Improved decision support for engine-in-the-loop experimental design optimization

D Gladwin[1], **P Stewart**[2*], **J Stewart**[2], **R Chen**[3], and **E Winward**[3]
[1]Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, UK
[2]School of Engineering, University of Lincoln, Lincoln, UK
[3]Department of Aeronautical and Automotive Engineering, Loughborough University, Loughborough, UK

The manuscript was received on 16 March 2009 and was accepted after revision for publication on 29 July 2009.

DOI: 10.1243/09544070JAUTO1213

Abstract: Experimental optimization with hardware in the loop is a common procedure in engineering and has been the subject of intense development, particularly when it is applied to relatively complex combinatorial systems that are not completely understood, or where accurate modelling is not possible owing to the dimensions of the search space. A common source of difficulty arises because of the level of noise associated with experimental measurements, a combination of limited instrument precision, and extraneous factors. When a series of experiments is conducted to search for a combination of input parameters that results in a minimum or maximum response, under the imposition of noise, the underlying shape of the function being optimized can become very difficult to discern or even lost. A common methodology to support experimental search for optimal or suboptimal values is to use one of the many gradient descent methods. However, even sophisticated and proven methodologies, such as simulated annealing, can be significantly challenged in the presence of noise, since approximating the gradient at any point becomes highly unreliable. Often, experiments are accepted as a result of random noise which should be rejected, and vice versa. This is also true for other sampling techniques, including tabu and evolutionary algorithms.

After the general introduction, this paper is divided into two main sections (sections 2 and 3), which are followed by the conclusion. Section 2 introduces a decision support methodology based upon response surfaces, which supplements experimental management based on a variable neighbourhood search and is shown to be highly effective in directing experiments in the presence of a significant signal-to-noise ratio and complex combinatorial functions. The methodology is developed on a three-dimensional surface with multiple local minima, a large basin of attraction, and a high signal-to-noise ratio.

In section 2, the methodology is applied to an automotive combinatorial search in the laboratory, on a real-time engine-in-the-loop application. In this application, it is desired to find the maximum power output of an experimental single-cylinder spark ignition engine operating under a quasi-constant-volume operating regime. Under this regime, the piston is slowed at top dead centre to achieve combustion in close to constant volume conditions.

As part of the further development of the engine to incorporate a linear generator to investigate free-piston operation, it is necessary to perform a series of experiments with combinatorial parameters. The objective is to identify the maximum power point in the least number of experiments in order to minimize costs. This test programme provides peak power data in order to achieve optimal electrical machine design.

The decision support methodology is combined with standard optimization and search methods – namely gradient descent and simulated annealing – in order to study the reductions possible in experimental iterations. It is shown that the decision support methodology significantly reduces the number of experiments necessary to find the maximum power

*Corresponding author: School of Engineering, University of Lincoln, Brayford Pool, Lincoln, LN6 7TS, UK.
email: pstewart@lincoln.ac.uk

solution and thus offers a potentially significant cost saving to hardware-in-the-loop experimentation.

Keywords: experimental decision support, variable neighbourhood search, gradient descent, simulated annealing, hardware in the loop

1 INTRODUCTION

An experimental search with hardware or process in the loop is a common procedure in, for example, the engineering and pharmaceutical industries. These procedures are often applied to combinatorial problems during (in the engineering case, for example) the development of new hardware systems or control [**1**, **2**]. The systems under test can be described as a set of dependent variables that vary according to some functions of independent variables. In this case, there does not exist a complete specification of the function relating the variables. This implies that there is no accurate *a priori* knowledge of the fundamental cause and effect present in the system. Thus, for an example linear function, the values in the coefficients matrix would be unknown.

Commonly, it is required to identify the sets of independent variables that maximize or minimize the dependent variables [**3**]. To obtain the necessary information to have a confident estimate of the parameters, it is possible to vary the independent parameters over successive trials (designed experiments) and to measure the corresponding dependent variables. In order to examine this relationship fully, a large number of trials is often required to identify the location of the desired response. However, real-world problems are difficult to solve by this methodology for a number of reasons [**4**].

1. The number of possible solutions in the experimental space is so large as to preclude an exhaustive search for the best (or acceptable) answer.
2. The evaluation function that describes the solutions is extremely noisy and/or complex.
3. The cost of conducting an experiment at many points in the search space may be prohibitive in terms of time taken and/or resources used.

These constraints motivate the use of gradient descent (GD) methods in order to provide the decision support to direct the search and to minimize the number of experiments conducted. Other metaheuristics, such as genetic algorithms [**5**], are applicable to this class of problem, but are relatively difficult to implement and tune because of the number of parameters associated with this technique in comparison with GD methods. These methods are based upon the statistics of the neighbourhood around a given point, thus relying on local information at each step. However, basic GD methods provide only locally optimum solutions whose values depend on selection of the starting point [**6**]. There have been many metaheuristic methods developed to increase the efficiency of the experimental search, such as simulated annealing (SA) [**7**], tabu search [**8**], genetic algorithms [**2**], and variable neighbourhood search (VNS) [**9**].

As the nature of the experimental surface is an unknown, it is important to utilize methodologies that require the minimum number of parameters to be 'tuned' in order to conduct an effective search. With this caveat in mind, simple GD, SA, and VNS will be considered in this paper since, in most of their varieties, implementation is simple and basic tuning rules are readily available, making them commonly used heuristics in the experimental community.

It has been noted in the literature that the performance of metaheuristics such as SA are to some extent compromised when directing search over significantly noisy surfaces [**4**]. This paper describes the implementation of a weighted stochastic decision support (WSDS) operator based on response surfaces (RSs) which supports the heuristic and guides the experimental process to predicted areas of interest in the search space. Basic GD, SA, and VNS are supplemented by this methodology, and performance is compared with the basic form of the metaheuristics. The supplemented metaheuristics are shown to have significantly improved performance when searching noisy environments.

1.1 Scope of this paper

This paper is primarily concerned with the development and assessment of a novel problem-solving methodology for practical engineering applications, and hence it is helpful to define the scope of research presented here. This paper is primarily concerned with the following:

(a) experimentation on real engineering problems;

(b) problems that contain inherently noisy data and processes;
(c) decision support to reduce experimentation time;
(d) applying a decision support operator to common search methodologies.

It is important to note that the emphasis here is on the reduction of experimentation time by decision support and hence does not consider the following:

(a) performance comparison of heuristics and metaheuristics;
(b) metaheuristic tuning methodologies;
(c) 'toy' problems and surfaces.

Thus, the paper is concerned with the problem of finding a result in an unseen noisy search space, where every individual evaluation of a point in the search space is expensive in terms of time and/or resources.

2 ALGORITHM DEVELOPMENT STUDY

2.1 Search methodologies

A comprehensive description of GD-based methods can be found in reference [**4**]. In this section, the implementations of the algorithms are described. The class of problems addressed in this paper are in general of a minimization type [**10**, **11**]; i.e. it is desired to minimize a function $f(x)$ over choices of x that lie in the feasible set S such that

$$f^* = \min_{x \in S} f(x) \tag{1}$$

Thus, $\exists x^* \in S$, such that $f^* = f(x^*)$ and $x \in S \Rightarrow f(x^*) \leqslant f(x)$. Set S is the constraint set, f^* is the minimum of the problem, while x^* is the minimizer. Minimization problems are considered in this paper; however, conversion of the method to maximization is a trivial task.

2.1.1 Gradient descent

A sequence of intermediate values are successively generated by the algorithm. First, an initial random guess is made and then it is successively improved. In general, none of the iterates exactly solves the problem; therefore, a termination criterion is included that, when satisfied, will cause the algorithm to terminate with a suitable approximation to the exact solution. This is particularly applicable to real-world noisy surfaces. Iterative hill descent can be described with the general form of recursion

$$\boldsymbol{x}^{v+1} = \boldsymbol{x}^v + \alpha^v \, \Delta \boldsymbol{x}^v, \quad v = 0, 1, 2, \ldots \tag{2}$$

where

$\boldsymbol{x}^0$ = initial guess
v = iteration counter
$\boldsymbol{x}^v$ = value of the iterate at the vth iteration
$\alpha^v \in \Re_+$ = step size
$\Delta \boldsymbol{x}^v \in \Re^n$ = step direction
$\alpha^v \, \Delta \boldsymbol{x}^v$ = update to add to the current iterate x^v to obtain new iterate x^{v+1}

In the case of minimization, the step direction is chosen so as to reduce the objective f^*. If $\hat{x} \in \Re$, then the vector $\Delta \boldsymbol{x} \in \Re$ is called a descent direction for f at $\hat{\boldsymbol{x}}$ if $\exists \bar{\alpha} \Re_{++}$ such that

$$0 < \alpha \leqslant \bar{\alpha} \Rightarrow f(\hat{\boldsymbol{x}} + \alpha \, \Delta \boldsymbol{x}) < f(\hat{\boldsymbol{x}}) \tag{3}$$

and $\Delta \boldsymbol{x}$ is a descent direction for f at $\hat{\boldsymbol{x}}$ if the objective is smaller than $f(\hat{\boldsymbol{x}})$ at points along the line segment $\hat{\boldsymbol{x}} + \alpha \, \Delta \boldsymbol{x}$ for $\alpha > 0$ and $\alpha \leqslant \bar{\alpha}$. There are some caveats associated with GD methods.

1. The methods usually terminate at solutions which are only locally optimal.
2. No information is apparent as to how the discovered local optima deviates from the global minima or other local minima.
3. The optimum obtained depends on the original configuration.
4. In general it is not possible to calculate an upper bound for computation time.

GD thus exploits the best opportunities for improvements, but neglects to explore a large search space. In contrast, random search where points are sampled from S with equal probability explores thoroughly, but forgoes local exploitation. Thus, most GD methods execute a random 'jump' at local minima, to balance exploration with exploitation.

2.1.2 Variable neighbourhood search

The VNS algorithm implemented in this paper systematically exploits the idea of neighbourhood change in the descent to minima [**17**], and attempts to balance local exploitation with global exploration. It is simply an implementation of the basic GD method described in the previous section; however, in this case the step length α^v is variable rather than fixed. A number of variations have been reported, with both lengthening step length [**6**] and reducing step length [**9**]. In this case, reducing the step length

is implemented by a static schedule [**13**] using a step decrementation function given by

$$C_{k+1} = \alpha c_k, \quad k = 0, 1, 2, \ldots \tag{4}$$

The initial step length is usually chosen to be significant with respect to the search variable ranges, where α is chosen to be a positive constant greater than 1. The final value is fixed, generally related to the smallest feasible measurement or control increment of the variables. As with GD, a random 'jump' is implemented to escape local minima.

2.1.3 Simulated annealing

The implementation of SA used in this paper is based again on GD, accepting improvements in cost in traversing the search space; however, depending on a control parameter *c*, it will accept deteriorations to a limited extent to escape local minima. Initially, at large values of *c*, large deteriorations will be accepted; as *c* decreases, smaller deteriorations are accepted; finally, as the value of *c* approaches 0, no deteriorations are accepted. The probability of accepting deteriorations is achieved by comparing the value of $\exp\{[f(i) - f(j)]/c\}$ with a random number generated from a uniform distribution in the interval [0, 1]. In this case, the rate in decrease in the parameter *c* is achieved by implementing the VNS decrementation function.

2.2 Weighted stochastic decision support operator

Local search methods, such as GD, execute a random jump at local optima or other predefined termination metric based upon the implementation. SA-type methodologies typically execute a random jump at the termination of the cooling schedule if the global minimum or some upper bound of acceptable performance has not been reached. Obviously, with unknown experimental functions, the exact value of the global maximum will not be known; however, it is common for the designer or experimenter to have an 'acceptable' performance metric in mind when starting the experimental procedure that will act as a termination criterion.

Tabu search has been shown to be a particularly effective metaheuristic by directing the experimental search 'jumps' away from areas that have been found to be unproductive. However, this does not take advantage of the previously gathered data with respect to the possibility of 'predicting' promising areas of search. The RS methodology has been shown to be an effective tool in approximating complex and noisy functions for real-time control [**1**, **14**] and thus would appear to be a useful tool for direct experimentation based upon past results.

The RS methodology is a technique that was initially developed to optimize process control and experimentation by the application of designed experiments in order to characterize the relationship between the system variables and outputs [**3**]. The relationship between the response variable of interest, *y*, and the predictor variables $(\xi_1, \xi_2, \ldots, \xi_k)$ provides a description of the system of the form

$$\boldsymbol{y} = g(\xi_1, \xi_2, \ldots, \xi_k) + \boldsymbol{\varepsilon} \tag{5}$$

where $\boldsymbol{\varepsilon}$ represents the model error and includes measurement error and other variables such as background noise. The error will be assumed to have a normal distribution with zero mean and variance σ^2. In general, the experimenter approximates the system function *g* with an empirical model of the form

$$\boldsymbol{y} = f(\xi_1, \xi_2, \ldots, \xi_k) + \boldsymbol{\varepsilon} \tag{6}$$

where *f* is a polynomial of arbitrary order (generally, first or second order in the process control industry). This is the empirical or RS model. The variables are known as natural variables since they are expressed in physical units of measurement. The natural variables are transformed into coded variables x_1, x_2, ..., x_k, which are dimensionless, zero mean, and the same standard deviation, in order to minimize the effects of outliers, sparse, or unevenly distributed data, which are all likely in practical experimental applications. This approach is standard practice in the literature and in industry [**3**]. The response function now becomes

$$\eta = f(x_1, x_2, \ldots, x_k) \tag{7}$$

The successful application of RSs relies on the identification of a suitable approximation for *f*. This will often be a first-order model of the form

$$\eta = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k \tag{8}$$

or a second-order model of the form

$$\eta = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k} \beta_{jj} x_j^2 + \sum_{i<j} \sum \beta_{ij} x_i x_j \tag{9}$$

It may also be necessary to employ an approximat-

ing function greater than an order of two, based on the standard Taylor series expansion. The set of parameters can be estimated by regression analysis based upon the experimental data. The method of least squares is typically used to estimate the regression coefficients. With $n < k$ on the response variable available, giving y_1, y_2, ... y_n, each observed response will have an observation on each regression variable, with x_{ij} denoting the ith observation of variable x_j. Assuming that the error term $\boldsymbol{\varepsilon}$ has $E(\boldsymbol{\varepsilon}) = 0$ and $\mathrm{Var}(\boldsymbol{\varepsilon}) = \sigma^2$ and that the ε_i are uncorrelated random variables, the model can now be expressed in terms of the observations

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \varepsilon_i, \quad i = 1, 2, \ldots, n \tag{10}$$

The coefficients β in equation (10) are chosen such that the sum of the squares of the errors ε_i are minimized via the least-squares function

$$L = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{n} \beta_j x_{ij} \right)^2 \tag{11}$$

The model can be more usefully expressed in matrix form as

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{12}$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1k} \\ 1 & x_{21} & x_{22} & \ldots & x_{1k} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & x_{n1} & x_{n2} & \ldots & x_{nk} \end{bmatrix},$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ . \\ \beta_n \end{bmatrix}, \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ . \\ . \\ . \\ \varepsilon_n \end{bmatrix} \tag{13}$$

It is now necessary to find a vector $\boldsymbol{b}$ of least-squares estimators which minimizes the expression

$$L = \sum_{i=1}^{n} \varepsilon_i^2 = \boldsymbol{\varepsilon}'\boldsymbol{\varepsilon} = (\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})'(\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}) \tag{14}$$

and yields the least-squares estimator of $\boldsymbol{\beta}$ which is

$$\boldsymbol{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y} \tag{15}$$

and, finally, the fitted regression model is

$$\hat{\boldsymbol{y}} = \mathbf{X}\boldsymbol{b}, \quad \boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}} \tag{16}$$

where $\boldsymbol{e}$ is the vector of residual errors of the model. The RS method can thus be implemented upon either simulated or actual experimental results to derive a polynomial expression describing the relationship between the causal inputs and resulting outputs of the dynamic systems under consideration.

As data from the experimental results are gathered under the direction of the metaheuristics, it is possible to generate a surface approximation for the system under consideration.

2.2.1 *WSDS method*

The WSDS method in its basic form is encapsulated in the following pseudo-code:

1. WHILE
2. run meta-heuristic to global minimum (or acceptable value)
3. END
4. ELSE
5. add *new path data* to *old path data*
6. fit *normalized RS* to *old path data*
7. generate WSDS surface
8. perform weighted jump
9. END

The response surface can take any arbitrary order; for descriptive purposes, a second-order support surface in two variables is described.

If the metaheuristic finds a global or acceptable minimum, then the procedure terminates. Otherwise, a random jump is made to escape the local minima. In this case, the most recent search data (*traveldata*) is added to a data history file (e_1, e_2, *yresp*), where e_1 and e_2 are the natural variables, and *yresp* is the response. Thus

$$\begin{aligned} e_1 &= [e_1;\ traveldata(:, 1)] \\ e_2 &= [e_2;\ traveldata(:, 2)] \\ yresp &= [yresp;\ traveldata(:, 3)] \end{aligned} \tag{17}$$

The natural variables e_1 and e_2 are transformed into

dimensionless zero-mean coded variables x_1 and x_2. As the surface is unknown, *a priori* knowledge of the minimum and maximum of the natural variables is not possible. Hence, they vary dynamically as new data arrives according to

$$x_1 = e_1 - \frac{[\max(e_1) + \min(e_1)]/2}{[\max(e_1) - \min(e_1)]/2}$$
$$x_2 = e_2 - \frac{[\max(e_2) + \min(e_2)]/2}{[\max(e_2) - \min(e_2)]/2} \tag{18}$$

For a representative second-order support surface, the variable matrix **X** is thus

$$\mathbf{X} = \begin{bmatrix} x_0 & x_1 & x_2 & x_1^2 & x_2^2 & x_1x_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \tag{19}$$

The variable matrix shown in equation (19) has a cross-coupling term x_1x_2 which is optional and is dependent on the surface being fitted. The coded estimated coefficients of the RS are calculated according to equation (15), which are converted to natural coefficients by inverting the process in equation (18). For the second-order cross-coupled approximation under examination, the estimated natural coefficients are calculated from

$$\hat{b}_0 = b_0 + b_1 \frac{-[\max(e_1) + \min(e_1)]/2}{[\max(e_1) - \min(e_1)]/2} + b_2 \frac{-[\max(e_2) + \min(e_2)]/2}{[\max(e_2) - \min(e_2)]/2} + b_3 \left\{ \frac{[\max(e_1) + \min(e_1)]/2}{[\max(e_1) - \min(e_1)]/2} \right\}^2$$
$$+ b_4 \left\{ \frac{[\max(e_2) + \min(e_2)]/2}{[\max(e_2) - \min(e_2)]/2} \right\}^2 + b_5 \frac{\{[\max(e_1) + \min(e_1)]/2\}\{[\max(e_2) + \min(e_2)]/2\}}{\{[\max(e_1) - \min(e_1)]/2\}\{[\max(e_2) - \min(e_2)]/2\}}$$
$$\hat{b}_1 = \frac{b_1}{[\max(e_1) - \min(e_1)]/2} - 2b_3 \frac{[\max(e_1) + \min(e_1)]/2}{\{[\max(e_1) - \min(e_1)]/2\}^2}$$
$$- b_5 \frac{[\max(e_2) + \min(e_2)]/2}{\{[\max(e_1) - \min(e_1)]/2\}\{[\max(e_2) - \min(e_2)]/2\}}$$
$$\hat{b}_2 = \frac{b_2}{[\max(e_2) - \min(e_2)]/2} - 2b_4 \frac{[\max(e_2) + \min(e_2)]/2}{\{[\max(e_2) - \min(e_2)]/2\}^2}$$
$$- b_5 \frac{(\max(e_1) + \min(e_1))/2}{\{[\max(e_1) - \min(e_1)]/2\}\{[\max(e_2) - \min(e_2)]/2\}}$$
$$\hat{b}_3 = \frac{b_3}{\{[\max(e_1) - \min(e_1)]/2\}^2}$$
$$\hat{b}_4 = \frac{b_4}{\{[\max(e_2) - \min(e_2)]/2\}^2}$$
$$\hat{b}_5 = \frac{b_5}{\{[\max(e_1) - \min(e_1)]/2\}\{[\max(e_2) - \min(e_2)]/2\}} \tag{20}$$

The natural coefficients are now recombined into the RS polynomial

$$y = b_0 + b_1\varepsilon_1 + b_2\varepsilon_2 + b_3\varepsilon_1^2 + b_4\varepsilon_2^2 + b_5\varepsilon_1\varepsilon_2 \tag{21}$$

where $[\varepsilon_1\ \varepsilon_2]$ is a matrix of coordinate values for the search space, and y is the corresponding predicted response value array. Since the 'true' system response surface is unknown, this predicted array represents the current view of the likely response. It is this polynomial that forms the basis for the weighted jump from local minima. The y values are normalized according to

$$y_{\text{norm}} = 1 - \left\{ y - \frac{[\max(y) + \min(y)]/2}{[\max(y) - \min(y)]/2} + 1 \right\} \Big/ 2 \tag{22}$$

which yields an RS over the search space bounded from zero to one, where increasing value represents increasing interest inferred from previous searches. These monotonically increasing values correspond to coordinates in a probability space from which the next jump coordinates are chosen according to a random number generation, with probability of being chosen based on relative value in the support space. Thus, the probability of selection of the next jump point is based statistically on the results of previous searches.

2.2.2 *Development surfaces*

For the development of this methodology, a realistic noisy surface with multiple local minima, plateaux, and one global minimum was considered for algorithm development. The standard MATLAB 'peaks' surface (Fig. 1) describes a combinatorial process in two variables as

$$y = 3(1-x_1)^2 \exp\left(-x_1^2 - x_2^2\right) - 10\left(\frac{x_1}{5} - x_1^3 - x_2^5\right)\exp\left(-x_1^2 - x_2^2\right) - \frac{1}{3}\exp\left[-(x_1+1)^2 - x_2^2\right] \quad (23)$$

In order to investigate the effects of noise, progressively larger amounts of Gaussian noise are added to the smooth surface (peaks 0) to give peaks 1 (Fig. 2), peaks 2 (Fig. 3), and peaks 3 (Fig. 4).

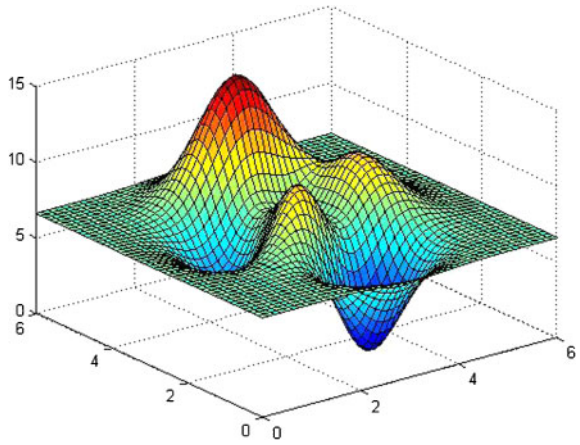

Fig. 1 Noise-free algorithm development fitness landscape: peaks 0

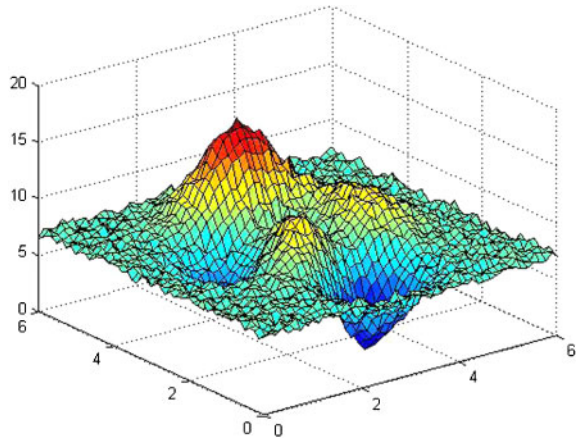

Fig. 2 Noisy algorithm development fitness landscape: peaks 1

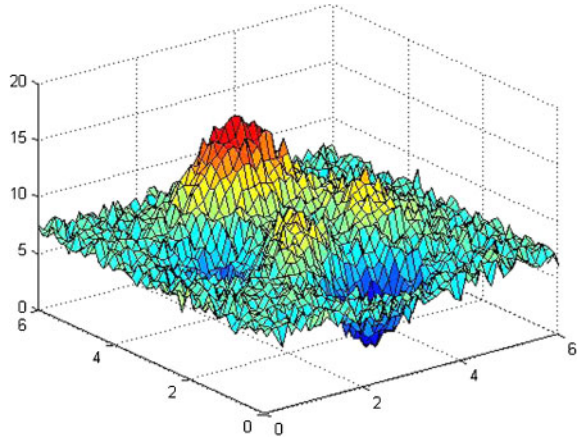

Fig. 3 Noisy algorithm development fitness landscape: peaks 2

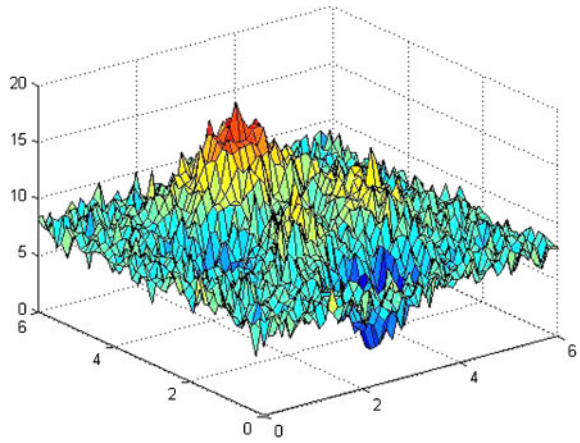

Fig. 4 Noisy algorithm development fitness landscape: peaks 3

For the search heuristics presented here, performance is degraded by the number of local optima in the search space. Local optima are formed by two mechanisms. The first mechanism is the underlying 'shape' of the search space. Basically, higher order functions tend to create more complex shapes with more local minima. Measurement or process noise adds numerous local minima to the underlying surface. Hence, in the paper, the noise added to the fundamental search space peaks 0 is defined by a statistical mean and variance in the text. As expected, performance degrades with increasing noise. An example of a custom-designed surface, namely 'bumps', is also considered. This surface is standard in the literature and presents complexity in terms of the number of local minima and the proportion of the area of local minima to global minima.

The magnitude of the noise is given as a fraction of the range of values of this input array. The addition of the noise is achieved by utilizing the R function

'jitter' written by Werner Stahel and Martin Maechler (Eidgenössische Technische Hochschule Zurich, Zurich, Switzerland).

The development surfaces are designated peaks 0 to peaks 3 with increasing levels of noise imposed on the clean peaks 0 surface according to the following:

(a) peaks 1: mean, 0.1189; variance, 0.0836;
(b) peaks 2: mean, 0.2842; variance, 0.3705;
(c) peaks 3: mean, 1.7277; variance, 0.7648.

This paper does not set out to compare the performance of search methodologies, but to investigate the benefits that can be obtained by integrating elements of decision support into the process. The parameters associated with each method were tuned in order to obtain a reasonably rapid convergence to an acceptable solution. Based upon the range of noise levels between peaks 0 and peaks 3 surfaces, a stop criterion of 0.5 was found to be an acceptable trade-off between performance and convergence time. This stop criterion was applied to all the heuristics. Step sizes and, where appropriate, cooling schedules were also tuned to give acceptable performance. This was particularly important, in order to give results with reasonable statistical significance; each method–surface combination was run 100 times to achieve the mean values. This approach was adopted because a significant feature of these heuristics is the stochastic nature of aspects of the search.

The GD method was run with a fixed step size of 0.2, the VNS method had an initial step size of 0.1, dividing by 2 at each step to a final step size of 0.01, and finally the SA method had a step size of 0.1, with a cooling schedule of 0.5 per step from an initial temperature of 10 to a final temperature of 0.001. The parameters for the search methods are given in Table 1.

In order to examine the method's effectiveness, another search space was introduced, namely the 'bump' problem [**15**, **16**], which is a smooth surface consisting of many peaks, all of similar sizes. Also, the optimal value is defined adjacent to a constraint boundary. It has been noted that these features render it relatively difficult for most optimizers to deal with.

The bump problem is defined as

$$\max\left\{\frac{\text{abs}\left[\sum_{i=1}^{n}\cos^4(x_i)-2\prod_{i=1}^{n}\cos^2(x_i)\right]}{\sqrt{\sum_{i=1}^{n} ix_1^2}}\right\} \qquad (24)$$

for

$$0<x_i<10, \quad i=1,\ldots,n$$

subject to

$$\prod_{i=1}^{n} x_i>0.75, \quad \sum_{i=1}^{n} x_i<\frac{15n}{2}$$

starting from

$$x_i=5, \quad i=1,\ldots,n$$

where x_i are the variables (in radians) in the range from 0 to 10, subject to two constraints, and n is the number of dimensions. The highly contoured surface that this function produces is shown in Fig. 5. In this case, a two-dimensional surface has been chosen. The global optimum is defined by the product constraint.

The surface of the function in two variables is shown in Fig. 6. The parameters of the heuristics

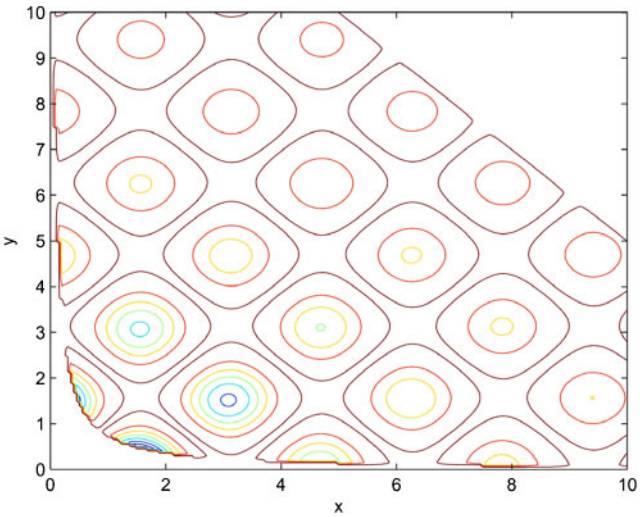

Fig. 5 Contour map for a two-variable bump function

Table 1 Search parameters: peaks

Method	Stop criterion	Step size	Schedule
GD	Minimum + 0.5	0.2	0
VNS	Minimum + 0.5	4 → 0.01	0.5
SA	Minimum + 0.5	0.1	10 → 0.001 (0.5)

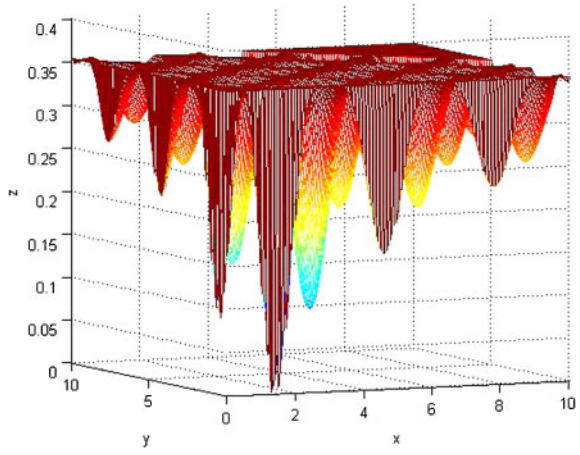

Fig. 6 Two-variable bump function surface

were tuned slightly to reflect this new surface and are given in Table 2.

2.3 Results

Each of the peaks surfaces was searched by each of the heuristics both with and without stochastic support. In addition, a variety of support surfaces of different orders were examined.

Each instance of surface type–with support–without support–order–cross-coupling was run 100 times producing mean results to negate the effects of the inherently stochastic methods. In the search methods presented here, each step is associated by a number of computations, evaluating the surrounding points (with positions specified by step size). In the comparisons of search performance, the total number of computations to the stop criterion are presented, together with the maximum number of steps that occurred in the worst-case search.

2.3.1 Gradient descent

The GD methodology was applied to the five surfaces initially without any stochastic support, resulting in the performance presented in Table 3.

In the case of the methodologies presented in this paper, a trade-off in performance was made during the initial tuning of the search parameters. In the case of GD, the main parameter is step size, which was adjusted to give an acceptable computation to convergence on the 'worst' surfaces, i.e. peaks 3 and

Table 2 Search parameters: bump

Method	Stop criterion	Step size	Schedule
GD	Minimum + 0.05	0.1	0
VNS	Minimum + 0.5	$4 \rightarrow 0.001$	0.5
SA	Minimum + 0.05	0.1	$10 \rightarrow 0.001 (0.5)$

Table 3 GD performance: no support

Surface	Mean computations	Worst-case computations
Peaks 0	327	565
Peaks 1	450	679
Peaks 2	1087	7614
Peaks 3	2750	16 698
Bump	6561	29 268

bump. This was done to make the problem computationally tractable as each run was carried out multiple times. The effect of this tuning is to give a relatively poor performance on the smooth peaks 0 surface which could have been improved considerably; however, as tuning and performance comparison between methods is not at the core of this paper, this pragmatic approach was deemed acceptable. As would be expected, mean and worst-case computations deteriorate as the level of complexity or noise increases, and the heuristic reaches more and more local minima above the level of the stop criterion.

A typical unsupported GD search on the peaks 2 surface is shown in Fig. 7 and illustrates a typical operation, with local terminations, stochastic jumps, and stop criterion at approximately $(3, 1)$.

The GD heuristic was then run on the same surfaces, with stochastic decision support ranging from first- to fifth-order surfaces. The results of the experiments are given in Table 4.

An example of a stochastic third-order support surface for the bump function is shown in Fig. 8, with the contour lines denoting the probability of the next jump. This particular support map is shown after three jumps. The structure and value of the

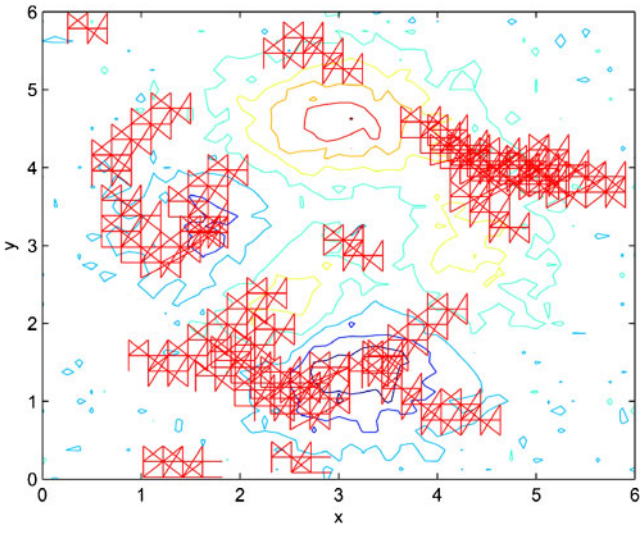

Fig. 7 Peaks 2: GD; no support; search example (computations, 953; steps, 101; stochastic jumps, 19)

Table 4 GD performance: with support (upper value, mean; lower value, worst case)

Surface	Value for the following				
	First order	Second order	Third order	Fourth order	Fifth order
Peaks 0	244	234	176	136	168
	1597	1361	646	599	633
Peaks 1	217	237	254	269	376
	985	1080	1278	1416	2298
Peaks 2	573	896	1009	669	1679
	6883	4947	10 397	5379	30 367
Peaks 3	1649	2435	2395	1204	2611
	6171	4287	32 402	6669	12 203
Bump	1668	1682	2014	2063	3051
	10 023	7237	6688	11 109	13 097

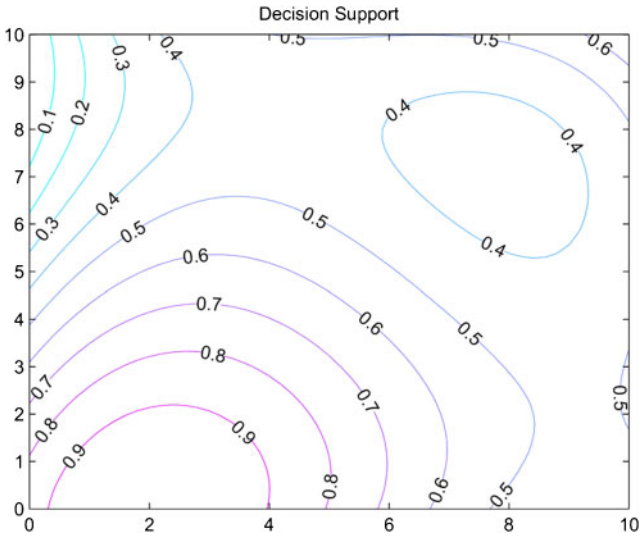

Fig. 8 Bump surface: GD; third-order stochastic support surface after three jumps

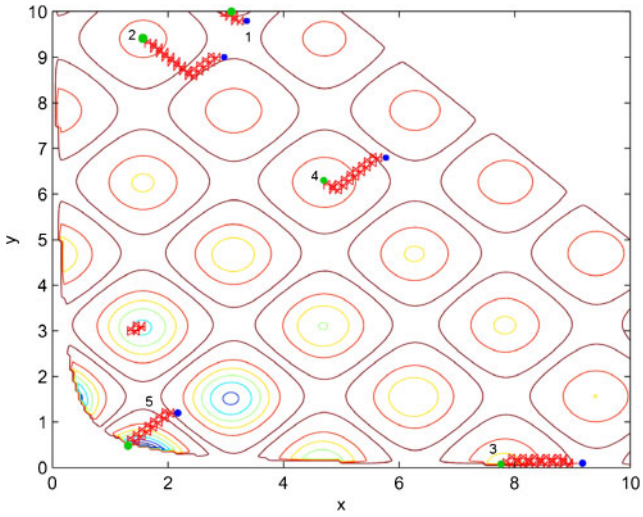

Fig. 9 Bump surface: GD; search example with third-order support (computations, 479; steps, 54; stochastic jumps, 6; start point, blue; finishing point, green). The search order is numbered

probability contours change on the basis of new information available after every search.

Figure 9 shows the corresponding bump function contour map with the searches that have been directed by the stochastic support operator. The stop criterion is satisfied after the final search reaches a point near (1.25, 0.5).

2.3.2 *Variable neighbourhood search*

The VNS was applied to the surfaces without any stochastic support, resulting in the performance presented in Table 5.

The heuristic was then run on the same surfaces, with stochastic decision support ranging from first- to fifth-order surfaces. The results of the experiments are given in Table 6.

2.3.3 *Simulated annealing*

Finally, SA was applied to the surfaces without any stochastic support, resulting in the performance presented in Table 7.

Table 5 VNS performance: no support

Surface	Mean computations	Worst-case computations
Peaks 0	723	2701
Peaks 1	936	2896
Peaks 2	1479	12 919
Peaks 3	3087	63 587
Bump	2502	98 784

The heuristic was then run on the same surfaces, with stochastic decision support ranging from first- to fifth-order surfaces. The results of the experiments are given in Table 8.

2.4 Discussion

Figures 10 and 11 present a comparison of the three search methodologies for the five surfaces. In all the figures presented in this section, the x axis represents increasingly noisy surfaces from peaks 0 to peaks 3, followed by the complex surface G2-inv. For both mean and worst-case computations, simple

Table 6 VNS: with support (upper value, mean; lower value, worst case)

Surface	Value for the following				
	First order	Second order	Third order	Fourth order	Fifth order
Peaks 0	660	334	507	453	454
	2201	1505	1793	1785	2683
Peaks 1	545	421	482	694	632
	3770	2226	1869	2681	2505
Peaks 2	730	880	974	699	788
	6883	4947	10 397	5379	30 367
Peaks 3	2448	1959	1913	1313	1487
	35 661	16 450	13 682	8965	11 941
Bump	4125	4358	4961	8550	6619
	18 903	15 572	18 423	54 847	39 062

Table 7 SA search performance: no support

Surface	Mean computations	Worst-case computations
Peaks 0	402	1937
Peaks 1	597	2386
Peaks 2	1542	10 184
Peaks 3	8134	40 868
Bump	9922	40 610

GD outperforms the other methodologies on the peaks surface. As predicted, SA performs increasingly poorly with increasing noise but shows a better performance than VNS on the complex G2_inv surface. No conclusions will be drawn concerning the relative methods used here, as the prime motivation of this research is to develop a generic decision support to increase the performance of generalized search methods.

Next, the performance of the search methods with stochastic decision support will be considered.

Figures 12 to 14 present a performance comparison for each method supported by a range of support surfaces from first to fifth order. The first observation of note is that the effectiveness of the decision support is dependent on its order, and hence its level of fit to the data surface being estimated. This will be investigated in future research, to incorporate an 'adaptive ordered' fit that optimizes the benefit of the support method. In the case of GD

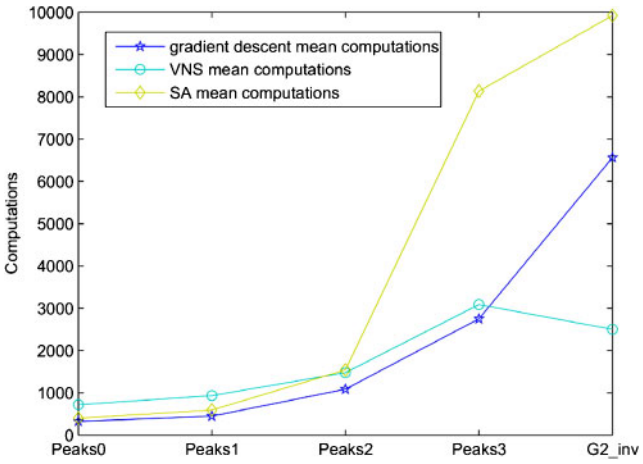

Fig. 10 Mean computations comparison of unsupported search methods on the five surfaces

(Fig. 12), excluding fifth-order support, the supported searches deliver increasing comparative levels of performance with increasing noise and complexity.

In the case of the VNS (Fig. 13), all the supported searches outperform unsupported searches in the presence of increasing noise; however, this does not hold true for the complex G2_inv surface. Finally, in the case of SA (Fig. 14), supported searches outperform unsupported searches only in the cases of high

Table 8 SA: with support (upper value, mean; lower value, worst case)

Surface	Value for the following				
	First order	Second order	Third order	Fourth order	Fifth order
Peaks 0	474	545	293	328	494
	2564	3669	1129	1667	2307
Peaks 1	644	822	853	716	644
	2614	4384	3981	4573	3002
Peaks 2	1515	2214	1530	2179	2747
	6151	21 859	12 312	22 382	49 213
Peaks 3	3838	4558	5014	3899	7804
	18 250	35 506	44 304	40 157	62 188
Bump	3042	5235	4522	4977	4834
	10 557	31 227	17 355	20 882	18 181

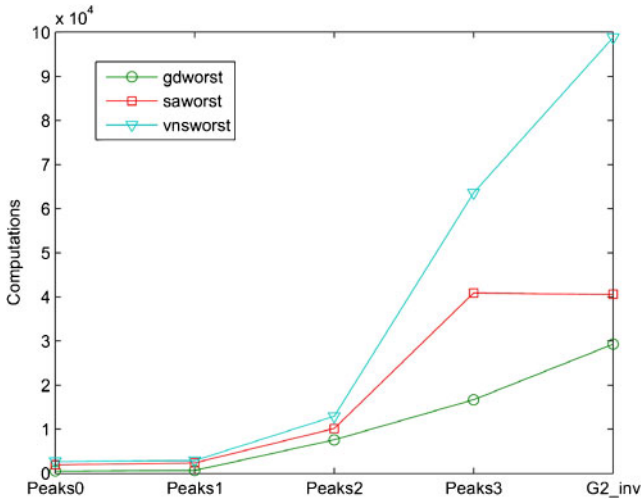

Fig. 11 Worst-case computations comparison of unsupported search methods on the five surfaces

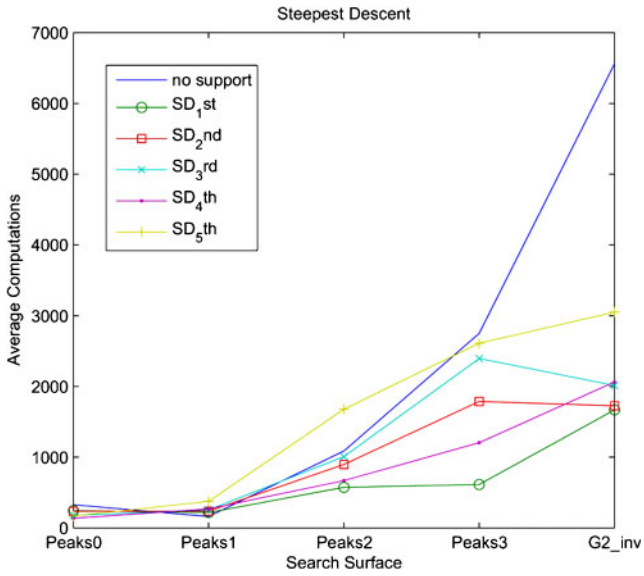

Fig. 12 Comparison of GD method with no support with first- to fifth-order supported searches

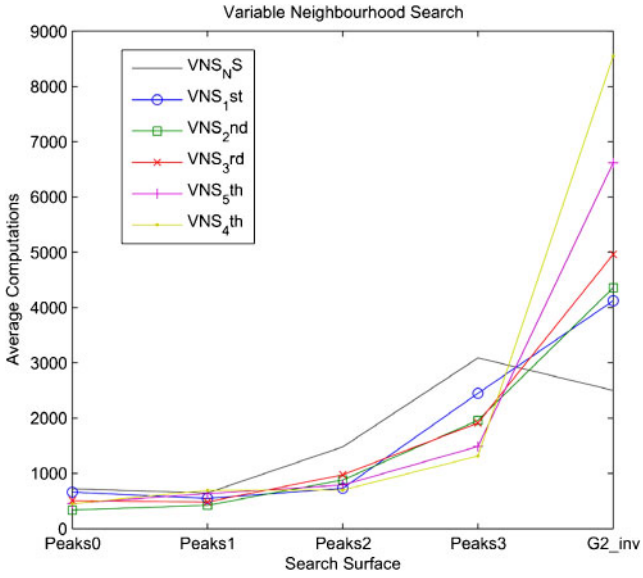

Fig. 13 Comparison of VNS method with no support with first- to fifth-order supported searches

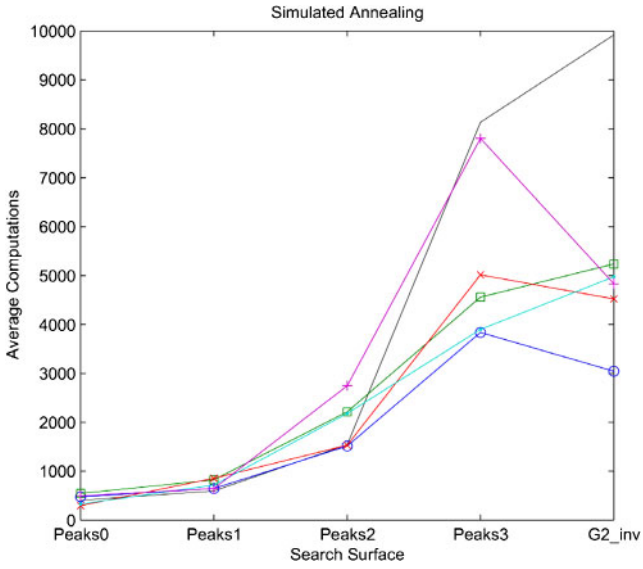

Fig. 14 Comparison of SA method with no support with first- to fifth-order supported searches

levels of noise or high levels of complexity. At this point in the development of the methodology, it can be concluded that the WSDS component makes a significant improvement to search efficiency, particularly with noisy real-world data. As with everything associated with search heuristics, there is no universal methodology, evidenced by the fact that unsupported SA outperforms the supported instances on the bump surface.

3 ENGINE-IN-THE-LOOP EXPERIMENTAL APPLICATION

In this section, a decision support system for hardware-in-the-loop optimization is investigated. The method, which has previously been developed on test surfaces in simulation, is applied to an automotive combinatorial search in the laboratory, on a real-time engine in the loop application. It is desired to find the maximum power output of an experimental single-cylinder spark ignition (SI) engine operating under a quasi-constant-volume (QCV) operating regime. Under this regime, the piston is slowed at top dead centre (TDC) to achieve combustion in close to constant-volume conditions.

As part of the further development of the engine to incorporate a linear generator to investigate free

piston operation, it is necessary to perform a series of experiments with combinatorial parameters. The objective is to identify the maximum power point in the least number of experiments in order to minimize costs. This test programme provides peak power data in order to achieve optimal electrical machine design.

The decision support methodology is combined with standard optimization and search methods, namely GD and SA, in order to study the reductions possible in experimental iterations. It is shown that the decision support methodology significantly reduces the number of experiments necessary to find the maximum power solution and thus offers a potentially significant cost saving to hardware-in-the-loop experimentation.

In this section, GD and SA are supplemented by the decision support methodology and applied to the real-life task of identifying the peak power operating point of a novel internal combustion engine experimental rig and control methodology. The supplemented methods are compared with the basic methodologies and offer considerable savings in experimental effort.

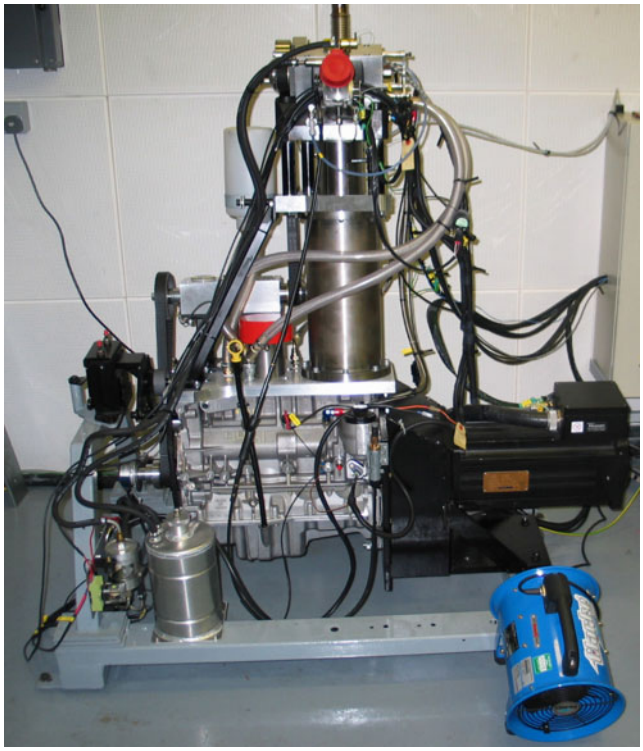

Fig. 15 The QCV experimental engine

3.1 Engine-in-the-loop operation

The engine employed in this research, as shown in Fig. 15, is a purposely converted single-cylinder research engine. Its combustion chamber, the head and the piston, are based on the GM Family One 1.8-l engine architecture, of four-valve type with a bore of 80.5 mm and a stroke of 88.2 mm. The bottom part of the engine is converted from a standard four-cylinder engine block. The combustion chamber has been lifted significantly from the bottom. The extension in between is reserved for further future work on free-piston engines. The purpose of the work described in this paper is to identify the peak power of the engine under a QCV regime [**17**], in order to design optimally a linear motor–generator that will replace the extension tube.

During operation of conventional internal combustion (IC) engines, the piston can only reciprocate continuously between TDC and bottom dead centre (BDC) at a frequency proportional to the engine speed. The chemical reaction process associated with combustion events, however, essentially takes a fixed time to complete, which is relatively independent of the engine speed. In order to maximize the work obtained from the heat energy released by combustion, the air–fuel mixture has to be ignited prior to the piston reaching TDC, and the ignition timing should be adjusted according to the engine speed and the quality of the air–fuel mixture.

Clearly, the early stage of the heat release before the piston reaches TDC results in negative work. During the combustion event, the piston movement is defined by the crank rotation, so that truly constant-volume heat release is not achievable. Further, to scavenge the burned gas efficiently, the exhaust valve has to be opened well before BDC, while the pressure of the burned gas is still high. Thus, a large portion of the thermal energy is expelled into the exhaust, which further reduces the engine efficiency. The ideal scenario is to initiate and complete the combustion event while the piston remains at or close to the TDC position in order to achieve the maximum thermal potential and eliminate the negative work which results with early ignition, and to extend the expansion stroke further in order to use the thermal energy fully as well as to provide sufficient time for post-combustion reactions, thereby reducing partial burned emissions. One practical method of achieving such an optimization without changing the engine design and sacrificing engine performance in series-hybrid applications is to reduce the engine crank rotation velocity significantly at the TDC position to provide sufficient time for combustion to be completed and then to accelerate the engine during the compression and

expansion phases to maintain the high average crank speed to deliver the high power output. This will then generate a new combustion cycle which is between the combustion cycle of a conventional IC engine and the ideal Otto constant-volume combustion cycle. This can be therefore be called a QCV combustion cycle, as illustrated in Fig. 16.

Theoretically, the series-hybrid power train enables a higher efficiency and power output from IC engine configurations [**18**], while the QCV concept offers an even greater potential for higher combustion efficiency. In order to investigate the QCV combustion concept, a proof-of-principle engine system has been developed, as shown in Fig. 17. The system consists of a high torque-to-inertia ratio, high-bandwidth permanent-magnet brushless a.c. electric machine, a single-cylinder research SI engine, and a control system.

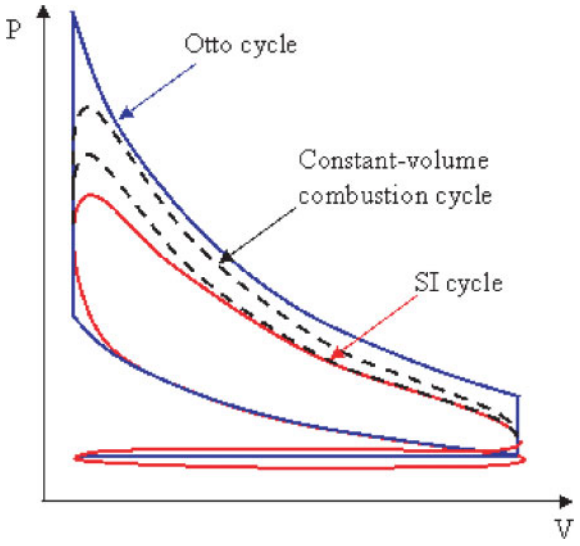

Fig. 16 Typical pressure–volume diagram

3.1.1 *Piston trajectory*

For a proof-of-principle system, control was implemented on the electrical machine that could deliver a sinusoidal crank velocity with defined average and magnitude quantities. An example of a simple variable-crank-velocity profile was selected; a sinusoidal wave velocity form at an average speed of 600 r/min with a wave magnitude of $\pm$200 r/min has been employed in the study. Figure 18 shows the theoretical variable crank rotation velocity at various crank positions in comparison with equivalent conventional constant-speed data. The piston TDC position is 0° and 360°.

The residual time at TDC has been extended, while the residual time at BDC is reduced, as shown by the solid curve. This offers longer time for the combustion event to complete at the TDC region which delivers higher combustion efficiency than in the conventional case. Figure 19 shows the fired cylinder pressure of the conventional cycle at 600 r/min and the QCV cycle at a sinusoidal speed of 600 r/min average with $\pm$200 r/min amplitude at a normalized cycle time. The engine was 5.75 per cent throttled in the conventional cycle and 5.3 per cent for the QCV cycle. The fuel-injection pulse width for both scenarios were the same with a length of 5.65 ms. The SI timing of the conventional cycle was optimized at 10.2° crank angle (CA) before top dead centre (BTDC). For the QCV cycle, it was optimized at 9.8 °CA BTDC. Clearly, the QCV cycle uses a later optimized ignition timing, but produces a later but higher peak cylinder pressure, which further leads to an overall higher in-cylinder pressure during the expansion stroke.

The work produced by a combustion engine is an integration of the pressure over an engine cycle. Clearly, the higher expansion pressure of the QCV cycle can produce higher work than its conventional

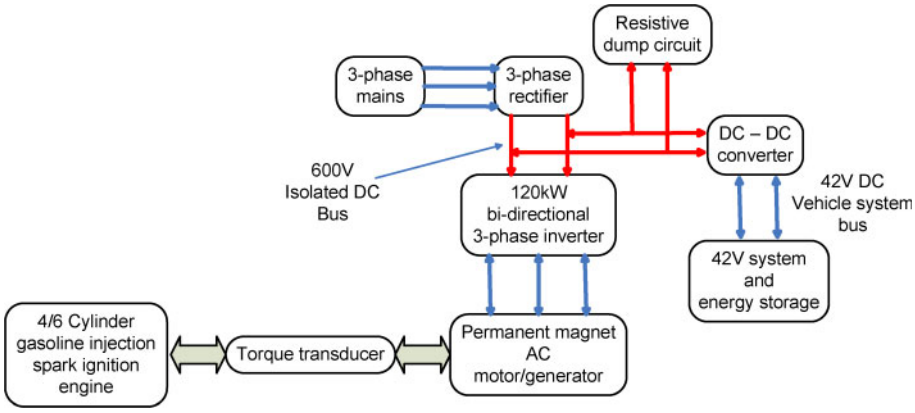

Fig. 17 Schematic diagram of the QCV electrical power system

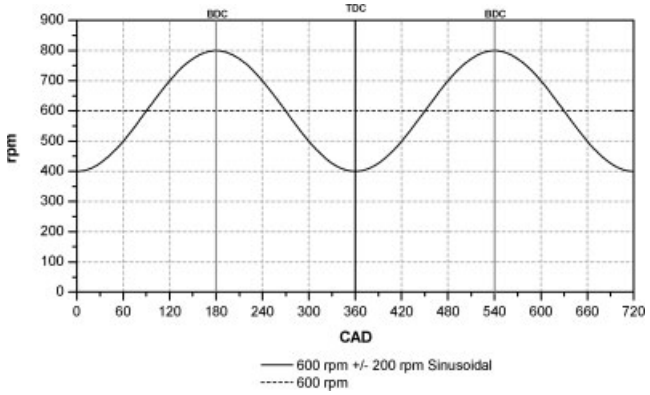

Fig. 18 Conventional constant and variable sinusoidal crank velocity (CAD, crank angle (deg))

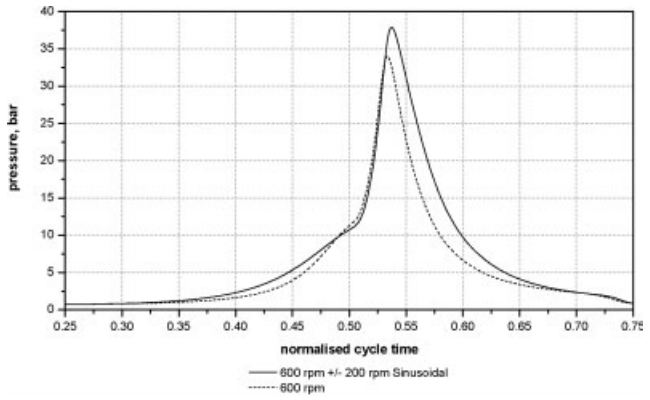

Fig. 19 Peak cylinder pressure at firing; QCV versus conventional operation

counterpart. Overall, the pressure integral of the QCV cycle is 11 per cent higher than that of the conventional cycle.

3.2 Decision support and search methods

For the next stage of the development of this experimental rig, and the main exposition of the application of the decision support methodology, it is necessary to find the combinations of mean crankshaft velocity and sinusoidal amplitude at each throttle setting and injection timing that deliver the highest peak cylinder pressure (and hence power). These data are required for the design of a linear generator that will be built into the rig to investigate free-piston operation. As an example, the application of the experimental decision support operator will be examined to find the maximum power operating point of the engine at the throttle and injection settings given earlier in the section. In order to evaluate the relative performance of the decision support method, the engine was characterized by an exhaustive search, which is shown in Fig. 20.

GD-based methods will be used to find the combination of mean crankshaft velocity versus crankshaft sinusoidal perturbation that produces the highest peak cylinder pressure.

Figure 21 shows a typical hardware-in-the-loop search using a simple GD method. The method uses a total of 41 stochastic jumps, and a total of 311 steps to find a solution within 0.02 bar of the maximum identified by exhaustive search. Under the same experimental parameters, a simple GD search with WSDS is performed.

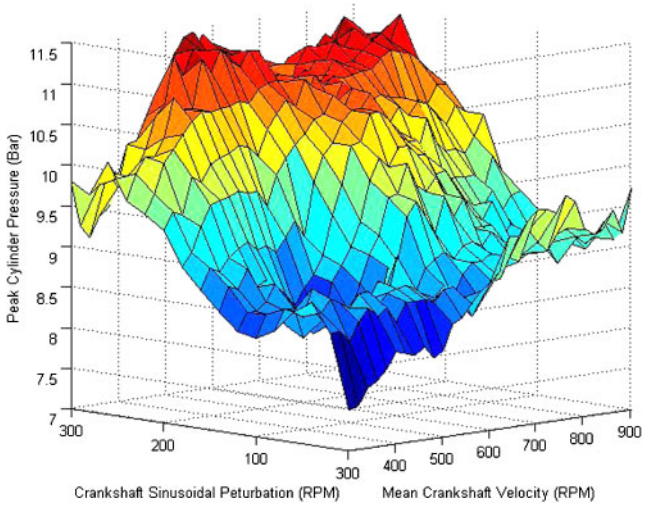

Fig. 20 Engine experimental map for peak cylinder pressure under QCV operation

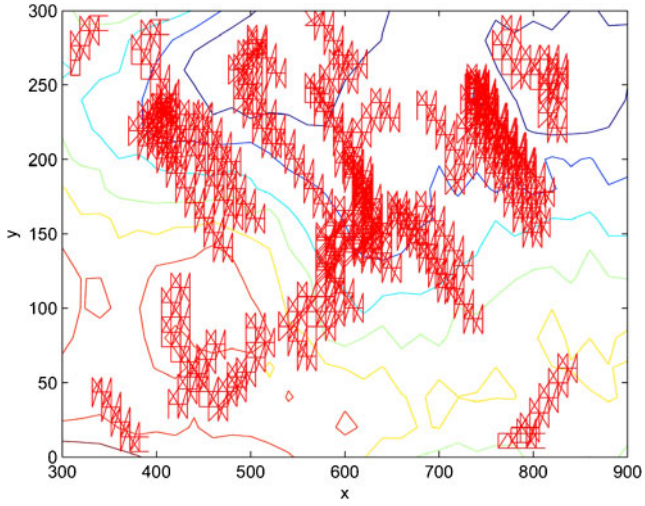

Fig. 21 Example of a simple GD search with no WSDS (terminating coordinates (516, 280); *x*, average crankshaft speed (r/min); *y*, sinusoidal perturbation (r/min))

In Fig. 22, the decision support surface based upon previous searches is shown. This particular surface is stopped at fulfilment of the termination criterion. The associated searches are shown in Fig. 23.

In this case, the termination criterion was achieved in 24 jumps and 126 steps. Apart from the improvement in search efficiency, the effect of the weighted surface on the stochastic jumps can be observed. Search activity increases around areas of higher interest based upon previous searches.

3.3 Discussion

For simplicity of analysis here, the decision support RSs applied to the experimental data were fixed second order, although further research will investigate the efficacy of adaptive schemes. GD and SA were run both with and without support. Since the terminal value had been identified by exhaustive search, the relaxation of termination value was also varied to investigate its influence (Fig. 24).

Irrespective of the search algorithm, WSDS support reduces the mean number of searches by a significant amount and, in the case of the tightest stop criterion, by approximately 75 per cent.

It is important to remember that this is not a comparative study of search heuristics, but an observation of the advantages that can be gained by extending simple search heuristics with a simple RS-based decision support operator. It can be seen that, in all cases, the WSDS extension significantly improves the performance of the search methodologies. The improvement in performance of all the searches as the stop criterion is relaxed reflects one of the basic effects of experimental noise. As noise or the 'tightness' of the stop criterion to the global minimum increases, then these methodologies become increasingly sensitive to the step size and starting point. In other words, the noise dominates the search.

As can be seen from the typical support surface in Fig. 22, there is an overall correlation between the 'area of interest' and the actual experimental surface; however, this would be improved by higher-order RSs (section 2). Future work will include the development of an adaptive WSDS, which will choose the best RS method representation based on the fit to the received data. Of future interest will also be the integration of certain aspects of tabu search [**8**]. After a certain number of searches, it might be advanta-

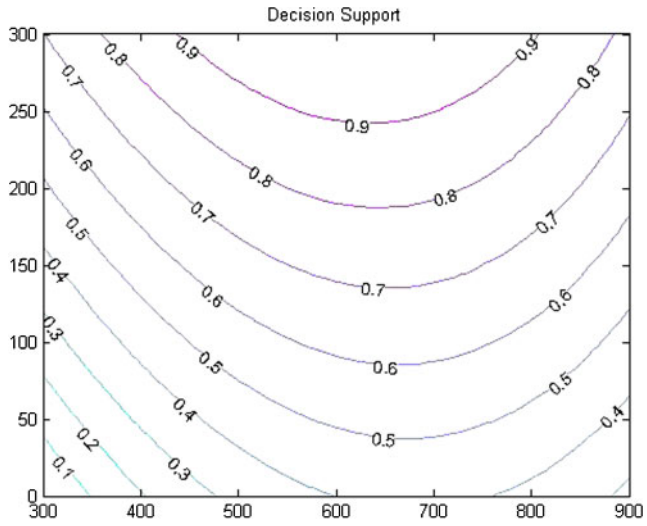

Fig. 22 WSDS support for experimental search

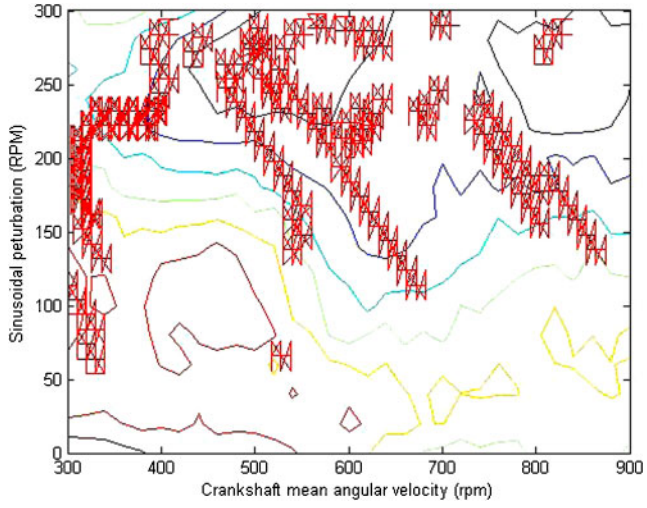

Fig. 23 Example of a simple GD search with WSDS (terminating coordinates (515, 279))

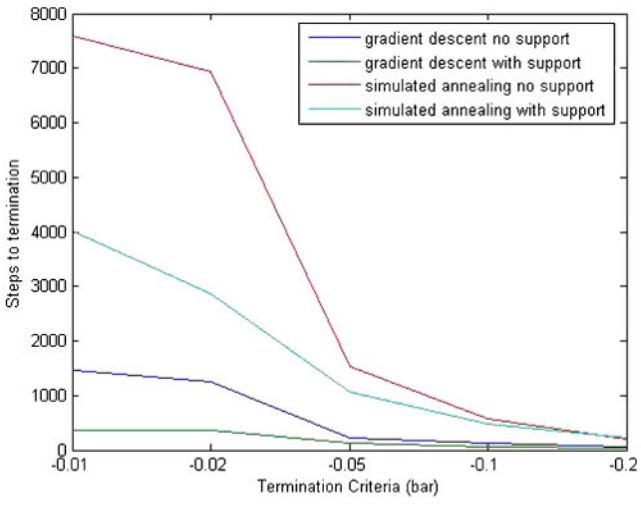

Fig. 24 Mean experimental steps to termination versus relaxation of stop criterion

geous to weaken areas with low confidence, and to strengthen the areas with high confidence. This would have the effect of higher concentration of jumps away from low-confidence areas, a corollary with tabu search. This will also be researched at a later date.

4 CONCLUSION

A method has been presented to add decision support to the previously random jumps of GD methods commonly used in combinatorial experimentation. Significant promise has been shown in the performance improvements of even the simplest GD method. However, as with all search methodologies, there is no universal solution. The basic heuristics themselves have been shown to perform better on some surfaces than others, which is to be expected. In a similar way, the support method has been shown to be more effective in certain surface–method–support order combinations.

In section 3 of this paper, the support methodologies were applied to a real-life combinatorial experiment conducted in an engine laboratory, to assess their performance in a realistic environment. Significant improvements in search efficiency was exhibited by the WSDS methodology, even under a relatively tight stop criterion.

Future work will address implementing an adaptive order support surface, and an adaptive search hybrid (effectively a hyperheuristic) to maximize the effectiveness of this method. In particular, an adaptive heuristic search parameter will be investigated.

A decision support methodology has been presented to support hardware-in-the-loop experimentation. Since the majority of this type of experimentation is performed on plant with an unknown response, it is general practice to utilize simple heuristics such as GD or SA. Genetic algorithms have also been shown to be effective, and these will be the subject of future study in terms of decision support. The methodology as presented in this paper has shown itself to be effective in dealing with 'unknown' search spaces and relatively 'untuned' search heuristics.

A decision support operator has been presented, which uses the past history of searches to construct a 'confidence map' based upon the RS methodology. This confidence map influences the stochastic jumps of the heuristics towards areas of increased interest. In line with the philosophy of the heuristics, the decision support operator is extremely simple to implement.

In section 2, the performance of the decision support operator was investigated on a series of test surfaces with various levels of noise present. The operator was shown to be highly effective in reducing the number of steps to termination of the chosen heuristics.

In section 3, the methodology has been applied to the task of finding the maximum power point of a single-cylinder engine under a novel operating regime at a defined operating point. The two chosen heuristics were GD and SA. In both cases, the search performance was significantly improved by decision support. With respect to the experimental application under consideration in this paper, the identified peak pressures with predicted peak power profiles allowed the design of a linear motor–generator for the project, under reduced experimental time to extract the necessary data and, as such, the methodology is shown to be advantageous.

ACKNOWLEDGEMENTS

Technical support and the hardware used in this research was provided by the Engineering and Physical Sciences Research Council Project Zero-Constraint Free Piston Energy Converter (Grant GR/S97507/01) and Lotus Engineering Ltd UK.

REFERENCES

1 **Stewart, P., Fleming, P. J.,** and **MacKenzie, S. A.** Real time simulation and control systems design by the response surface methodology and designed experiments. *Int. J. Systems Sci.*, 2003, **34**(14–15), 837–850.
2 **Stewart, P., Stone, D. A.,** and **Fleming, P. J.** Design of robust fuzzy-logic control systems by multi-objective evolutionary methods with hardware in the loop. *IFAC J. Engng Applic. Artif. Intell.*, 2004, **70**(3), 275–284.
3 **Myers, R. H.** and **Montgomery, D. C.** *Response surface methodology: process and product optimization using designed experiments*, 1995 (John Wiley, New York).
4 **Michalewicz, Z.** and **Fogel, D. B.** *How to solve it, algorithms for engineering systems*, 2006 (Cambridge University Press, Cambridge).
5 **Narayana Naik, G., Gopalakrishnan, S.,** and **Ganguli, R.** Design optimization of composites using genetic algorithms and failure mechanism based failure criterion. *Composite Structs*, 2008, **83**(4), 354–367.

6 **Mladenovic, M.** and **Hansen, P.** Variable neighbourhood search. *J. Comput. Ops Res.*, 1997, **24**(11), 1097–1100.

7 **Burke, E.** and **Kendall, G.** Comparison of metaheuristic algorithms for clustering rectangles. *J. Comput. Ind. Engng*, 1990, **37**(1), 383–386.

8 **Glover, F.** and **Hanfi, S.** Tabu search and finite convergence. *Appl. Math.*, 2002, **119**(1–2), 3–36.

9 **Toksari, M. D.** and **Guner, E.** Solving the unconstrained optimisation problem by a variable neighbourhood search. *J. Math. Analysis and Applic.*, 2007, **328**(2), 1178–1187.

10 **Baldick, R.** *Applied optimization: formulation and algorithms for engineering systems*, 2006 (Cambridge University Press, Cambridge, UK).

11 **Kirkpatrick, S., Gelatt, C. D.,** and **Vecchi, M. P.** Optimization by simulated annealing. *Science*, 1983, **220**, 671–680.

12 **Burke, E.** and **Kendall, G.** *Search methodologies: introductory tutorials in optimisation and decision support techniques*, 2005 (Springer, New York).

13 **Kirkpatrick, S., Gelatt, C. D.,** and **Vecchi, M. P.** Optimization by simulated annealing. *Science, New Ser.*, 1983, **220**(4598), 671–680.

14 **Stewart, P.** and **Fleming, P. J.** The response surface methodology for rapid prototyping of real-time control systems. In Proceedings of the 2002 American Control Conference, Anchorage, Alaska, USA, 8–10 May 2002, pp. 3343–3348 (IEEE, New York).

15 **Keane, A. J.** Experiences with optimizers in structural design. In Proceedings of the First International Conference on *Adaptive computing in engineering design and control* (Ed. I. C. Parmee), Plymouth, Devon, UK, 21–22 September 1994, pp. 14–27 (University of Plymouth, Plymouth, UK).

16 **Keane, A. J.** Genetic algorithm optimization of multi-peak problems: studies in convergence and robustness. *Artif. Intell. Engng*, 1995, **9**(2), 75–83.

17 **Chen, R.** Quasi-constant volume (QCV) spark ignition combustion. SAE paper 2009-01-0700, 2009.

18 **Stone, R.** *Introduction to internal combustion engines,* 3rd edition, 1999 (Macmillan, London).