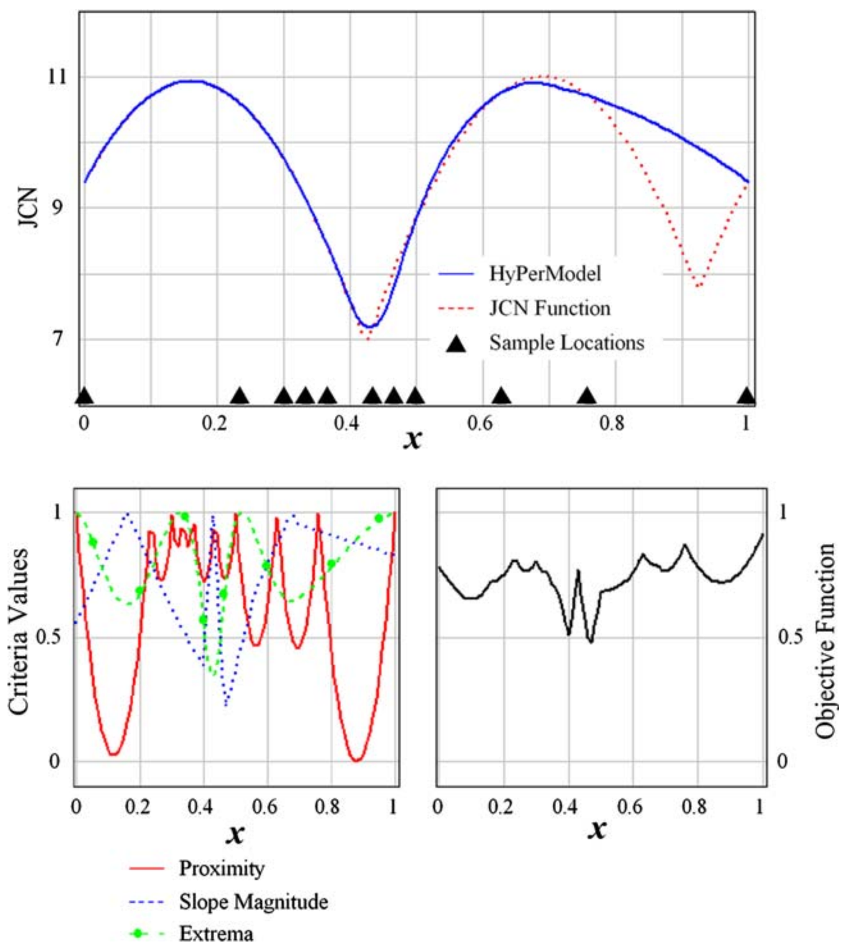


Fig. 8 After eight iterations the model is a much better representation of the underlying function. The objective function is defined by a combination of the three criteria, with the slope magnitude criterion exhibiting significant influence on the shape of the objective function. The model correlation to the actual function is now 89%

a smaller maximum RMS error (4.3 vs. 16.7%), but also required more samples to achieve convergence (331 vs. 236) and more points were involved in fitting the HyPerModel (157 vs. 79). Both HyPerModels converged after achieving model correlation. The resulting models are shown in Fig. 13.

4.4 Hansen problem

The Hansen function, Eq. 29, like the local variability problem, is a highly multimodal function defining 96 local extrema. Only about half of these extrema were modeled with either the 2^k or the 5^k initial datasets (see Fig. 14). Both approaches converged by exhausting the available experimental point budget, making a strong case for further sampling. Consequently, neither HyPerModel demonstrates a particularly high correlation coefficient and both also exhibit large maximum RMS errors.

4.5 Crane location problem

The crane location problem is also interesting due to its complexity. In order to model the objective function, the six performance indices were modeled within a single 3D-input/6D-output HyPerModel. By constructing a single HyPerModel, the costs of sampling the 6D outputs can be "shared" between dimensions, since features in one dimension may lead to the discovery of correlated features in other dimensions.

In order to deal with the impact of multiple dimensions on the slope magnitude and model criteria, the results for each output dimension are combined through a tensor product. With the individual performance index models constructed, they can be combined into a single objective function model. Figure 15 shows a slice of the objective function model for the length of reach, L, equal to zero.

The correlation between the 2^k initial dataset model and the actual function is 86.4%, while the 5^k initial dataset model achieves a correlation of 85.6%. Both approaches have significant RMS errors of 34.8 and 27.1%, respectively. Due to the high dimensionality of the problem, both HyPerModels required a significant number of data points to converge (1,806 and 4,259, respectively), and both use a significant number of points to define their HyPerModels (626 and 552,

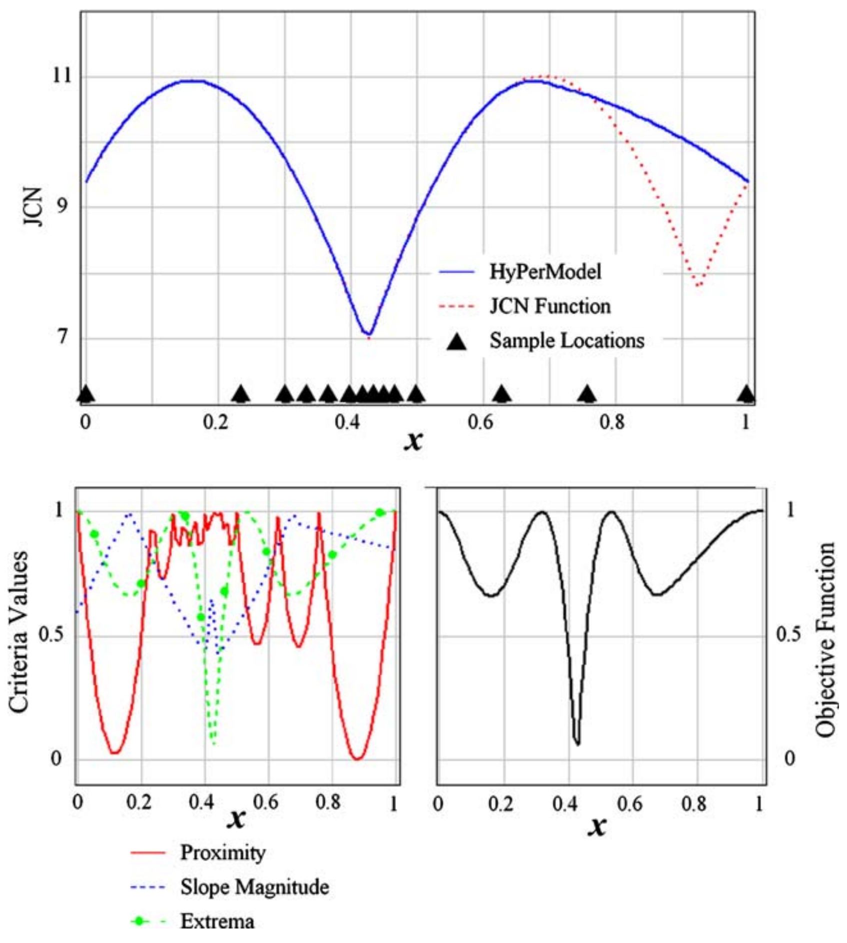


Fig. 9 After 15 iterations the model has refined its representation of the optimum near $x = 0.4$. The objective function is now defined by the model extrema criterion. The model correlation to the actual function is now 89%

respectively). Interestingly, the 2^k HyPerModel converges by achieving model correlation, while the 5^k HyPerModel exhausts the experimental sampling budget of 4096 (163) points. However, both approaches correctly represent 10 of the 11 extrema in the actual design space.

Both approaches generate optimal locations that are close matches to those of the actual model. Not surprisingly, the crane selected is only as long as necessary to reach all of the placement points (i.e. $L = 0$) and it is located at approximately either (4, 0) or (0, 4). This minimizes the angle to be transversed, while maintaining a similar distance to most of the placement points.

4.6 Overall performance

Overall, the HyPerSample generated HyPerModels produce more accurate representations of the actual functions than HyPerModels using randomly generated datasets. Both the global correlation coefficient and the local RMS error demonstrate that HyPerSample generated models are superior for the five functions studied. The global correlation coefficient results are summarized in Table 2 and the local RMS error results are summarized in Table 3.

In each case shown in Table 2, the HyPerModels built using HyPerSample exhibit higher global correlation coefficients with respect to the actual function than the corresponding HyPerModels generated using random datasets. For simple functions with few local optima, the differences are relatively minor. However, for highly multivariate functions, HyPerSample produces dramatically superior models.

Similarly, in each case shown in Table 3, the HyPerModels built using HyPerSample exhibit lower local RMS errors with respect to the actual function than the corresponding HyPerModels generated using random datasets. Again, for simple functions, the differences are not as pronounced as they are for the more complex multivariate examples. Clearly, HyPerSample produces HyPerModels that more accurately represent the actual function than randomly generated HyPerModels.

Table 4 compares the performance of HyPerSample when using a small initial dataset versus a large initial

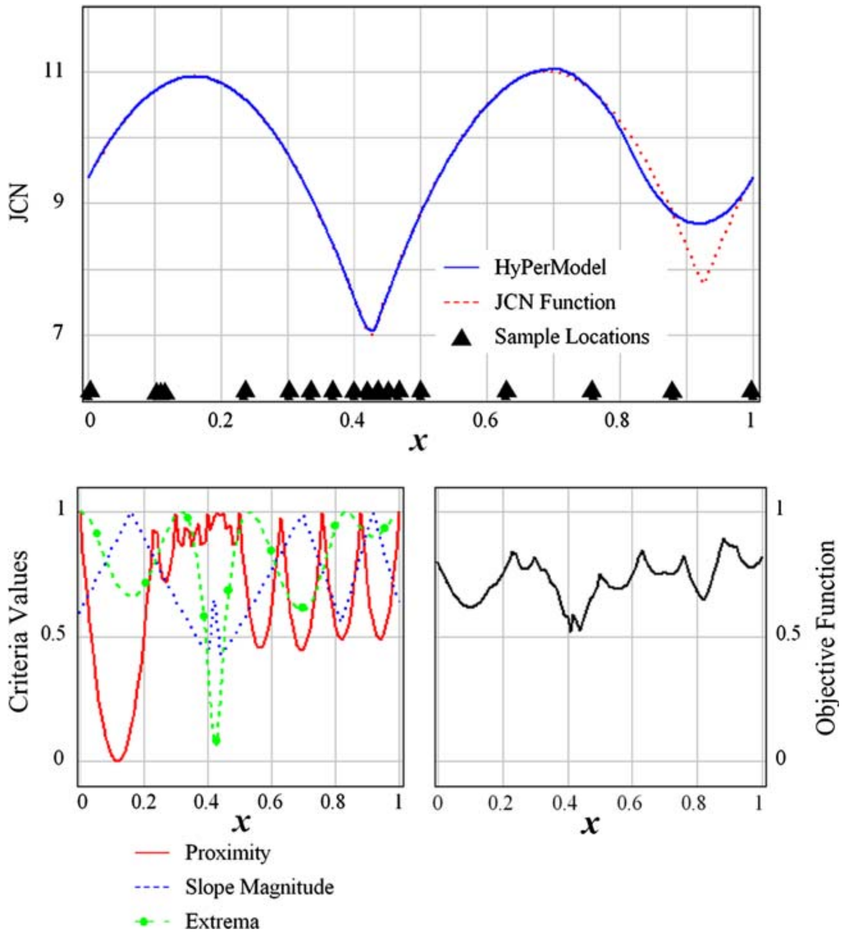


Fig. 10 After 23 iterations points have been added in the voids between $x = [0, 0.25]$ and $x = [0.75, 1]$ leading to a refinement of the model for larger values of x. The model correlation to the actual function is now 99%

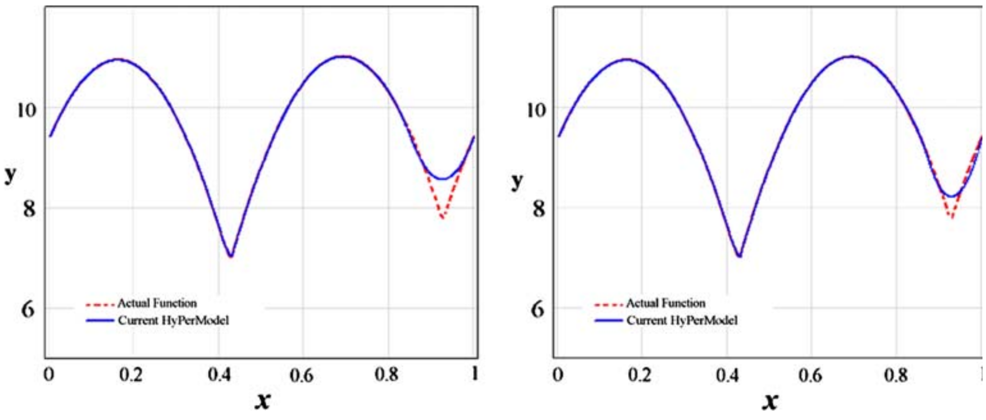


Fig. 11 The final models for a 2^k initial dataset (*left*) and an 11 point dataset (*right*) show similar results with correlations of greater than 99% to the actual function

dataset to define the initial HyPerModel. In general, the advantages of using a larger initial dataset are negligible. Both approaches yield comparable average correlations and maximum RMS errors. The resulting models capture a similar percentage of extrema and represent those extrema with similar errors. The major difference among all examples is the average percentage of the experimental budget used, where small initial datasets typically required half the data as the larger initial datasets required. While not always the case, the Hansen function demonstrates that even a 2^k search may exhaust the experimental budget. This suggests that the 2^k initial dataset, supplemented with a multi-criterion sequential sampling technique, and terminated with suitable convergence criteria, can be highly effective.

In these ten trials, six models terminated with sufficient evidence to support an argument that the

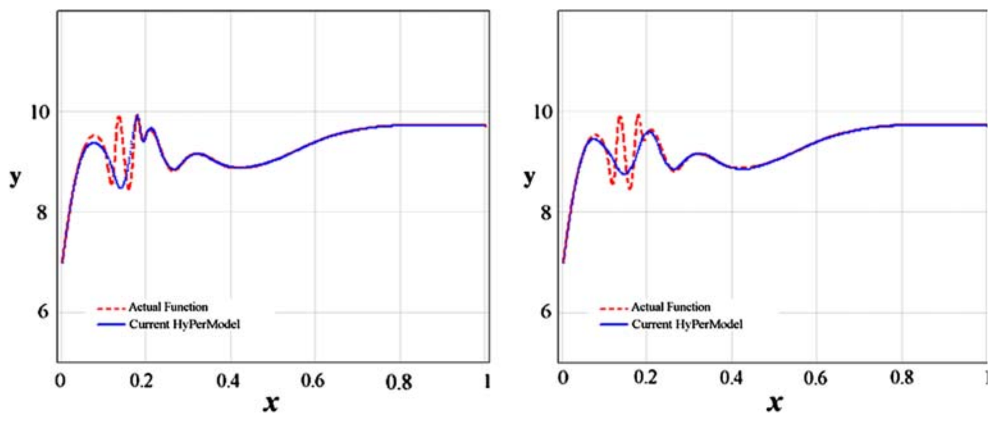


Fig. 12 The final models for a 2^k initial dataset (*left*) and an 11 point dataset (*right*) show similar results. The 2^k dataset identifies 11 of the 13 local extrema in the problem, while the larger, 11 point, initial dataset models only 9 local extrema since half of its initial sample set is in the relatively constant region for $x > 0.5$

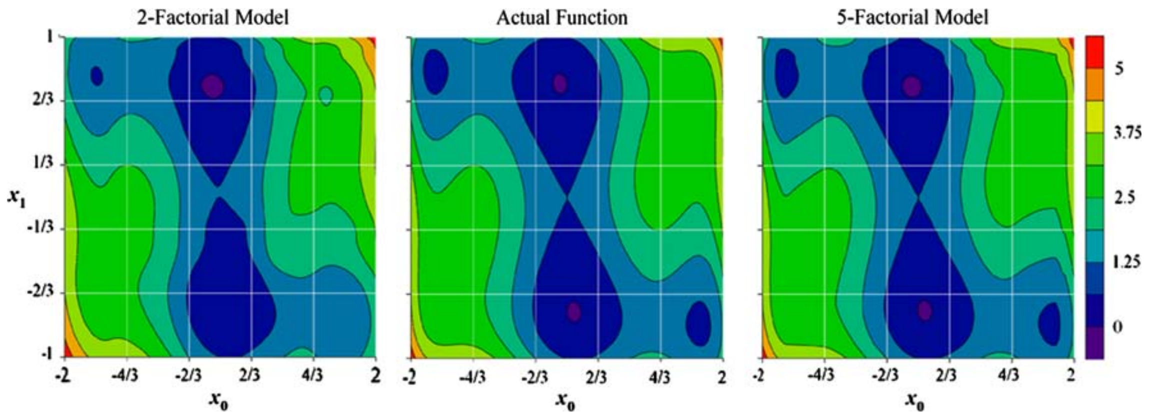


Fig. 13 The final models for a 2^k initial dataset (*left*) and 5^k initial dataset (*right*) show similar results to the actual function (*middle*)

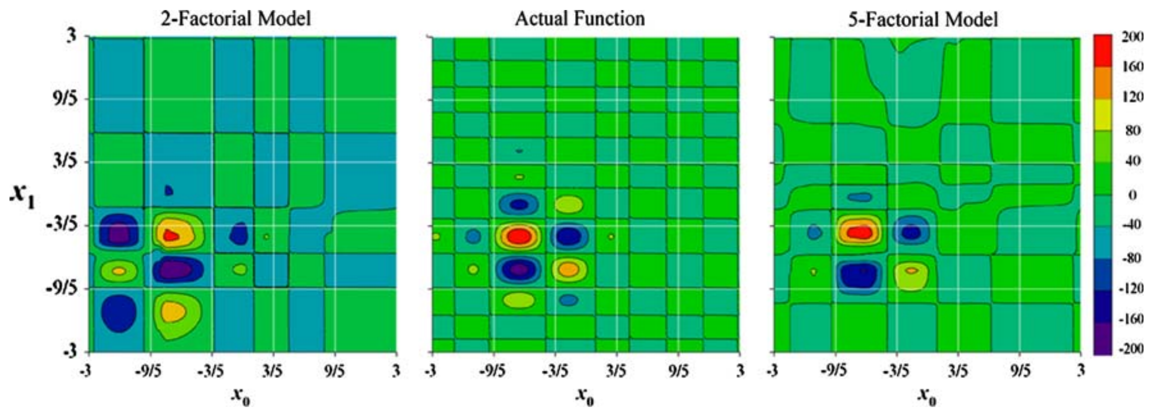


Fig. 14 The final models for a 2^k initial dataset (*left*) and 5^k initial dataset (*right*) show similar results to the actual function (*middle*)

experiment is complete. Three other trials suggested that additional experimentation would be justified since there is insufficient data to validate the resulting metamodel. Such decisions though are left to the experimenter, but could be automated in the future. HyPerSample currently performs no automatic decision-making concerning the adequacy of the experiment. Only one trial was inconclusive, the local variability function with a large initial dataset, and would require a bootstrapping restart method in addition to the multicriteria sequential sampling algorithm.

The performance of the trial problems is characteristic of the performance of HyPerSample when ap-

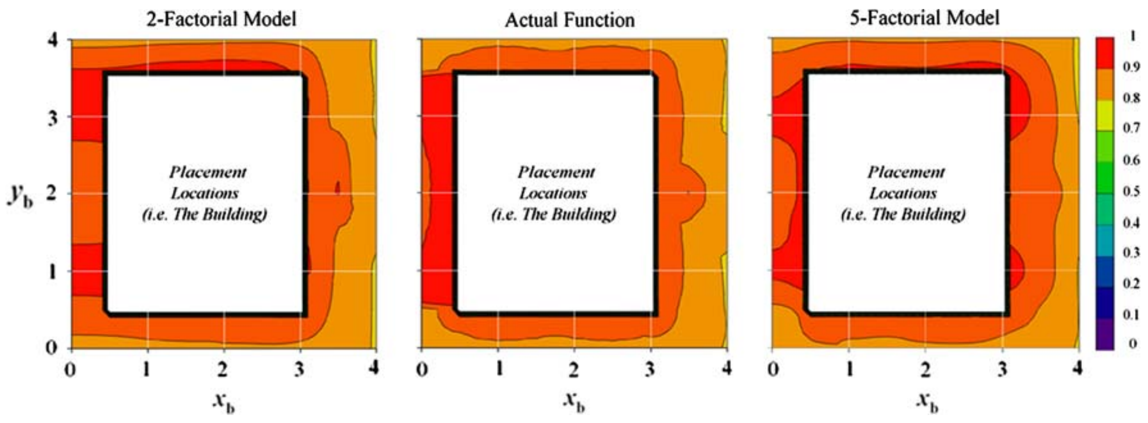


Fig. 15 The final models for a 2^k initial dataset (*left*) and 5^k initial dataset (*right*) show similar results to the actual function (*middle*), shown at $L = 0$

Table 2 Global Correlation Coefficient with respect to the actual function. Higher percentages indicate better representation of the actual function by the HyPerModel

Function (section)	Data points	HyPerSample correlation (%)	Random dataset correlation (%)
Small initial dataset			
JCN (4.1)	19	99.4	94.3 ± 5.8
Local variability (4.2)	17	90.9	52.1 ± 15.0
Six-hump camel back (4.3)	79	99.2	91.6 ± 2.2
Hansen (4.4)	128	53.7	21.4 ± 6.0
Crane location (4.5)	626	86.4	58.3 ± 3.4
Average		85.9	63.6 ± 28.5
Large initial dataset			
JCN (4.1)	25	99.8	99.0 ± 0.7
Local variability (4.2)	14	93.3	60.9 ± 14.1
Six-hump camel back (4.3)	157	99.9	94.1 ± 1.4
Hansen (4.4)	189	69.5	42.7 ± 17.9
Crane location (4.5)	552	85.6	60.4 ± 0.4
Average		89.6	71.4 ± 23.9

Table 3 Local RMS Error with respect to the actual function. Lower percentages indicate better representation of the actual function by the HyPerModel

Function (section)	Data points	HyPerSample RMS error (%)	Random dataset RMS error (%)
Small initial dataset			
JCN (4.1)	19	15.5	35.5 ± 7.9
Local variability (4.2)	17	48.0	78.3 ± 59.4
Six-hump camel back (4.3)	79	16.7	35.7 ± 7.6
Hansen (4.4)	128	23.5	51.0 ± 21.8
Crane location (4.5)	626	34.8	31.8 ± 17.4
Average		27.7	46.5 ± 32.4
Large initial dataset			
JCN (4.1)	25	10.3	16.9 ± 7.2
Local variability (4.2)	14	38.0	131.0 ± 98.9
Six-hump camel back (4.3)	157	4.3	27.0 ± 7.4
Hansen (4.4)	189	33.5	42.0 ± 22.6
Crane location (4.5)	552	27.1	38.9 ± 11.9
Average		22.6	51.1 ± 59.2

plied to other problems of engineering interest. Details concerning the additional experimentation with HyPerSample generated HyPerModels can be found in Turner [4].

5 Summary, conclusions and future work

Effectively sampling an unknown design space to determine the underlying functional behavior is an

Table 4 Detailed trial problem performance summary

Initial data	Trial	Correlation % (% RMS error)		Points sampled (% of point budget)	Points used (% of points sampled)	Optimum position error (% full scale)	Optimum value error (% full scale)	% of extrema modeled	Termination cause
		Unused dataset	Actual func. data						
Small initial dataset	JCN	100 (0.46)	99.4 (15.5)	78	49	0.3	0.2	100	Correlation
	LV	100 (0.46)	90.9 (48.0)	74	46	0.0	0.0	85	Correlation
	6HCB	99.9 (2.36)	99.2 (16.7)	59	33	1.1	0.6	75	Correlation
	Hansen	95.3 (17.2)	53.7 (23.5)	114	28	2.2	4.5	50	Point limit
	Crane	100 (0.44)	86.4 (34.8)	44	35	4.5	2.6	91	Correlation
	Average	99.1 (4.18)	85.9 (27.7)	51	34	1.6	1.6	80	
Large initial dataset	JCN	100 (0.24)	99.8 (10.3)	82	61	0.2	0.9	100	Correlation
	LV	100 (0.41)	93.3 (38.0)	68	41	0.0	0.0	69	CS limit
	6HCB	99.98 (1.29)	99.9 (4.3)	83	47	1.0	0.4	75	Correlation
	Hansen	93.0 (16.6)	69.5 (33.5)	110	43	0.9	5.6	52	Point limit
	Crane	94.2 (37.1)	85.6 (27.1)	104	13	7.5	0.6	91	Point limit
	Average	97.4 (11.1)	89.6 (22.6)	102	18	1.9	1.5	61	
Overall average		98.2 (7.65)	87.8 (25.2)	79	39	1.8	1.5	71	

important challenge in engineering design applications. Effective functional representations, in the form of metamodels, can be used to discover variable relationships, identify the significance of trends, and to locate optimal variable combinations. However, in order to build accurate and useful metamodels, an adaptive sequential sampling technique is highly desirable to facilitate the efficient collection of data, particularly when data collection is costly or the design space to be sampled is vast.

A multicriteria sequential sampling approach using a cooling schedule to translate between criteria is an effective method of adaptively collecting data about unknown functions. When coupled with a set of convergence criteria, it becomes possible to determine when an experiment has collected sufficient data to determine the adequacy of the metamodel, or if further experimentation may be warranted. The effectiveness of the HyPerSample algorithm in generating accurate HyPerModels was demonstrated for five trial problems of engineering interest.

HyPerSample uses an initial dataset, obtained from a factorial search, a latin hypercube approach or a random search, to produce an initial dataset to define an initial HyPerModel. This initial HyPerModel is used to define the criteria by which HyPerSample selects future sampling locations. HyPerSample outperforms randomly generated datasets and is effective with a relatively small initial dataset. Furthermore, HyPerSample provides a mechanism by which the validity of the metamodel can be judged and by which conclusions about the adequacy of the experimental budget can be made.

While the criteria used in HyPerSample are specifically formulated for HyPerModels, comparable criteria can be defined for other types of metamodels. The chief challenge in doing so will be to develop normalization techniques so that the criteria magnitudes do not play a role in the cooling schedule formulation of the objective function. The use of a cooling schedule to blend and transition between criteria can be applied to any type of metamodel, as can the idea of simultaneously generating a validation as well as metamodel fitting datasets in order to establish convergence criteria for an adaptive sampling algorithm.

Of concern in the current HyPerSample implementation is the solution to the optimization problem that determines the next sampling location. HyPerSample currently employs a multi-start approach, using the HyPerModel control points as starting locations, which has yielded satisfactory results. However, as the model size increases and the number of control points climbs, the number of optimizations required also increases, and the algorithm slows. Preliminary results suggest that allowing the user to specify a limit on the number of optimizations, and determining the starting points with a random multi-start approach, may be similarly effective. Other approaches, such as using stochastic or heuristic optimization approaches with a time limit for optimization remain to be explored. The goal of each individual sequential sampling optimization problem is not necessarily the globally optimal sampling location, but rather a good locally optimal solution that yields valuable data for calculating the metamodel and its validity.

In some cases, the algorithm fails to converge and cannot proceed on its own. In these cases, we speculate that a change in strategy is warranted. The development of additional criteria or the application of different cooling schedules may yield techniques to allow the algorithm to bootstrap itself out of such conditions. In addition, a better understanding of the tradeoffs resulting from the selection of different convergence criteria values should be undertaken.

Future work also will focus on exploring alternative formulations in the HyPerModel and HyPerSample relations and parameters. This research should improve the efficiency of HyPerSample and further optimize the algorithm. In addition, improved handling of convergence limits, improved optimization of the criteria objective function and a boot strap recovery from the inconclusive convergence outcome are highly desirable.

Acknowledgments The assistance and support of Troy Harden, Chris James, and Kane Fisher of Los Alamos National Laboratory and the students of The University of Texas at Austin Manufacturing and Design Laboratory to complete this work is greatly appreciated.

References

1. Hamada M, Wu C (2000) Experiments: planning, analysis, and parameter design optimization. Wiley, New York
2. Montgomery D (1997) Design and analysis of experiments, 4th edn. Wiley, New York
3. Myers R, Montgomery D (2002) Response surface methodology: process and product optimization using designed experiments, 2nd edn. Wiley, New York
4. Turner C (2005) HyPerModels: hyperdimensional performance models for engineering design. Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin, Austin
5. Rogers D, Adams J (1990) Mathematical elements of computer graphics, 2nd edn. McGraw-Hill, New York
6. Piegl L, Tiller W (1997) The NURBS book, 2nd edn. Springer, Berlin Heidelberg New York
7. Cohen E, Riesenfeld R, Elber G (2001) Geometric modeling with splines: an introduction. A.K. Peters, Natick
8. Turner C, Campbell M, Crawford R (2004) Metamodel defined embedded multidimensional sequential sampling criteria. In: Proceedings 2004 ASME IDETC/CIE, Salt Lake City, Utah, CIE-57722
9. Turner C, Crawford R (2005) Adapting non-uniform rational B-spline fitting techniques to metamodeling. In: Proceedings 2005 ASME IDETC/CIE, Long Beach, CIE-85544
10. Gopi M, Manohar S (1997) A unified architecture for the computation of B-spline curves and surfaces. IEEE Trans Parall Dist Sys 8:1275–87
11. Farhang-Mehr A, Azarm S (2005) Bayesian meta-modelling of engineering design simulations: a sequential approach with adaptation to irregularities in the response behavior. Int J Numer Meth Eng 62:2104–26
12. Osio I, Amon C (1996) An engineering design methodology with multistage Bayesian Surrogates and optimal sampling. Res Eng Design 8:4, 189–206
13. Martin J, Simpson T (2002) Use of adaptive metamodeling for design optimization. In: 9th AIAA/ISSMO symposium on Multidis. Anal. and Opt., Atlanta, Georgia, AIAA-2002–5631
14. Wang G, Simpson T (2004) Fuzzy clustering based hierarchical metamodeling for design space reduction and optimization. Eng Optim 35:313–35
15. Sasena M, Parkinson M, Goovaerts P, Papalambros P (2002) Adaptive experimental design applied to an ergonomics testing procedure. In: Proceedings 2002 ASME IDETC/CIE, Montreal, DAC-34091
16. Sasena M (2002) Flexibility and efficiency enhancements for constrained global design optimization with kriging approximations. Ph.D. Dissertation, University of Michigan, Ann Arbor, MI
17. Pérez VM, Renaud J, Watson L (2002) Adaptive experimental design for construction of response surface approximations. AIAA J 40:2495–2503
18. Gutmann H (2001) A radial basis function method for global optimization. J Global Optim 19:201–27
19. Jin R, Chen W, Sudjianto A (2002) On sequential sampling for global metamodeling in engineering design. In: Proceedings 2002 ASME IDETC/CIE, Montreal, DAC-34092
20. Kushner H (1964) A new method of locating the maximum of an arbitrary multi-peak curve in the presence of noise. J Basic Eng 86:97–106
21. Locatelli M (1997) Bayesian algorithms for one-dimensional global optimization. J Global Optim 10:57–76
22. Mockus J (1989) Bayesian approach to global optimization. Kluwer, New York
23. Schonlau M (1997) Computer experiments and global optimization. Ph.D. Dissertation, University of Waterloo, Waterloo
24. Cox D, John S (1997) SDO: a statistical method for global optimization. In: Multidisciplinary Design Optimization: State of the Art, SIAM, Philadelphia, pp 315–29
25. Farhang-Mehr A, Azarm S (2002) A sequential information-theoretic approach to design of computer experiments. In: 9th AIAA/ISSMO symposium on Multidis. Anal. and Opt., Atlanta, AIAA, AIAA-2002–5571
26. Watson A, Barnes R (1995) Infill sampling criteria to locate extremes. Math Geol 27:589–608
27. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–679
28. Cerny V (1985) A thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45:41–51
29. Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. J Comp Surv 35:268–308
30. Lundy M, Mees A (1986) Convergence of an annealing algorithm. J Math Prog 34:111–124
31. Osman I (1993) Metastrategy simulated annealing and tabu search algorithms for vehicle routing problems. Ann Optim Res 41:421–451
32. Friedman P (1994) The adaptive multidimensional data acquisition in physical experiments. Signal Process 38: 253–258
33. Jones D, Schonlau M, Welch W (1998) Efficient global optimization of expensive black-box functions. J. Global Opt. 13: 455–492

34. Jones D (2001) A taxonomy of global optimization methods based on response surfaces. J Global Optim 21: 345–383
35. Turner C (2002) Metamodels for Planar 3R workspace optimization. In: Proceedings 2002 ASME IDETC/CIE, Montreal, CIE-34500
36. Murphy T, Lin Y, Tsui K, Allen J, Chen V, Mistree F (2004) Robust engineering design. In: Proceedings 2004 NSF Design, Service and Manuf. Grantees and Res. conference, Dallas
37. Adorio E (2005) MVF–Multivariate test functions in C for unconstrained global optimization. Updated January 14, 2005, available at http://www.geocities.com/eadorio/mvf.pdf, Last Accessed February 6, 2005