Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational and Applied Mathematics 217 (2008) 110–122
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

Finding effective points by surrogate models with overcomplete bases

Weichung Wang[a], Ray-Bing Chen[b,*]

[a]*Department of Mathematics, National Taiwan University, Taipei 10617, Taiwan, ROC*
[b]*Institute of Statistics, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC*

Received 14 September 2004

Abstract

We develop algorithms to find the so-called effective points (EPs) $\mathbf{x} \in \mathscr{X} \subset \mathbb{R}^n$ such that the corresponding responses $f(\mathbf{x}) \in \mathbb{R}$ belong to a specific region of interest (ROI). Examples of an ROI include extreme values, bounded intervals, and positivity. We are especially interested in the problem defined by the following characteristics: (i) the definition of $f(\mathbf{x})$ is either complicated or implicit, (ii) the response surface $f(\mathscr{X})$ does not fit simple patterns, and (iii) computational costs of function evaluations is high. To solve this problem, we iteratively approximate the true yet unknown response surface with simplified surrogate models and then use the surrogate models to predict the possible EPs. Unlike interpolation schemes, the surrogate models are formed by linear combinations of a set of overcomplete bases and they are not obliged to fit the known response value. A numerical example that involves finding positive Lyapunov exponents of a dynamical system shows that the algorithm is efficient and practical.

Keywords: Effective point; Region of interest; Response surface; Overcomplete basis representation; Surrogate models

1. Introduction

We consider the problem of finding *effective points* (EPs) $\mathbf{x} \in \mathscr{X} \subseteq \mathbb{R}^n$, such that the corresponding function values $f(\mathbf{x}) \in \mathbb{R}$ belong to a *region of interest* (ROI). The EPs search problem has many potential applications. For example, an optimization problem, min $f(\mathbf{x})$, subject to $\mathbf{x} \in \mathscr{X}$, is equivalent to the problem in which the ROI represents minimal values. Another example, shown in Section 3.1, involves finding suitable parameter combinations of a dynamical system such that the corresponding Lyapunov exponents are positive. In this example, a function value $f(\mathbf{x})$ represents the Lyapunov exponent corresponding to a certain parameter set $\mathbf{x}$ and the ROI is the set of positive $f(\mathbf{x})$'s. In addition, computer program performance tune-up can also be formulated in the form of the target problem. In this case, the domain $\mathscr{X}$ is the range of the program parameters, whereas the function value $f(\mathbf{x})$ could be iteration number, convergence rate, or CPU time.

In this article, we further assume the following problem characteristics. First, the computational or experimental cost for obtaining such values is very expensive. Secondly, the function $f(\mathbf{x})$ is either very complicated or defined implicitly. Traditional methods based on manipulations (e.g., derivatives) of an explicitly defined $f(\mathbf{x})$ are thus inefficient or

* Corresponding author. Tel.: +886 7 591 9523; fax: +886 7 591 9360.
E-mail addresses: wwang@math.ntu.edu.tw (W. Wang), rbchen@nuk.edu.tw (R.-B. Chen).

0377-0427/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2006.12.033

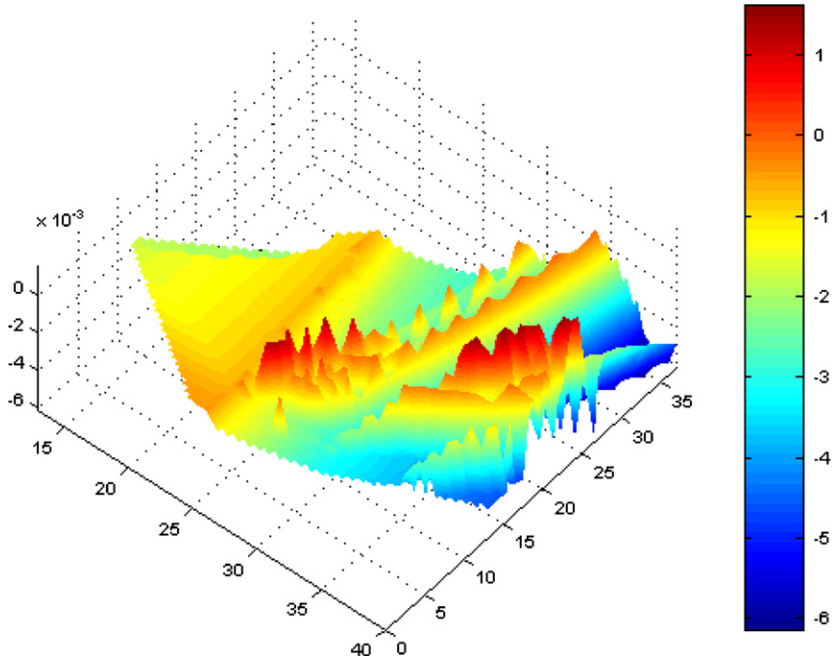

Fig. 1. An example illustrating an oscillatory response surface.

inapplicable. Thirdly, the response surface formed by the function values over $\mathscr{X}$ is complicated and is not easily modeled by simple functions. Fourthly, the feasible domain is a certain grid $\mathscr{G} \subseteq \mathscr{X}$. The last assumption is needed in some applications like the one to be discussed in Section 3.1.

Although many applications can be formulated as EP search problems, few systematical methods exist. Besides the simple grid search method, response surface method (RSM) is another popular option for optimization problems. RSM has been proposed and studied theoretically in [1,9,12] and other papers. RSM has also been used successfully in practical applications such as the study of clinical enzyme assays [17], optimization of the shape of a supersonic turbine [13], and an evaluation of experimental conditions for bioconversion [3]. In RSM, the response variable y is modeled as

$$y = f(\mathbf{x}) + \varepsilon,$$

where ε is white noise. Even though the model of y is a noise model, RSM is still concerned with the functional relationship between y and $\mathbf{x}$, i.e., $E(y) = f(\mathbf{x})$, and usually assumes that the noise is sufficiently small that it can be ignored. A general RSM, therefore, works as follows. The first step in RSM is to find a suitable approximation for the true, yet unknown, response surface defined by $f(\mathbf{x})$. This is typically accomplished with a first-order polynomial model of $\mathbf{x}$. The path of improvement toward the general vicinity of the target is then determined. After the RSM appears to reach the neighborhood of the target, a more complicated model, e.g., the second-order polynomial model, is used to obtain the approximate solution.

An important key point of RSM is that, under the smoothing assumption of the response function $f(\mathbf{x})$, the local response surface can be approximated well by the first- and the second-order polynomial models with respect to $\mathbf{x}$. However, in many situations, the surface oscillate violently. It is therefore difficult to find suitable approximations of such complicated surfaces using simple representations over $\mathbf{x}$. In such situations, traditional RSM may not perform accurately and efficiently. Fig. 1 illustrates an example of an oscillatory surface.

In this article, we propose an effective points search algorithm (EPSA) to solve this problem. The algorithm chooses a subset of the grid points from $\mathscr{G}$ as the experimental points and evaluates the function values of these experimental points. The function values are then used to form a surrogate surface represented by a set of overcomplete bases. Note that the surrogate surface does not necessarily fit the known response values. This surrogate surface is then used to predict possible EPs. These EP candidates are verified by computing their corresponding function values to see if the

function values belong to the ROI. If not, these EP candidates are added to the experimental point set. The algorithm iterates the process until one of the stopping criteria is satisfied.

The general framework of the new algorithm is discussed and analyzed in Section 2. An application example and the corresponding numerical results of EPSA and RSM are demonstrated and discussed in Section 3. Finally, we present a brief conclusion in Section 4.

2. The effective points search algorithm

In this section, we propose an EPSA to find EPs for a certain ROI. We first discuss the main ingredients of the algorithm in Section 2.1 and then discuss a key idea of using overcomplete bases to form the surrogate surfaces in Section 2.2. The complete algorithm is summarized and analyzed in Section 2.3.

2.1. Main components of the algorithm

In the following discussions, we introduce the main components of the algorithm.

- *Defining the grid*: Let $\mathscr{G}$ denote a n-dimensional grid containing p_i $(i = 1, \ldots, n)$ grid points along each of the dimensions of $\mathscr{X}$. In total there are $N = p_1 p_2 \ldots p_n$ grid points. For convenience, we concatenate the grid points to form a long vector of length N. Taking a two-dimensional grid as an example, by labeling the grid points as $\mathscr{G}_{i_1,i_2}$ for $i_1 = 1, \ldots, p_1$ and $i_2 = 1, \ldots, p_2$, we can rearrange the points to form a $p_1 p_2$-by-1 vector $[\mathscr{G}_{1,1}, \mathscr{G}_{1,2}, \ldots, \mathscr{G}_{p_1,p_2}]^{\mathrm{T}}$.
- *Choosing initial experimental points*: The initial experimental points $\mathscr{P}_{\text{init}}$ are chosen uniformly over the grid $\mathscr{G}$. This is a reasonable choice since we have no information concerning the shape of the response surface. The uniform experimental points allow us to start the search without any bias.
- *Forming the reference response surfaces*: At each iteration, a portion of the grid points is chosen to act as the set of experimental points. We denote these experimental points as $\mathscr{P}_{\text{exp}}$ and $\mathscr{P}_{\text{exp}} \subseteq \mathscr{G}$. The response values, or $f(\mathbf{x})$'s are computed for all $\mathbf{x} \in \mathscr{P}_{\text{exp}}$. By setting $f(\mathbf{x})$ to 0 for all points in the set $\mathscr{G} \backslash \mathscr{P}_{\text{exp}}$, we define the *reference* response surface as

$$\mathscr{S}_{\mathscr{P}_{\text{exp}}} = \sum_{\mathbf{x} \in \mathscr{P}_{\text{exp}}} f(\mathbf{x}) e_{\mathbf{x}}.$$

  Here $\mathbf{e}_{\mathbf{x}}$ is a $N \times 1$ *unit vector* whose entries are all 0, except the entry associated with the point $\mathbf{x}$, which is 1. The set $\mathscr{P}_{\text{exp}}$ will be updated by adding new experimental points over the iterative process. At the beginning of the algorithm, we choose N_{init} points (usually $N_{\text{init}} \ll N$) from $\mathscr{G}$ to form the initial experimental point set.
- *Constructing the surrogate surfaces*: By using the information in the reference response surfaces, we want to construct surrogate surfaces that will be used to assist in identifying the possible effective points. Constructing the surrogate surfaces involves the following two main parts. Note that these ideas and their effectiveness will be further discussed in Section 2.2.
  (i) *Determining overcomplete bases*: Here we suggest approximating the true response surface by a linear combination of a pre-defined set of bases

$$\mathscr{D} = \{\phi_i \in \mathbb{R}^N | i = 1, 2, \ldots, M\}, \tag{1}$$

  where $M \geqslant N$, ϕ_i's are bases defined on the grid $\mathscr{G}$. Note that the notation $\mathscr{D}$ is used because a picture is decomposed into a set of bases called "dictionary" in image processing.
  (ii) *Forming the surrogate surfaces by the overcomplete bases*: The true response surface is then approximated by the surrogate surface

$$\tilde{\mathscr{S}}_{\mathscr{P}_{\text{exp}}} = \sum_{i=1,\ldots,M} c_i \phi_i, \tag{2}$$

  where c_i's are the corresponding weighting coefficients of the bases. If most of the weighting coefficients c_i are 0, we then have sparse representation of the response surface based on the bases $\mathscr{D}$.

- *Predicting next possible EPs*: We use the surrogate surface $\tilde{\mathscr{S}}_{\mathscr{P}_{\text{exp}}}$ to find possible effective points from the points $\mathscr{G}\backslash\mathscr{P}_{\text{exp}}$. In other words, we choose point $\tilde{\mathbf{x}}$'s that are located in $\mathscr{G}\backslash\mathscr{P}_{\text{exp}}$ and the corresponding surrogate surface values (denoted as $\tilde{\mathscr{S}}_{\mathscr{P}_{\text{exp}}}|_{\tilde{\mathbf{x}}}$) satisfy the ROI conditions. We collect these candidate points in the set $\mathscr{P}_{\text{new}}$.
  For example, in the case that ROI are maximum values, we can simply use a grid search (all the points in $\mathscr{G}$ are scanned and compared) to find a $\tilde{\mathbf{x}}$ resulting in the largest value of $\tilde{\mathscr{S}}_{\mathscr{G}}|_{\tilde{\mathbf{x}}}$, or

$$\tilde{\mathbf{x}} = \arg\max\{\tilde{\mathscr{S}}_{\mathscr{G}}|_{\mathbf{x}} \text{ for all } \mathbf{x} \in \mathscr{G}\}. \tag{3}$$

- *Verifying the possible EPs*: The possible EPs $\tilde{\mathbf{x}}$'s are then verified to see whether the points satisfy the defined ROI conditions. That is, we check whether the function values $f(\tilde{\mathbf{x}})$'s, $\tilde{\mathbf{x}} \in \mathscr{P}_{\text{new}}$ satisfy the ROI conditions of the true response surface. If not, these points are added to $\mathscr{P}_{\text{exp}}$.
- *Updating the set of experimental points*: At the end of the iteration, we update the experimental point set

$$\mathscr{P}_{\text{exp}} \equiv \mathscr{P}_{\text{exp}} \cup \mathscr{P}_{\text{new}},$$

  and the new reference response surface becomes

$$\mathscr{S}_{\mathscr{P}_{\text{exp}}} \equiv \mathscr{S}_{\mathscr{P}_{\text{exp}}} + \sum_{\mathbf{x}\in\mathscr{P}_{\text{new}}} f(\mathbf{x}) e_{\mathbf{x}}.$$

  One simple implementation is letting $\mathscr{P}_{\text{new}} = \tilde{\mathbf{x}}$, which is defined in (3).
- *Checking stopping criteria*: The algorithm terminates if there is no more candidate point (i.e., $\mathscr{P}_{\text{new}}$ is the empty set) or all points in $\mathscr{G}$ belong to $\mathscr{P}_{\text{exp}}$ (i.e., $\mathscr{P}_{\text{exp}} = \mathscr{G}$). Furthermore, it is also useful to terminate the algorithm if we have tested a certain number of experimental points. (i.e., $|\mathscr{P}_{\text{exp}}| = N_{\text{exp}}$) points. Here $N_{\text{exp}} \leqslant N$ is a pre-defined constant that roughly stands for the maximum computational cost we can offer.

Here we introduce the main ideas of the algorithm. Before discussing of how the surrogate surfaces are constructed, it is worth mentioning that our approach is different from the interpolation type algorithms because we do not force the surrogate surfaces to fit the known function values. This approach is reasonable since the surrogate surfaces are used to predict the next possible EPs rather than to fit certain data or to interpolate the unknown function values. A critical characteristic of this algorithm is that the "trend" of the true response surface can be detected by the surrogate surfaces.

2.2. Surrogate surfaces with overcomplete bases

Now we discuss how to choose the overcomplete bases and how to determine the surrogate surfaces by using the bases. These ideas are mainly inspired by recent advances in image (or signal) decomposition.

Many image (or signal) decomposition methods have been developed recently. These methods suitably choose bases from a pre-defined dictionary $\mathscr{D}$ to represent or approximate the target images by linear combinations of the bases chosen. A so-called dictionary is composed by a set of atoms (or waveforms) [11] and can be denoted as

$$\mathscr{D} = \{\phi_i | i = 1, 2, \ldots, M\},$$

where $\phi_i \in \mathbb{R}^N$ are the atoms of the dictionary defined on the same domain of the target image. An arbitrary image $\mathscr{I}$, which usually does not fit any special patterns, can then be decomposed as

$$\mathscr{I} = \sum_{\phi_i \in \Gamma} c_i \phi_i + \eta, \tag{4}$$

where c_i's are the corresponding weights, Γ is a set of atoms with nonzero weights, and η is the vector of the residual. The atoms in $\mathscr{D}$ may be pairwise orthogonal, linear independent, or even linear dependent. Furthermore, for an image containing N pixel, the dictionary may be complete ($M = N$), undercomplete ($M < N$), overcomplete ($M > N$), or continuous ($M = \infty$) [2]. Some well-developed image decomposition methods include the method of frames [6], matching pursuit [11,16], best orthogonal basis [4], and basis pursuit [2]. Popular dictionary sets include Dirac, Heaviside, Fourier, Wavelets, Gabor, Cosine Packets, Chirplets, and Warplets representation.

As the response surface would be sampled on grid points, which are similar to the pixels in images, the surface over the grid can be treated as an image. We can thus use the concepts of image decomposition to decompose the reference response surfaces for constructing surrogate surfaces. Note that the term image fits nicely in our discussion here as a problem is defined on $\mathbb{R}^2$ surface. For general response surfaces in $\mathbb{R}^n$, an image should be understood as a "$\mathbb{R}^n$ image". All the ideas are still applicable.

We first emphasize that a overcomplete dictionary is employed in our algorithm. That is, the number of atoms is larger than the dimension of the image, or the true response surface. The use of overcomplete image representations has been advocated recently because the wider range of generating elements allow more flexibility in image representation. An overcomplete dictionary is especially useful if the image is composed of a wide scope of patterns: some are smooth globally and some vary widely in a local spot. An overcomplete dictionary may also avoid some numerical difficulties while approximating an image by limited bases. An analogy of an overcomplete dictionary is described in [11] as follows. Although small vocabularies may express all ideas, full sentences would be needed for unavailable words. In contrast, large vocabularies can easily and compactly express all concepts, even those that contain subtle differences. For details, see [7] and the references therein.

Next, we discuss a specific overcomplete dictionary named "Gabor bases", which is used in our algorithm. In image (or signal) decomposition, Gabor bases are usually chosen when the overcomplete dictionary is considered. A general definition of the n-dimensional Gabor dictionary is given in [5]. First let $g^k \in L^2(\mathbb{R}^n)$ for $k = 1, \ldots, L$. Then the general d-dimensional Gabor dictionary is defined as

$$\{g^k_{\mathbf{m},\mathbf{n}} : \mathbf{m}, \mathbf{n} \in \mathbb{Z}^n, k = 1, \ldots, L\}, \tag{5}$$

where

$$g^k_{\mathbf{m},\mathbf{n}}(\mathbf{x}) = g^k(\mathbf{x} - B\mathbf{n}) \exp(2\pi \mathrm{i} A\mathbf{m} \cdot \mathbf{x}) \tag{6}$$

for some nondegenerate linear maps A, B on $\mathbb{R}^n$. Here we would suggest choosing g^k to be

$$g^k(\mathbf{x}) \propto \exp(-\tfrac{1}{2}\mathbf{x}^{\mathrm{T}} M_k \mathbf{x}), \tag{7}$$

where M_k is an $n \times n$ positive definite matrix. To provide a visual understanding of the Gabor bases, Fig. 2 illustrates two two-dimensional real-valued Gabor functions [15]

$$g(u, v) = \frac{1}{Z} \exp\left[-\frac{1}{2}(\sigma_u u^2 + \sigma_v v^2)\right] \cos\left[\frac{2\pi u}{\lambda} + \varphi\right], \tag{8}$$

$$u = u_0 + x_1 \cos\theta - x_2 \sin\theta, \tag{9}$$

$$v = v_0 + x_1 \sin\theta - x_2 \cos\theta, \tag{10}$$

where Z is the normalizing constant, (x_1, x_2) are the coordinates, u_0, v_0, σ_u, and σ_v are user chosen parameters of a two-dimensional Gaussian window satisfying relations $\sigma_v = \sqrt{2}\sigma_u$ and $\lambda = \sqrt{2\pi}\sigma_u$, λ and φ are parameters of a sinusoidal grating, and θ is the angle between the x_1-axis and the u-axis of the Gabor function. The figures clearly show that Gabor functions can be used to represent patterns both locally or globally (determined by σ_u) with various ridge directions (determined by θ).

Having defined the overcomplete dictionary, we need to determine the weighting coefficients c_i's to form the surrogate surfaces. Here we use matching pursuit [11,15] to decompose the surface

$$\mathscr{S}_{\mathscr{P}_{\exp}} = \sum_{j \in \Gamma} c_j \phi_j + \varepsilon_{\mathscr{P}_{\exp}}. \tag{11}$$

We set the overcomplete dictionary $\mathscr{D} = \{\phi_i \mid \|\phi_i\| = 1,\ i = 1, 2, \ldots, M\}$, the residual vector $\mathbf{r} = \mathscr{S}_{\mathscr{P}_{\exp}} - \sum_{i=1}^{M} \mathbf{c}_i \phi_i$ for a certain coefficient vector $\mathbf{c} \in \mathbb{R}^M$, and the inner product vector $\mathbf{p} \in \mathbb{R}^M$ with the kth entry $\mathbf{p}_k = \phi_k^{\mathrm{T}} \cdot \mathbf{r}$. Now we briefly describe the tth iteration of the matching pursuit scheme in Algorithm 1. See [15] for detail discussion of matching pursuit.

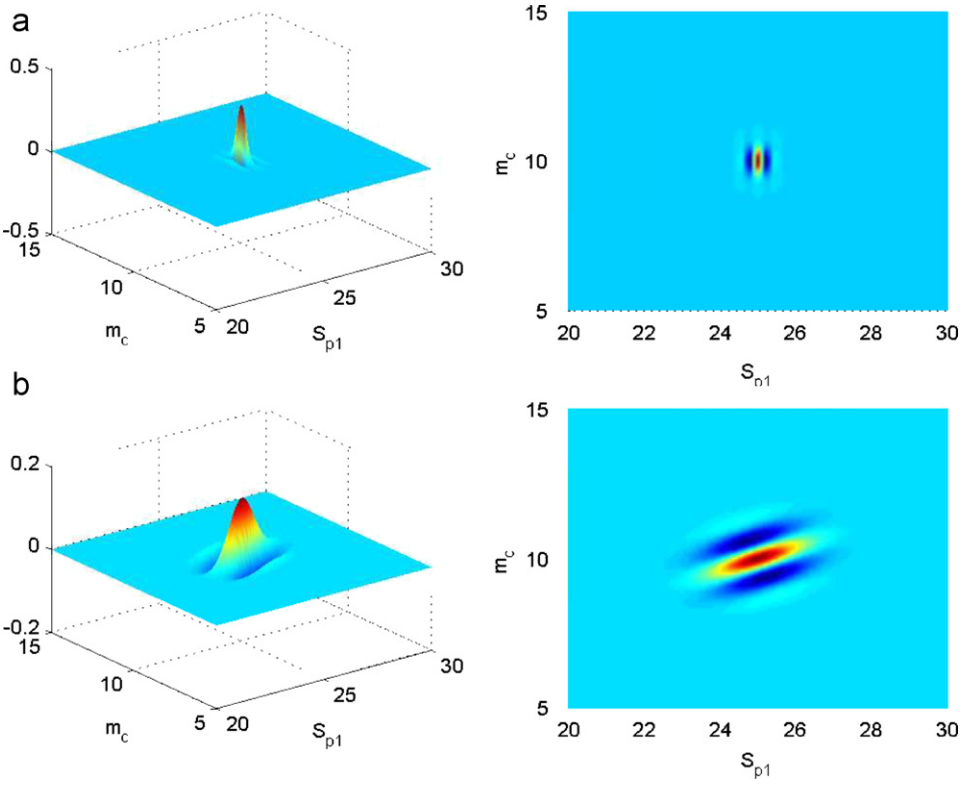

Fig. 2. The diagrams of Gabor functions for (a) $\sigma_u = 0.2$, $\theta = 0$, $\varphi = 0$, and (b) $\sigma_u = 0.6$, $\theta = 3\pi/8$, $\varphi = 0$. The center of the functions $(u_0, v_0) = (25, 10)$.

Algorithm 1. *Matching pursuit* (*the tth iteration*)

(*I*) *Select and Updatea coefficient from*
$S = \{\phi_i \in \mathscr{D}_G \mid \forall j \neq i, |\mathbf{p}_i| > |\mathbf{p}_j|$, *where* $\mathbf{p}_k = \phi_k^{\mathrm{T}} \cdot \mathbf{r}$,
for $k = 1, \ldots, M.\}$ *so that* $\Delta \mathbf{c}_i(t) = \begin{cases} \mathbf{p}_i(t) & \text{if } \phi_i \in S(t), \\ 0 & \text{otherwise.} \end{cases}$

(*II*) *Update the residual image* $\Delta \mathbf{r}(t+1) = -\sum_{i=1}^{M} \Delta \mathbf{c}_i(t) \phi_i$
and the inner products $\Delta \mathbf{p}_i(t+1) = \phi_i^{\mathrm{T}} \cdot \Delta \mathbf{r}(t+1)$,
for $i = 1, \ldots, M$.

Note that we can improve the overall performance by suitably choosing the initial image of matching pursuit. In our implementation, the initial images are composed of a linear combination of the so-called "*unit Gabor atoms*" $\hat{\phi}_{\mathbf{x}}$ with weights $f(\mathbf{x})$ for all $\mathbf{x} \in \mathscr{P}_{\exp}$. Here $\hat{\phi}_{\mathbf{x}}$'s are the Gabor functions in which $\sigma_u = 0.1$, $\theta = 0$, $\varphi = 0$ and the center $(u_0, v_0) = \mathbf{x}$. That is, we use

$$\sum_{\mathbf{x} \in \mathscr{P}_{\exp}} f(\mathbf{x}) \hat{\phi}_{\mathbf{x}}$$

as the initial image in each iteration of Step (II.3) while decomposing the image

$$\mathscr{S}_{\mathscr{P}_{\exp}} = \sum_{\mathbf{x} \in \mathscr{P}_{\exp}} f(\mathbf{x}) \mathbf{e}_{\mathbf{x}}.$$

It is worth noting that the accuracy requirements of image decomposition vary over the whole course of iterations. In other words, it is not always necessary to find precise approximations when decomposing the surfaces $\mathscr{S}_{\mathscr{P}_{\text{exp}}}$. A mild accuracy tolerance of the image decomposition may be sufficient in the beginning stages, since the target surface $\mathscr{S}_{\mathscr{P}_{\text{exp}}}$ may still be far away from the final response surface. Thus, an accurate image approximation does not necessarily lead to correct prediction of the true response surface or the EPs. However, as the number of experimental points increase, the surrogate surface gradually approaches the real true response surface. A higher accuracy is considered more suitable. How the accuracy necessary to lead to efficient performance is adaptively determined remains an interesting open question.

2.3. The algorithm

We integrate all the ideas discussed above in Algorithm 2 and then provide convergence and complexity analysis of the algorithm in this section. Some possible generalizations of the algorithm are also discussed. Note that same notations introduced in Section 2 are used to describe the algorithm.

Algorithm 2. *Effective points search algorithm*

(I) *Initialize the problem.*

- *(1)* *Choose grid* $\mathscr{G}$ *containing* $N = p_1 p_2 \cdots p_n$ *points and maximum number of experimental points* $N_{\text{exp}} \leqslant N$.
- *(2)* *Choose* $\mathscr{P}_{\text{init}} \subset \mathscr{G}$, *where* $\mathscr{P}_{\text{init}}$ *contains* N_{init} *points.*
- *(3)* *Choose* $\mathscr{D} = \{\phi_j | \phi_j \in \mathbb{R}^N,\ \ j = 1, \ldots, M,\ \text{and}\ M > N\}$ *by Eqs. (5)–(7).*
- *(4)* *Set* $\mathscr{P}_{\text{new}} = \mathscr{P}_{\text{init}}$, $\mathscr{P}_{\text{exp}} = \mathscr{P}_{\text{init}}$, *and* $\mathscr{P}_{\text{EF}} = \phi$
- *(5)* *Compute* $f(\mathbf{x})$ *for all* $\mathbf{x} \in \mathscr{P}_{\text{init}}$.

(II) *Repeat (II.1)–(II.4) until* ($\mathscr{P}_{\text{new}} = \emptyset$ *or* $\mathscr{P}_{\text{exp}} = \mathscr{G}$ *or* $|\mathscr{P}_{\text{exp}}| \geqslant N_{\text{exp}}$)

- *(1)* *Compute* c_j*'s by Algorithm 1 so that* $\tilde{\mathscr{S}}_{\mathscr{P}_{\text{exp}}} = \sum c_j \phi_j \approx \mathscr{S}_{\mathscr{P}_{\text{exp}}} \equiv \sum_{\mathbf{x} \in \mathscr{P}_{\text{exp}}} f(\mathbf{x}) \mathbf{e}_{\mathbf{x}}$.
- *(2)* *Set* $\mathscr{P}_{\text{new}} = \{\tilde{\mathbf{x}} \mid \tilde{\mathbf{x}} \in \mathscr{G} \backslash \mathscr{P}_{\text{exp}}$, *and* $\tilde{\mathscr{S}}_{\mathscr{P}_{\text{exp}}}|_{\tilde{\mathbf{x}}}$ *satisfy the ROI conditions.*$\}$
- *(3)* *Compute* $f(\tilde{\mathbf{x}})$ *for all* $\tilde{\mathbf{x}} \in \mathscr{P}_{\text{new}}$.
- *(4)* *If* ($f(\tilde{\mathbf{x}})$ *satisfies the ROI condition for* $\tilde{\mathbf{x}} \in \mathscr{P}_{\text{new}}$) *then* *Update* $\mathscr{P}_{\text{EF}} = \mathscr{P}_{\text{EF}} \cup \tilde{\mathbf{x}}$ *end.*
- *(5)* *Update* $\mathscr{P}_{\text{exp}} \equiv \mathscr{P}_{\text{exp}} \cup \mathscr{P}_{\text{new}}$.

(III) *Output* $\mathscr{P}_{\text{EF}}$.

Now we discuss the convergence of Algorithm 2. If the domain is a grid, as we assume in this article, there is only a finite number of feasible points to be choose from. Algorithm 2 would ultimately find the EPs since the algorithm would visit all of then grid points eventually given the following two assumptions. First, we set $N_{\text{exp}} = N$. Secondly, whenever $\mathscr{P}_{\text{new}} = \emptyset$, we randomly choose one point from $\mathscr{G} \backslash \mathscr{P}_{\text{exp}}$ and then add the point into $\mathscr{P}_{\text{new}}$. By adopting the two assumptions discussed above, Algorithm 2 converges because it would eventually find all the EPs in the grid. On the other hand, if $\mathscr{X}$ is a continuous bounded domain and the EPs are the minimal (or maximal) points, the effective points search problem is actually equivalent to an optimization problem. Algorithm 2 can be modified to obtain asymptotic convergence of the optimization problem. To achieve this goal, Algorithm 2 needs to incorporate dynamic grids that are determined adaptively and the grid sizes should approach 0 while iterating the algorithm. The details of these modifications are investigated by the authors in another project. However, it is worth mentioning that the modified algorithm is closely related to the pattern search algorithm in which asymptotic convergence is justified [18]. Finally, as we use matching pursuit (Algorithm 1) in Step (II.1), we also need to look at the convergence of matching pursuit.

Basically matching pursuit is a greedy strategy for computing the suboptimal surface approximation. Regardless of whether finite or infinite dimensional Hilbert space is used, the convergence of the norm of residual vector, $\|\mathbf{r}\|$, to 0 is shown in [11]. In other words, for finite-dimensional problems, matching pursuit can be applied to approximate the reference response surfaces in our algorithm.

The complexity of Algorithm 2 can be analyzed theoretically as follows. In the algorithm, Step (I) is a one-time initial overhead and Steps (II.2), (II.4), and (II.5) are updating procedures. The computational cost of these steps is relatively cheap compared to the computational intensive tasks in Steps (II.1) and, probably, (II.3). Therefore, we will focus on these two steps to analyze the complexity of the algorithm. Mallat and Zhang [11] have shown that, under some mild assumptions, the matching pursuit converges in $\mathcal{O}(\overline{M}N\log_2 N)$, where $\overline{M}$ is the number of bases chosen by matching pursuit as shown in Algorithm 1 and N is the total grid number. Consequently, the worst-case theoretical complexity of the algorithm is $\mathcal{O}(N_{\text{exp}}\overline{M}N\log_2 N)$, as Step (II) will be iterated at most N_{exp} times, where $N_{\text{exp}} \leqslant N$. The choice of N_{exp} is rather empirical, especially when we have little knowledge regarding the function. Basically one can choose N_{exp} depending on the availability of affordable computational resource. It is worth mentioning that practical performance of the algorithm can be much better than the theoretical worst-case bound. As shown in [11,10], the residual of the matching pursuit converges exponentially. In other words, the matching pursuit can decompose the surface efficiently to satisfy a suitably relaxed noise $\varepsilon_{\mathscr{P}_{\text{exp}}}$ in Eq. (11) by a small $\overline{M}$. Furthermore, our numerical experience shows that N_{exp} does not need to be large to obtain reasonable results. In the function evaluations in Step (II.3), the cost depends on the definition of the function itself. Let C_{f} be the cost for evaluating function value once. The total cost for function evaluations in the algorithm is thus equal to $|\mathscr{P}_{\text{exp}}|C_{\text{f}}$ and the worst-case upper bound is $N_{\text{exp}}C_{\text{f}}$. Combining all of the discussions above, we obtain the complexity of the algorithm

$$\mathcal{O}(N_{\text{exp}}(\overline{M}N\log_2 N + C_{\text{f}})).$$

Finally, we note that C_{f} can dominate the complexity, provided the cost for evaluating function is very expensive.

3. A two-dimensional real problem

In this section, we first introduce a two-dimensional model problem and then demonstrate how this problem can be solved by applying Algorithm 2.

3.1. The model problem

A dynamical system modeling absorptive bistable laser diodes with an electronic-controlled external drive was studied in [19]. The dynamical system is defined by the following rate equations:

$$\eta\frac{\mathrm{d}N_{\text{e1}}}{\mathrm{d}T} = S_{\text{p1}} + m_{\text{c}}\sin(2\pi\cdot m_{\text{f}}\cdot T) - \alpha_1 N_{\text{e1}}^2 - (\alpha_2 N_{\text{e1}}^2 + N_{\text{e1}} + \alpha_3)N_{\text{p}} - N_{\text{e1}}, \tag{12}$$

$$\eta\frac{\mathrm{d}N_{\text{e2}}}{\mathrm{d}T} = S_{\text{p2}} - \alpha_1 N_{\text{e2}}^2 - (\alpha_2 N_{\text{e2}}^2 + N_{\text{e2}} + \alpha_3)N_{\text{p}} - \alpha_4 N_{\text{e2}}, \tag{13}$$

$$\frac{\mathrm{d}N_{\text{p}}}{\mathrm{d}T} = N_{\text{p}}[\gamma_1(\alpha_2 N_{\text{e1}}^2 + N_{\text{e1}} + \alpha_3) + \gamma_2(\alpha_2 N_{\text{e2}}^2 + N_{\text{e2}} + \alpha_3)] - N_{\text{p}} + \varepsilon(\gamma_1 N_{\text{e1}}^2 + \gamma_2 N_{\text{e2}}^2). \tag{14}$$

The study aimed to assert the existence of chaotic light output due to the system and then to apply the light output to secure optical communications. One essential indicator characterizing the dynamical system is the Lyapunov exponents. A positive Lyapunov exponent implies that the system is chaotic for the corresponding parameter settings. More details regarding the definition and computation of Lyapunov exponents can be found in [14]. Here we intend to find certain combinations of the adjustable parameter S_{p1} (the pump rate) and m_{c} (the modulation current) in Eq. (12), such that the associated Lyapunov exponents are positive.

Finding suitable parameter combinations leading to positive Lyapunov exponents is a challenging problem for the following two reasons. First, computing Lyapunov exponents of the dynamical system is extremely time consuming. Secondly, the relations between the adjustable parameters and the resulting Lyapunov exponents are exceeding complicated. Therefore, it is essential to develop a practical numerical scheme that is capable of efficiently identifying the desired parameter combinations among all the feasible parameters.

3.2. Numerical results

The specific algorithm discussed in Section 2.2 is implemented by Matlab [8] to conduct numerical experiments. The Lyapunov exponents were computed by the Fortran 95 codes developed in [19]. The codes implement the algorithms proposed in [14]. In our experiments, we choose the grid set

$$\mathscr{G} = \{(S_{\mathrm{p1}}, m_{\mathrm{c}}) | S_{\mathrm{p1}} \in \{20, 20.5, \ldots, 30\} \text{ and } m_{\mathrm{c}} \in \{5, 5.5, \ldots, 15\}\}. \tag{15}$$

In other words, the grid contains 441 (21×21) points in the two-dimensional domain $[20, 30] \times [5, 15]$. In Step (I.2) of Algorithm 2, we choose nine initial experimental points to form the set $\mathscr{P}_{\mathrm{init}} = \{(S_{\mathrm{p1}}, m_{\mathrm{c}}) | S_{\mathrm{p1}} \in \{22, 25, 28\}$ and $m_{\mathrm{c}} \in \{7, 10, 13\}\}$. For the Gabor dictionary, we set $\sigma_u = \{0.2, 0.6, 1.0\}$, $\theta = \{0, \pi/8, 2\pi/8, \ldots, 7\pi/8\}$, and phase $\varphi = 0$. The Gabor dictionary thus contains $M = 3 \times 8 \times 1 \times 441 = 10,584$ atoms.

Before presenting numerical results, we first illustrate what the true response surface looks like. We compute all the Lyapunov exponents over the grid points in $\mathscr{G}$ and demonstrate the results in Fig. 3. As shown in the figure, the surface is smooth in the left part; but, it oscillates violently in the right. For a surface such as this, traditional RSM is usually not efficient. Our numerical experiments suggest that the proposed algorithm is promising.

Here we highlight observations from the experimental results.

(1) The algorithm successfully identified an EP $\mathbf{x}$, where $f(\mathbf{x}) > 0$, in the 11th iteration. In this case, 20 experimental points (9 initial points plus 11 chosen experimental points) were used. In other words, the algorithm only evaluates Lyapunov exponents on 4.5% of all the possible grid points in $\mathscr{G}$ to find an EP to solve the target problem. Instead of blindly scanning all the grid points, the algorithm saves the user a significant amount of time.

(2) To explore the overall performance of the algorithm, we continue the process until all the grid points that are associated with positive Lyapunov exponents are found. The algorithm successfully found all 21 EPs. To be precise, the algorithm identifies positive Lyapunov exponents when 20, 24, 25, 28, 38, 40, 44, 61, 68, 71, 87, 88, 95, 98, 100, 105, 112, 114, 115, 116, and 147 experimental points are used. These results suggest that the

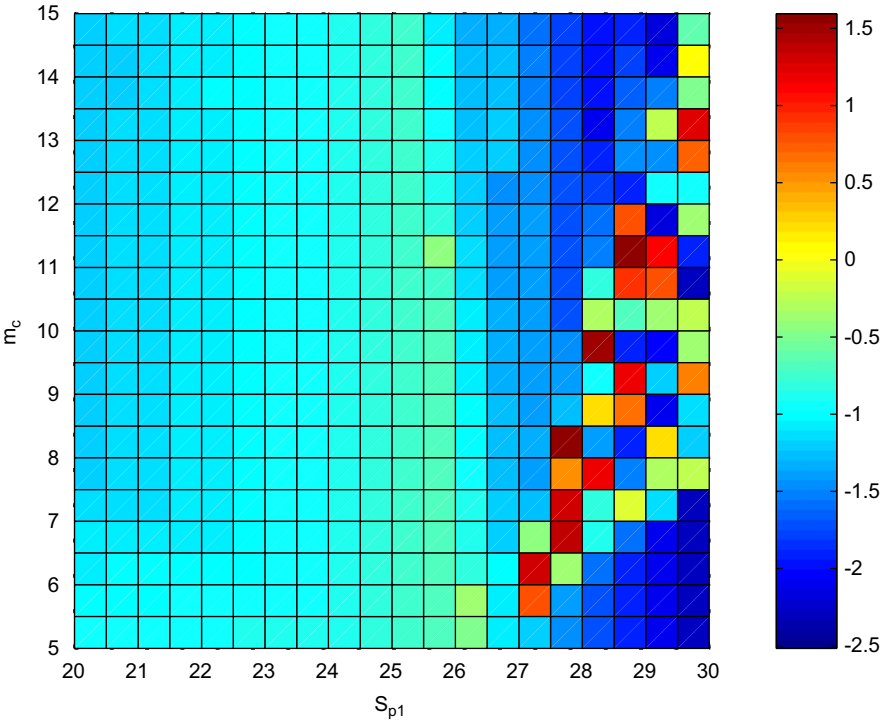

Fig. 3. The true response surface over the grid. Values of the Lyapunov exponents for $S_{\mathrm{p1}} \in [20, 30]$ and $m_{\mathrm{c}} \in [5, 15]$ are shown in color.

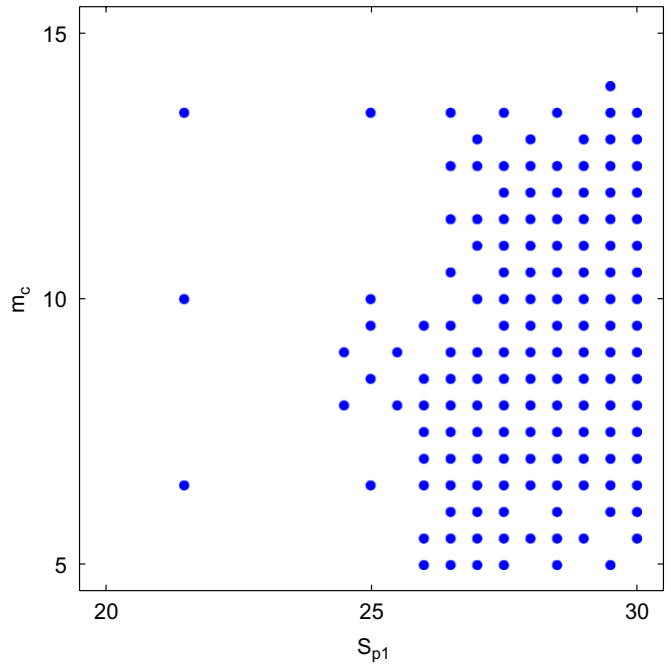

Fig. 4. Profile of the first 147 experimental points. The algorithm identifies all 21 effective points by using these points. The profile shows that the algorithm focuses on the right part of the domain, where the ROI of the model problem is located, to search for the effective points.

algorithm is able to approximate the response surface to assist in searching the EPs. It is also efficient since only one-third of the points in $\mathscr{G}$ are needed to find all EPs.

(3) As shown in Fig. 4, the algorithm chooses the 147 experimental points, which are largely located in the right part of the domain and successfully identifies all of the 21 EPs. The results show that the algorithm correctly predicts the trend of the surface by continuing to search for EPs and refining the response surface in the right part of the grid.

(4) Fig. 5 shows four surrogate surfaces containing 20, 50, 93, and 147 experimental points, which are shown in each of the sub-figures, respectively. It is clear that as the searching process continues, the surrogate surface becomes closer to the true response surface shown in Fig. 3. In Part (D) of Fig. 5, where all EPs have been identified, the surrogate surface is quite similar to the actual response surface. This also suggests that the Gabor dictionary is suitable for the response surface in this example.

(5) As shown in Fig. 3, the response surface contains several ridges in the northeast–southwest directions. Similar structures with different directions and different sizes can also be observed in Fig. 5. The directions and sizes of the surrogate surfaces are gradually corrected with the computation of additional Lyapunov exponents. Such consequences are due to the Gabor dictionary, whose atoms are able to represent directional ridges controlled by the parameter θ.

3.3. Results of the RSM

We also solve the problem described in Section 3.1 by RSM and the results are presented in this section. Although we are interested in finding the positive Lyapunov exponents, the problem can also be solved by using RSM to successively find local maxima. General response surface methodology usually assumes the model of responses contains a white noise term. Hence the replicate observations are collected at the experiment points and then analysis of variance (ANOVA) is used to determine the order of the polynomial for the approximation model. Since the target problem has no replication, we simplify the RSM procedure by using only the second order polynomial model in this article.

To conduct the experiments, we use each of the grid points described in (15), except the boundary points, as the initial points so that we have 361 (19×19) initial grid points. We find the behavior of the RSM associated with all these

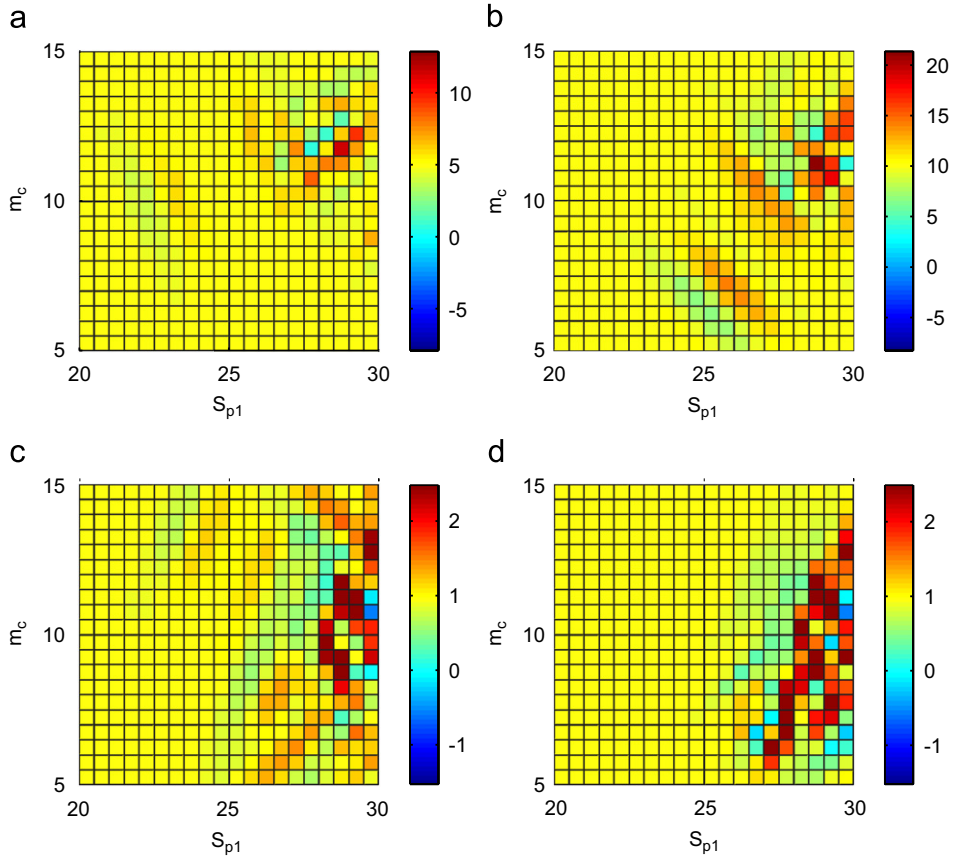

Fig. 5. Evolution of the surrogate surfaces generated by Algorithm 2. The four surrogate surfaces containing 20, 50, 93, and 147 experimental points are shown in each of the sub-figures, respectively.

initial points can be classified into the three categories as shown in Fig. 6: (a) the RSM path hits the boundary of the domain and fail to locate a local maximum; (b) and (c) the RSM successfully finds a local maximum with a negative Lyapunov exponent; (d) the RSM successfully finds a local maximum with a positive Lyapunov exponent. Clearly the RSM is capable of finding local maxima that are close to the initial points. However, the RSM is easily "trapped" by the local maxima associated with negative Lyapunov exponents, if the initial points are roughly located in the left part of the grid (Fig. 6(a)–(c)). Furthermore, if the initial points are located in the right part of the grid, the second-order polynomial models are also not suitable for the oscillatory surface. Therefore, RSM is not recommended for a problem in which the corresponding true response surface is complicate and oscillatory.

While the qualitative analysis is presented in Fig. 6, Table 1 demonstrates the quantitative performance of RSM. The table shows the number of successful and failed trials ("Succ." and "Fail." in the table) observed when attempting to locate a positive Lyapunov exponent and the global maximum. The table also shows that if we pick an initial point blindly, RSM has 24% and 3% probabilities of locating a positive Lyapunov exponent and the global maximum, respectively. Furthermore, we also compare the experimental points ("Pt. no." in the table) used by RSM and EPSA to locate positive Lyapunov exponents or the global maximum. When RSM successfully locates a positive Lyapunov exponent, it requires an average of 13 experimental points. Clearly RSM does not require many experimental points to locate a positive Lyapunov exponent if the starting point is near the target point. In contrast, EPSA requires 20 and 100 experimental points to find the first and the maximum positive Lyapunov exponent, respectively.

Both Fig. 6 and Table 1 suggest that RSM is highly dependent on the initial point and that the current version of RSM may not be suitable for finding multiple positive Lyapunov exponents (or multiple extreme values). For example, the probability for RSM to find two positive Lyapunov exponents by starting from two arbitrary initial points is roughly

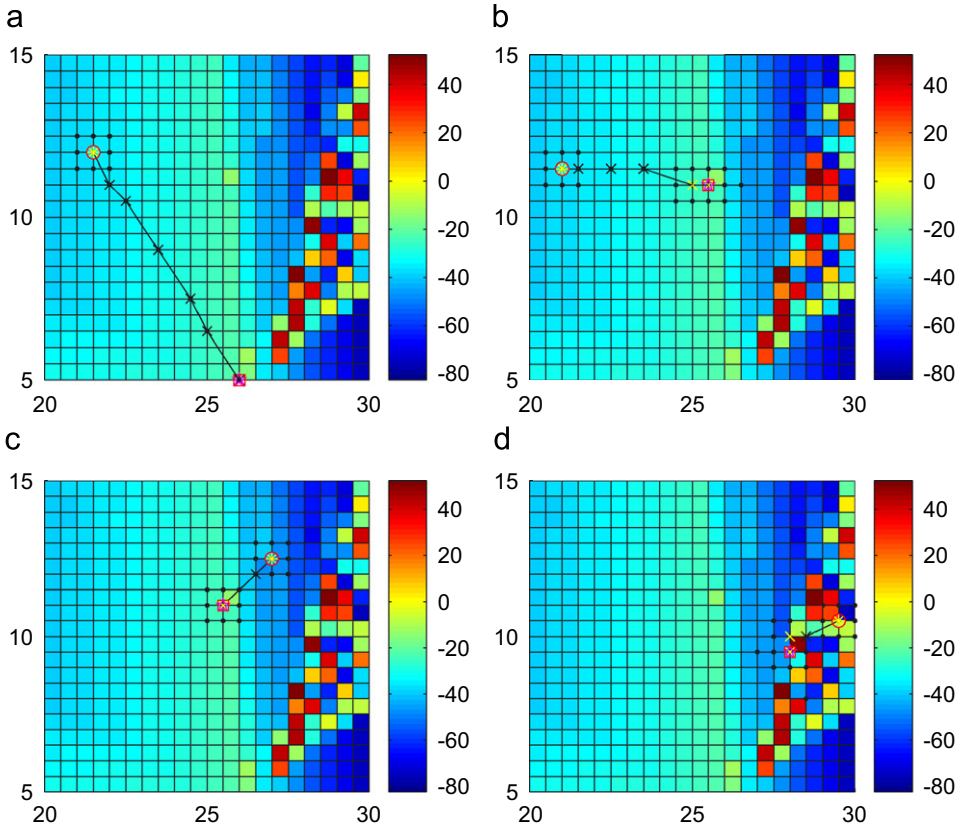

Fig. 6. Three typical RSM paths generated while solving the target problem described in Section 3.1. (a) The RSM path hits the boundary of the domain and fail to locate local maxima. (b) and (c) The RSM successfully finds a local maximum with a negative Lyapunov exponent. (d) The RSM successfully finds a local maximum with a positive Lyapunov exponent. The starting and ending points of the RSM paths are represented by circles and squares, respectively.

Table 1
Performance summary of RSM and effective points search algorithm

Target	RSM				EPSA
	Succ.	Fail.	Prob.	Pt. no.	Pt. no.
Max. pos. L.E.	12	349	3%	13	100
One pos. L.E.	87	274	24%	13	20
All pos. L.E.	N/A	N/A	N/A	N/A	147

equal to 6% ($\approx 0.24^2$). The probability of success drops quickly as additional positive Lyapunov exponents are needed. In contrast, EPSA can continue locating all the positive Lyapunov exponents in an efficient manner. However, we would like to point out that RSM can be modified to find multiple extreme values in a more efficient manner. This modified RSM has been developed by the authors and will be presented in another article.

We conclude this section with the following note. Traditional RSMs typically use a linear or second-order polynomial model. It is easy to obtain the steepest ascent or descent directions in these models. However, direction determination may be difficult for complicated models and problems with multiple extremes. In contrast, our scheme (Algorithm 2) does not attempt to find the search directions. Instead, we choose the next potentially EPs directly with the assistance of the surrogate models. This direct search approach allows us to find EPs that are not related to extreme values. For example, we can apply this approach to find all EPs that result in positive function values. Furthermore, since the true

response surface is approximated by a set of bases, less smoothing assumption of the true response surface is needed here. If the bases are appropriately chosen, the approximation can be very efficient.

4. Conclusions

We have proposed a novel algorithm to solve effective point pursuit problems. The algorithm constructs the surrogate models iteratively by using the overcomplete dictionary based vector decompositions skills. The numerical results are quite promising even for a model problem in which the response surface is complicated and composed of both smooth and violent oscillatory portions. Further improvement in performance of the algorithm can be achieved by developing more efficient methods for choosing bases, decomposing the response vectors, and selecting experimental points. Algorithm 2 can also be generalized in various manners. For example, we can use other base dictionaries that are more suitable for specific problems. Other methods, e.g., basis pursuit, can be used to approximate the true response surface. Besides, the algorithm has a great potential to be parallelized in many parts of the algorithm. We are also conducting a project to investigate the performance of the proposed algorithm on various type of the true response surfaces.

Acknowledgments

The authors are grateful to Dianne P. O'Leary, Ying Nian Wu, John E. Dennis, Eric Jaehnig, and the anonymous referees for the helpful comments and suggestions. This work is partially supported by the National Science Council and the National Center for Theoretical Sciences in Taiwan.

References

[1] G.E. Box, N.R. Draper, Empirical Model-Building and Response Surface, Wiley, New York, 1987.
[2] S. Chen, D.L. Donoho, M.A. Saunders, Atomic decomposition by basis pursuit, SIAM J. Sci. Statist. Comput. 20 (1998) 33–61.
[3] V. Cheynier, M. Feinberg, C. Chararas, C. Ducauze, Application of response surface methodology to evaluation of bioconversion experimental conditions, Appl. Environmental Microbiology 45 (2) (1983) 634–639.
[4] R.R. Coifman, M.V. Wickerhauser, Entropy-based algorithms for best basis selection, IEEE Trans. Inform. Theory 38 (2) (1992) 713–718.
[5] W. Czaja, Characterizations of gabor systems via the fourier transform, Collect. Math. 51 (2) (2000) 205–224.
[6] I. Daubechies, Time-frequency localization operators: a geometric phase space approach, IEEE Trans. Inform. Theory 34 (4) (1988) 605–612.
[7] D.L. Donoho, M. Elad, V. Temlyakov, Stable recovery of sparse overcomplete representations in the presence of noise, IEEE Trans. Inform. Theory 52 (1) (2006) 6–18.
[8] D.J. Higham, N.J. Higham, MATLAB Guide, SIAM, Philadelphia, PA, 2000.
[9] I.A. Khuri, J.A. Cornell, Response Surface: Designs and Analysis, Dekker, New York, 1996.
[10] S. Mallat, A Wavelet Tour of Signal Processing, Academic Press, New York, 1993.
[11] S. Mallat, Z. Zhang, Matching pursuit with time-frequency dictionaries, IEEE Trans. Signal Process. 41 (1993) 3397–3415.
[12] R.H. Myers, D.C. Montgomery, Response Surface Methodology: Process and Product Optimization Using Designed Experiments, Wiley, New York, 2002.
[13] N. Papila, W. Shyy, L.W. Griffin, D.J. Dorney, Shape optimization of supersonic turbines using response surface and neural network methods, J. Propulsion and Power 18 (2002) 509–518.
[14] T.S. Parker, L.O. Chua, Practical Numerical Algorithms for Chaotic Systems, Springer, Berlin, 1989.
[15] A. Pece, N. Petkov, Fast atomic decomposition by the inhibition method, in: Proceedings of the 15th International Conference on Pattern Recognition, 2000, pp. 215–218.
[16] S. Qian, D.P. Chen, Signal representation using adaptive normalized gaussian functions, Signal Process. 36 (1) (1994) 1–11.
[17] L. Theodorsen, J.W. London, L.M. Shaw, J.H. Stromme, Application of response surface methodology to the assay of gamma-glutamyltransferase, Clinical Chemistry 28 (1982) 1140–1143.
[18] V. Torczon, On the convergence of pattern search algorithms, SIAM J. Optim. 7 (1) (1997) 1–25.
[19] W. Wang, T.-M. Hwang, C. Juang, J. Juang, C.-Y. Liu, W.-W. Lin, Chaotic behaviors of bistable laser diodes and its application in synchronization of optical communication, Japanese J. Appl. Phys. 40 (10) (2001) 5914–5919.