J Glob Optim (2010) 46:273–286
DOI 10.1007/s10898-009-9425-z

Stopping and restarting strategy for stochastic sequential search in global optimization

Zelda B. Zabinsky · David Bulger · Charoenchai Khompatraporn

Received: 26 June 2008 / Accepted: 21 April 2009 / Published online: 8 May 2009

Abstract Two common questions when one uses a stochastic global optimization algorithm, e.g., simulated annealing, are when to stop a single run of the algorithm, and whether to restart with a new run or terminate the entire algorithm. In this paper, we develop a stopping and restarting strategy that considers tradeoffs between the computational effort and the probability of obtaining the global optimum. The analysis is based on a stochastic process called Hesitant Adaptive Search with Power-Law Improvement Distribution (HASPLID). HASPLID models the behavior of stochastic optimization algorithms, and motivates an implementable framework, Dynamic Multistart Sequential Search (DMSS). We demonstrate here the practicality of DMSS by using it to govern the application of a simple local search heuristic on three test problems from the global optimization literature.

Keywords Stopping criteria · Sequential search · Pure adaptive search

1 Introduction

Simulated annealing (SA) and other stochastic global optimization algorithms are being applied to a wide variety of problems. Yet a common question is when to terminate the algorithms. While SA has been shown to converge to global optima in probability [8,11,14,15], in

Z. B. Zabinsky
Industrial Engineering, University of Washington, Seattle, WA 98195-2650, USA
e-mail: zelda@u.washington.edu

D. Bulger (✉)
Department of Statistics, Macquarie University, Sydney, NSW 2109, Australia
e-mail: dbulger@efs.mq.edu.au

C. Khompatraporn
Department of Production Engineering, King Mongkut's University of Technology Thonburi,
126 Pracha-utit Rd., Thungkru, Bangkok 10140, Thailand
e-mail: ckhomp@gmail.com

practice the algorithm often appears to get "trapped" at a suboptimal point, i.e., the algorithm cannot find a better solution after a large number of iterations.

A common practice when a stochastic optimization algorithm is trapped is to restart the algorithm with a different starting point (often drawn from a uniform distribution). Various studies show that stochastic optimization algorithms seem to benefit from restarts. For instance, Theodosopoulos [19] performed a study on algorithmic restart. He concluded that a nonzero level of randomness enhanced performance robustness of stochastic global optimization algorithms in an unknown "rugged" objective function terrain. Glidewell et al. [10] applied SA to a medical problem and reported that restarting increased the likelihood of converging to a global optimum. Treadgold and Gedeon [18] combined SA with a gradient descent method and found that restarting enhanced the overall performance of the algorithm. Likewise, Li and Lim [13] embedded SA in a Tabu search with restarts to solve a set of pickup and delivery problems. Their algorithm outperformed other algorithms for that particular problem set. Atkinson [3] designed an experiment to investigate numerically the effect of multiple short runs of SA. He found that the algorithm's performance was improved when a single run was stopped and restarted at a fifth to a quarter of the total available number of function evaluations for potentially long searches. These studies are, however, empirical in nature.

Some theoretical results on stopping criteria for multi-start algorithms are summarized in Boender and Romeijn [6]. Stopping strategies based on the probability of obtaining the global optimum are discussed in [4,5,16,25]. Zielinski [25] developed a stopping rule for a multinomial distribution whose cells correspond to the local optima of the objective function, and this was improved upon by Boender and Rinnooy Kan [5], who used the generalized multinomial distribution and computed the posterior distribution of the number of local optima based on estimates of regions of attraction. These methods use clustering methods to decide whether to initiate a local search from a possible starting point. The local search methods are assumed to find a local optimum in the region of attraction. However, it is difficult to estimate the extent of regions of attraction without further information about the objective function. These methods also assume the cost of each restart to be constant, whereas run lengths differ in practice. Therefore, there remains a need for a theoretical development that considers computational tradeoffs for sequentially dependent search with restarts, such as simulated annealing with restarts.

From a statistical point of view, an algorithm should be terminated once the estimated global solution has a high probability of being close to the global optimum. From a computational point of view, the algorithm should be terminated when the gain from further execution of the algorithm is not worth the computational effort of continuing. In the context of a multi-start algorithm, we want to stop a single run when it appears to be trapped in a local optimum and little or no improvement has been detected in a reasonable amount of time. The decision whether to restart another run or terminate the complete algorithm should consider the overall performance, and the chance that the local optimum is the global optimum. Our approach to developing a stopping and restarting criterion considers tradeoffs between the computational effort and the probability of obtaining a solution close to the global optimum. An initial stopping and restarting criterion was developed using Improving Hit-and-Run [12] and numerical success was demonstrated on fractional programs [9].

In this paper, we develop a theoretical approach to identify the length of a single run and the number of restarts of a stochastic algorithm for global optimization to achieve a high probability of getting close to the optimal solution while keeping computational effort minimal. In order to estimate the probability of getting close to the optimum, we model the actual algorithm with a stochastic process that we term Hesitant Adaptive Search with

Power-Law Improvement Distribution (HASPLID), which generalizes Pure Adaptive Search (PAS) [22,23] and is an instance of Hesitant Adaptive Search [7,21,22]. The form of HASPLID is chosen because it is tractable to analyze and captures both the probability of improvement and the amount of improvement in two parameters that can be estimated. Even though the modeling takes no account of the specific surface of the objective function f, the two parameters representing the probability as well as amount of improvement observed by the algorithm behavior give an implicit account of the interaction between the heuristic algorithm and the objective function.

Our performance analysis of HASPLID is on a class of Lipschitz optimization problems, and we parameterize the expected computational effort and probability of success through the length of each single run and the number of restarts. We use the output from the stochastic algorithm to dynamically estimate the parameters of HASPLID, enabling us to model the behavior of the algorithm with HASPLID. The performance analysis leads to a stopping and restarting strategy for stochastic sequential optimization algorithms.

To apply this theoretical analysis, we assume that we already have an optimization heuristic, and seek stopping and restarting criteria to govern its application. We model the record values (improving points) of the heuristic as records of HASPLID or, equivalently, as iterates of PAS. In fact, if each of the candidate points of an algorithm is truly drawn from a Boltzmann distribution, then this sampling will theoretically outperform PAS [17], and the results in this paper still hold. We develop Dynamic Multistart Sequential Search (DMSS), which is easy to implement, based on the theoretical analysis of HASPLID. We present numerical results for DMSS with a simple elitist random walk as the stochastic sequential search on a set of test problems. A numerical comparison with the same optimization heuristic without the DMSS framework demonstrates the effectiveness of our stopping and restarting strategy.

The remainder of the paper is organized as follows. Notation and assumptions surrounding the optimization problem are specified in Sect. 2. Sections 3 and 4 describe the idealized HASPLID model of the heuristic's range behavior, that is, of the random sequence of objective function values sampled by the heuristic. Section 5 presents a framework for using the idealized model to gauge the heuristic's performance and to decide when to restart and when to terminate. Section 6 reports promising numerical results of the DMSS framework on test functions in various dimensions.

2 Optimization problem

The optimization problem considered here is to minimize

$$f : S \to \mathbb{R},$$

where $(S, \mathcal{S}, \mu)$ is a probability space (typically a manifold). We make frequent use of the *range distribution*

$$\rho(T) = \mu(f^{-1}(T))$$

for $T \subseteq \mathbb{R}$ and its cumulative distribution function (CDF)

$$p(y) = \mu(f^{-1}((-\infty, y])).$$

Throughout, we assume the range distribution to be continuous, that is, we assume that

$$\rho(\{y\}) = \mu(\{x \,:\, f(x) = y\}) = 0$$

for all y. This is frequently true for continuous optimization problems and, more widely, may be a reasonable approximation.

Of course, this means that the set of globally optimal points has measure zero. For this reason, we do not expect to be able to sample from the global optima exactly. Rather, we will nominate a proportion $\epsilon \in (0, 1)$ and aim to sample from the optimal ϵ-quantile of the domain, that is, the *target region* will be

$$\{x : p(f(x)) \leq \epsilon\} = \{x : \mu(\{x' : f(x') \leq f(x)\}) \leq \epsilon\}.$$

3 Hesitant adaptive search with power-law improvement distribution

In this section, we develop a parametrized model of the range behavior of an optimization algorithm, called *Hesitant Adaptive Search with Power-Law Improvement Distribution* (HASPLID), to be used in Sect. 5 in formulating criteria for stopping a single run and determining whether to restart another run or terminate the whole algorithm. The model depends on the range distribution ρ and on two real parameters: the improvement probability parameter $\alpha \in [0, 1]$, controlling the difficulty of finding improvements, and the improvement quality parameter $\lambda \in \mathbb{R}$, controlling the distribution of the quality of improvements found.

The parameter λ is used to define a power-law transformation $\rho^{(\lambda)}$ of the range distribution ρ, given by

$$\rho^{(\lambda)}(T) = \int_{z \in T} \mathrm{d}(p(z))^{\lambda} = \lambda \int_{z \in T} (p(z))^{\lambda - 1} \, \mathrm{d}p(z).$$

The relationship between the CDF $p^{(\lambda)}$ of $\rho^{(\lambda)}$ and the CDF p of ρ is

$$p^{(\lambda)}(y) = (p(y))^{\lambda}.$$

The normalized restriction of $\rho^{(\lambda)}$ to the left half-line $(-\infty, y]$ (that is, the distribution of the objective function value Y, conditioned on $Y \leq y$) is given by

$$\rho_y^{(\lambda)}(T) = \lambda (p(y))^{-\lambda} \int_{z \in T \cap (-\infty, y]} (p(z))^{\lambda - 1} \, \mathrm{d}p(z),$$

with CDF

$$p_{y_1}^{(\lambda)}(y_2) = \left(\frac{p(y_2)}{p(y_1)} \right)^{\lambda}$$

for $y_2 < y_1$.

For convenience and clarity, we specify the parameterized stochastic process Hesitant Adaptive Search with Power-Law Improvement Distribution, HASPLID($\alpha, \lambda; \rho$), using the pseudocode below.

HASPLID($\alpha, \lambda; \rho$)
  sample Y_0 according to $\rho^{(\lambda)}$
  for $k = 1, 2, \ldots$
    with probability $(p(Y_{k-1}))^{\alpha}$
      sample Y_k according to $\rho_{Y_{k-1}}^{(\lambda)}$
    otherwise
      set $Y_k = Y_{k-1}$

4 Behavior of HASPLID

This section develops the theory and performance analysis of HASPLID. Our first three propositions regard the quality of the records produced by the HASPLID model. The last three propositions describe the behavior of HASPLID in terms of a ratio of the parameters α and λ. For a multi-start random search algorithm, e.g., simulated annealing, we estimate the parameters α and λ from observed record values within a single run. If non-improving points within a run (of simulated annealing for example) are accepted, HASPLID models this as hesitation. The final equations in this section are used in the DMSS framework to dynamically estimate the parameters. Then we use HASPLID with the estimated parameters to determine the probability of achieving the target region for the specified number of runs and associated lengths. If this probability is sufficiently small, we terminate the algorithm, otherwise we restart another run.

Proposition 1 *The iterates* (Y_k) *of HASPLID*$(\alpha, \lambda; \rho)$ *are stochastically equivalent to the iterates* $(\tilde{Y}_k)$ *of HASPLID*$(\alpha/\lambda, 1; \rho^{(\lambda)})$.

Proof Both sequences are Markov chains, so it suffices to equate their initial distributions and transition kernels. The CDFs of Y_0 and $\tilde{Y}_0$ are both given by $p^{(\lambda)}(y) = (p(y))^{\lambda}$. Also, for $k \in \mathbb{N}$,

$$\begin{aligned} P[\tilde{Y}_k \le w | \tilde{Y}_{k-1} = y] &= P[\tilde{Y}_k \le w | \tilde{Y}_{k-1} = y \text{ and } \tilde{Y}_k < \tilde{Y}_{k-1}] P[\tilde{Y}_k < \tilde{Y}_{k-1} | \tilde{Y}_{k-1} = y] \\ &= (p^{(\lambda)})_y^{(1)}(w) \times (p^{(\lambda)}(y))^{\alpha/\lambda} \\ &= (p^{(\lambda)}(w)/p^{(\lambda)}(y)) \times (p(y))^{\alpha} \\ &= p_y^{(\lambda)}(w) \times (p(y))^{\alpha} \\ &= P[Y_k \le w | Y_{k-1} = y \text{ and } Y_k < Y_{k-1}] P[Y_k < Y_{k-1} | Y_{k-1} = y] \\ &= P[Y_k \le w | Y_{k-1} = y] \end{aligned}$$

whenever $w < y$ (and both probabilities clearly equal 1 when $w \ge y$). □

Proposition 2 *Let* $N(y)$ *denote the number of records obtained by HASPLID* $(\alpha, \lambda; \rho)$ *before obtaining a value of* y *or better. Then* $N(y)$ *is Poisson distributed with mean* $-\lambda \ln p(y)$.

Proof By Proposition 1, $N(y)$ is equivalent to the number of records obtained by HASPLID $(\alpha/\lambda, 1; \rho^{(\lambda)})$ before obtaining a value of y or better. However, HASPLID$(\alpha/\lambda, 1; \rho^{(\lambda)})$ is an instance of hesitant adaptive search (HAS) [7,21,22] on the range distribution $\rho^{(\lambda)}$. The records of HAS are stochastically equivalent to the records of PAS [21, Lemma 1] and, in particular, the number of records exceeding y is Poisson distributed, with mean equal to

$$-\ln(\rho^{(\lambda)}((-\infty, y])) = -\ln((p(y))^{\lambda}) = -\lambda \ln p(y).$$

□

Proposition 2 provides the distribution characterizing the number of HASPLID records, which depends on λ and ρ. The parameter α describes the hesitation of HASPLID and so is not needed in Proposition 2. Proposition 3 further characterizes HASPLID with an expression for the probability that the jth record value has a value greater than y.

Let $Y_{(j)}$ denote the jth record value of HASPLID$(\alpha, \lambda; \rho)$.

Proposition 3 *The probability that $Y_{(j)} > y$ is given by*

$$P[Y_{(j)} > y] = 1 - (p(y))^{\lambda} \sum_{s=0}^{j-1} (-\lambda \ln p(y))^s / s!$$

or adapting the notation for an incomplete gamma function from [1, 6.5.1, 6.5.13]*, we have* $P[Y_{(j)} > y] = G(j, -\lambda \ln p(y))$ *where* $G(n, x) = 1 - e^{-x} \sum_{s=0}^{n-1} x^s / s!$.

Proof Since the sequence of records is decreasing, the jth record $Y_{(j)} > y$ if and only if $N(y)$, the number of records exceeding y, is j or greater, i.e., $P[Y_{(j)} > y] = P[N(y) \geq j]$. By Proposition 2, $N(y)$ is Poisson distributed with mean $-\lambda \ln p(y)$, hence

$$P[N(y) \leq j - 1] = (p(y))^{\lambda} \sum_{s=0}^{j-1} (-\lambda \ln p(y))^s / s!$$

and thus $P[N(y) \geq j] = 1 - (p(y))^{\lambda} \sum_{s=0}^{j-1} (-\lambda \ln p(y))^s / s!$ which by [1, 6.5.13] equals $G(j, -\lambda \ln p(y))$. □

The remaining results in this section relate to parameter estimation for the HASPLID model, and using the probability expressions to determine a termination criterion.

For positive integers k and j with $j \leq k$ and real y, let $U_{j,k}(y)$ denote the event that there are exactly j records in the first k iterates $Y_0, Y_1, \ldots, Y_{k-1}$ of HASPLID($\alpha, \lambda; \rho$) with the final record value less than or equal to y. Note that $Y_{(1)} = Y_0$ because the initial sample point is considered the first record.

Proposition 4 *The probability of $U_{j,k}(y)$ is given by*

$$\Theta_{j,k}(y) = \frac{\lambda^{j-1} |s(k, j)|}{\alpha^{j-1} (k-1)!} \int_{t=0}^{(p(y))^{\lambda}} (1 - t^{\alpha/\lambda})^{k-1} \, \mathrm{d}t \tag{1}$$

where $s(k, j)$ is a Stirling number of the first kind [1, 24.1.3].

Proof The proof proceeds by induction on k. Consider $k = 1$. The first iterate Y_0 is the first record, $Y_{(1)} = Y_0$, so for $k = 1$ and $j = 1$ the above expression reduces to $\Theta_{1,1}(y) = (p(y))^{\lambda}$, which is indeed the CDF of Y_0.

Now suppose that (1) holds for a given positive integer k, for all $j \in \{1, \ldots, k\}$. Fix $j \in \{1, \ldots, k+1\}$ and $y \in \mathbb{R}$. The event $U_{j,k+1}(y)$ can happen in three ways: the jth record may occur within the first k iterates and fall below y, with the $(k+1)$th iterate failing to produce the $(j+1)$th record; the jth record may occur on the $(k+1)$th iterate, with the $(j-1)$th record already falling below y; the jth record may occur on the $(k+1)$th iterate and fall below y, with the $(j-1)$th record exceeding y. Thus $\Theta_{j,k+1}(y)$ can be written

$$\int_{z=-\infty}^{y} (1 - (p(z))^{\alpha}) \, \mathrm{d}\Theta_{j,k}(z) + \int_{z=-\infty}^{y} (p(z))^{\alpha} \, \mathrm{d}\Theta_{j-1,k}(z) + \int_{z=y}^{\infty} (p(z))^{\alpha} \left(\frac{p(y)}{p(z)} \right)^{\lambda} \mathrm{d}\Theta_{j-1,k}(z).$$

With the substitution $w = (p(z))^\lambda$, this becomes

$$\begin{aligned}
\Theta_{j,k+1}(y) &= \frac{\lambda^{j-1}|s(k,j)|}{\alpha^{j-1}(k-1)!}\int_{w=0}^{(p(y))^\lambda}(1-w^{\alpha/\lambda})^k\,\mathrm{d}w \\
&\quad+\frac{\lambda^{j-2}|s(k,j-1)|}{\alpha^{j-2}(k-1)!}\left(\int_{w=0}^{(p(y))^\lambda} w^{\alpha/\lambda}(1-w^{\alpha/\lambda})^{k-1}\,\mathrm{d}w\right. \\
&\quad\left.+(p(y))^\lambda\int_{w=(p(y))^\lambda}^{1} w^{\alpha/\lambda-1}(1-w^{\alpha/\lambda})^{k-1}\,\mathrm{d}w\right) \\
&= \frac{\lambda^{j-1}|s(k,j)|}{\alpha^{j-1}(k-1)!}\int_{w=0}^{(p(y))^\lambda}(1-w^{\alpha/\lambda})^k\,\mathrm{d}w \\
&\quad+\frac{\lambda^{j-2}|s(k,j-1)|}{\alpha^{j-2}(k-1)!}\left(\frac{-\lambda}{k\alpha}\int_{w=0}^{(p(y))^\lambda} w\,\mathrm{d}((1-w^{\alpha/\lambda})^k)\right. \\
&\quad\left.-\frac{\lambda(p(y))^\lambda}{\alpha k}\left[(1-w^{\alpha/\lambda})^k\right]_{w=(p(y))^\lambda}^{1}\right) \\
&= \frac{\lambda^{j-1}|s(k,j)|}{\alpha^{j-1}(k-1)!}\int_{w=0}^{(p(y))^\lambda}(1-w^{\alpha/\lambda})^k\,\mathrm{d}w \\
&\quad+\frac{\lambda^{j-2}|s(k,j-1)|}{\alpha^{j-2}(k-1)!}\left(\frac{-\lambda}{\alpha k}\left[w(1-w^{\alpha/\lambda})^k\right]_{w=0}^{(p(y))^\lambda}\right. \\
&\quad\left.+\frac{\lambda}{\alpha k}\int_{w=0}^{(p(y))^\lambda}(1-w^{\alpha/\lambda})^k\,\mathrm{d}w+\frac{\lambda(p(y))^\lambda}{\alpha k}(1-(p(y))^\alpha)^k\right) \\
&= \frac{\lambda^{j-1}}{\alpha^{j-1}k!}(k|s(k,j)|+|s(k,j-1)|)\int_{w=0}^{(p(y))^\lambda}(1-w^{\alpha/\lambda})^k\,\mathrm{d}w \\
&= \frac{\lambda^{j-1}|s(k+1,j)|}{\alpha^{j-1}k!}\int_{w=0}^{(p(y))^\lambda}(1-w^{\alpha/\lambda})^k\,\mathrm{d}w.
\end{aligned}$$

□

Proposition 5 *The probability of observing j records in the first k iterates of HASPLID $(\alpha, \lambda; \rho)$ is*

$$(\lambda/\alpha)^{j-1}|s(k,j)|\frac{\Gamma(1+\lambda/\alpha)}{\Gamma(k+\lambda/\alpha)}.$$

Proof The desired probability is $\lim_{y\to\infty} P[U_{j,k}(y)]$, which by Proposition 4 equals

$$\frac{\lambda^{j-1}|s(k,\,j)|}{\alpha^{j-1}(k-1)!}\int_{t=0}^{1}(1-t^{\alpha/\lambda})^{k-1}\,\mathrm{d}t.$$

With the substitution $s = t^{\alpha/\lambda}$ and [1, 6.2.1–2] we can rewrite this as

$$\frac{\lambda^{j-1}|s(k,\,j)|}{\alpha^{j-1}(k-1)!}\times\frac{\lambda}{\alpha}\times\frac{\Gamma(k)\Gamma(\lambda/\alpha)}{\Gamma(k+\lambda/\alpha)}$$

which simplifies to the expression required. □

Notice that the probability expression in Proposition 5 depends only on the ratio λ/α and not on ρ. This makes it practical to use because we can observe the records and estimate the ratio λ/α, but the ρ distribution is impractical to estimate.

Our last result is not required for the method presented in the next section, but may be useful for similar methods of this type.

Proposition 6 *The expected number of records found by HASPLID*$(\alpha, \lambda; \rho)$ *in the first k iterates is*

$$\frac{\lambda}{\alpha}\left(\psi\left(k+\frac{\lambda}{\alpha}\right)-\psi\left(\frac{\lambda}{\alpha}\right)\right) \tag{2}$$

where ψ *is the digamma function* [1, 6.3.1].

Proof A little manipulation of [1, 24.1.3.B] gives

$$\frac{\Gamma(x+k)}{\Gamma(x)}=\sum_{j=0}^{k}|s(k,\,j)|x^{j},$$

whence

$$\sum_{j=1}^{k}j|s(k,\,j)|x^{j}=x\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\Gamma(x+k)}{\Gamma(x)}\right)=x\frac{\Gamma(x+k)}{\Gamma(x)}(\psi(x+k)-\psi(x)).$$

Now by Proposition 5, we can write the expected number of records in the first k iterates as

$$\sum_{j=1}^{k}\frac{j\lambda^{j-1}|s(k,\,j)|\Gamma(1+\lambda/\alpha)}{\alpha^{j-1}\Gamma(k+\lambda/\alpha)}=\frac{\alpha\Gamma(1+\lambda/\alpha)}{\lambda\Gamma(k+\lambda/\alpha)}\times\frac{\lambda\Gamma(\lambda/\alpha+k)}{\alpha\Gamma(\lambda/\alpha)}(\psi(\lambda/\alpha+k)-\psi(\lambda/\alpha))$$

which simplifies to (2). □

Now suppose that R finite independent runs of HASPLID$(\alpha, \lambda; \rho)$ are observed, where α and λ are unknown. Suppose that the rth run is of length k_r and obtains j_r records. From Proposition 5, the log-likelihood depends on α and λ only via their ratio $\zeta = \lambda/\alpha$, being

$$\sum_{r=1}^{R}(\ln|s(k_r,\,j_r)|+(j_r-1)\ln(\zeta)+\ln\Gamma(1+\zeta)-\ln\Gamma(k_r+\zeta)).$$

A maximum likelihood estimate for ζ can be obtained by setting the derivative of the log-likelihood to zero (noting that $d(\ln \Gamma(\zeta))/\mathrm{d}\zeta = \psi(\zeta)$):

$$\sum_{r=1}^{R}(j_r - 1) + \zeta\left(R\psi\,(1+\zeta) - \sum_{r=1}^{R}\psi\,(k_r+\zeta)\right) = 0. \tag{3}$$

This expression decreases monotonically from the nonnegative value $\sum_{r=1}^{R}(j_r - 1)$ at $\zeta = 0$ to the nonpositive value $\sum_{r=1}^{R}(j_r - k_r)$ at $\zeta = \infty$, so that locating a root presents no computational difficulty (we used Matlab's `fzero` function for convenience).

In our Dynamic Multistart Sequential Search framework, detailed in Sects. 5 and 6, we use (3) and Proposition 3 to determine a termination criterion by modeling the performance of a real heuristic with HASPLID($\alpha, \lambda; \rho$). We will assume a value of α on theoretical grounds, and estimate ζ via (3) using our observed j_r and k_r values. Then Proposition 3 provides an expression for the probability of never sampling the ϵ-quantile target region, in R independent HASPLID runs, with numbers $j_1, \ldots, j_R$ of records obtained. We denote this probability as p_{FAIL} and state the result in the following proposition.

Proposition 7 *The probability that R independent runs of HASPLID($\alpha, \lambda; \rho$), with $j_1, \ldots, j_R$ records obtained, never sample the ϵ-quantile target region is*

$$p_{FAIL} = \prod_{r=1}^{R} G(j_r, -\alpha\zeta \ln \epsilon). \tag{4}$$

Hence, (4) is used to determine our termination criterion on the number of independent runs.

5 Dynamic multistart sequential search

This section presents a framework for determining stopping and restarting criteria for heuristic stochastic global optimization methods. The idea is to fit a HASPLID model to the sampled objective function values by estimating the parameters α and λ, and then to use this model to appraise the progress of the heuristic method.

The DMSS framework (pseudocode in Fig. 1 with notation described in Fig. 2) requires three parameters: ϵ represents the target proportion of the domain, e.g., $\epsilon = 0.01$ if we seek a point in the best percentile of the domain with respect to the sampling measure μ; δ represents the failure tolerance, i.e., we want to reach the target with probability at least $1 - \delta$; α is the first parameter of the HASPLID process modeling each run of the heuristic.

The framework also requires two methods of drawing random samples. A random initial domain point will be sampled for each restart, according to a sampling measure μ. (This instruction is denoted $x_{\text{RUN}} \rightsquigarrow \mu$ in the pseudocode in Fig. 1. The symbol '$\rightsquigarrow$' is intended as a combination of the symbols '$\leftarrow$', representing assignment, and '$\sim$', meaning 'distributed according to.') Also, the heuristic defines a candidate sampling measure $\mu[x]$ for each incumbent point x. (Sampling the next candidate is denoted $x_{\text{CAND}} \rightsquigarrow \mu[x_{\text{RUN}}]$.)

The main loop of the DMSS framework iterates through several independent runs of the heuristic. After each run, the value ζ (the ratio of HASPLID parameters λ/α) is reestimated, and then the probability p_{FAIL} of not having reached the target region yet is reestimated. When p_{FAIL} drops below the tolerance parameter δ, the DMSS framework terminates, outputting the location and value of the best point found in any of the runs.

DMSS$(\epsilon, \delta, \alpha)$
$y_{\text{OVERALL}} \leftarrow \infty$
$p_{\text{FAIL}} \leftarrow 1$
$R \leftarrow 0$
$\zeta \leftarrow 1$
while $p_{\text{FAIL}} > \delta$
    increment R
    $x_{\text{RUN}} \rightsquigarrow \mu$
    $y_{\text{RUN}} \leftarrow f(x_{\text{RUN}})$
    $j_R \leftarrow 1$
    $k_R \leftarrow 1$
    $n_{\text{RESTART}} \leftarrow n_{\text{LARGE}}$
    while $k_R < n_{\text{RESTART}}$
        $x_{\text{CAND}} \rightsquigarrow \mu[x_{\text{RUN}}]$
        $y_{\text{CAND}} \leftarrow f(x_{\text{CAND}})$
        increment k_R
        if $y_{\text{CAND}} < y_{\text{RUN}}$
            $(x_{\text{RUN}}, y_{\text{RUN}}) \leftarrow (x_{\text{CAND}}, y_{\text{CAND}})$
            increment j_R
            $n_{\text{RESTART}} \leftarrow k_R \frac{\ln(G(j_R+1, -\alpha\zeta \ln \epsilon))}{\ln(G(j_R, -\alpha\zeta \ln \epsilon))}$
            if $R > 1$
                $n'_{\text{RESTART}} \leftarrow \max\left\{k_1, \ldots, k_{R-1}, \sum_{r=1}^{R-1} k_r \frac{\ln G(j_R+1, -\alpha\zeta \ln \epsilon)}{\ln p_{\text{FAIL}}}\right\}$
                $n_{\text{RESTART}} \leftarrow \min\{n_{\text{RESTART}}, n'_{\text{RESTART}}\}$
    if $y_{\text{RUN}} < y_{\text{OVERALL}}$
        $(x_{\text{OVERALL}}, y_{\text{OVERALL}}) \leftarrow (x_{\text{RUN}}, y_{\text{RUN}})$
    set ζ to solve (3)
    $p_{\text{FAIL}} \leftarrow \prod_{r=1}^{R} G(j_r, -\alpha\zeta \ln \epsilon)$
output $(x_{\text{OVERALL}}, y_{\text{OVERALL}})$

Fig. 1 Pseudocode for the DMSS framework

Symbol list

x_{CAND}	position of candidate point
y_{CAND}	value of candidate point
x_{RUN}	position of best point in current run
y_{RUN}	value of best point in current run
x_{OVERALL}	position of best point found overall
y_{OVERALL}	value of best point found overall
k_R	number of iterations within the current run
j_R	number of records within the current run
R	number of independent runs
p_{FAIL}	estimated probability that we haven't found the target region yet
ζ	estimate of the HASPLID parameter ratio λ/α
n_{RESTART}	number of iterations after which the current run will be stopped (recalculated after each record)
n_{LARGE}	run length limit if no later sample improves initial sample

Fig. 2 Description of notation for the DMSS pseudocode

Each run of the heuristic is performed according to a given candidate sampling kernel, denoted $\mu[\cdot]$. The number of records in each run is stored as well as the best point encountered. The length of each run is decided by a cost-benefit analysis, according to the rate per evaluation at which the current run is reducing $\ln p_{\text{FAIL}}$. (We use this notion of algorithmic "progress" because decreases to the logarithm of the failure probability contribute additively across independent runs of the heuristic.) If the current "progress" rate looks poor in relation to the history of either the current run or all previous runs, then we stop the current run.

In particular, suppose that in run r, the j_rth record occurs at the k_rth evaluation. At this point, and given the current estimate of ζ, the probability that this run has not sampled the target region is given by Proposition 3 as $G(j_r, -\alpha\zeta \ln \epsilon)$. Thus the average rate of "progress," or reduction of $\ln p_{\text{FAIL}}$, per evaluation in the current run is $-\ln G(j_r, -\alpha\zeta \ln \epsilon)/k_r$. If the run continues, and the next record occurs at iteration k', then the new average rate of progress will be $-\ln G(j_r + 1, -\alpha\zeta \ln \epsilon)/k'$. If

$$k' > k_r \frac{\ln\left(G(j_r + 1, -\alpha\zeta \ln \epsilon)\right)}{\ln\left(G(j_r, -\alpha\zeta \ln \epsilon)\right)},$$

then the average rate of progress is decreasing, and the run should be stopped, since it is reasonable to suppose we might make progress more rapidly by restarting the heuristic.

The above criterion tries to ensure that each run stops once it is past its prime. However, some runs are more productive than others, and if a run appears to be making slower progress than previous runs, then we wish to abandon it, whether or not its own slow rate of progress has peaked. Additionally, therefore, from the second run onward, the current run is compared with previous runs. If the current run exceeds all previous runs in length, then it is only allowed to continue as long as its average rate of "progress" per iteration exceeds the average rate over all previous runs.

6 Experiments

To implement DMSS, a stochastic search must be selected. For simplicity we chose a simple elitist random walk in which, at each iteration, a candidate is chosen uniformly from the interior of an axially oriented hypercube of edge length 0.2, centered on the current point, and accepted only if it improves on the current point. In order to evaluate DMSS, we compared its performance against two "control" algorithms, both running the same elitist random walk for the same number of objective function evaluations: one with the same number of restarts as the DMSS run, but equally spaced during the computation, the other with no restarts at all.

The experiments were performed on three test problems from the global optimization literature: centered- and shifted-optimum versions of a sinusoidal problem [2], and the Lennard–Jones cluster problem [20]. A few versions of each test problem were optimized, varying in dimension; furthermore, the sinusoidal optimizations were repeated 180 times, and the Lennard–Jones optimizations were repeated 400 times.

We have opted for ϵ to shrink exponentially with the problem dimension d, loosely corresponding to a constant error tolerance in either the range or any single decision variable. The exponential rate and δ were chosen fairly arbitrarily. We should also point out here that, because HASPLID only approximately models the heuristic's behavior, the true probability of DMSS terminating within the target ϵ-quantile may be substantially greater or less than $1 - \delta$.

Table 1 Results on centered-optimum sinusoidal test function

$f : x \in [0, \pi]^d \mapsto -2.5 \prod_{j=1}^{d} \sin x_j - \prod_{j=1}^{d} \sin(5x_j), \epsilon = (0.01)^d, \delta = 0.001, \alpha = 1/d$

Dim d	Optimal value	Mean # eval.s	Mean best value found by			Outperformed	
			DMSS	Equal	Single	Equal	Single
2	−3.5	5,826	−3.467	−3.489	−2.136	86(0.6)	$164(4 \times 10^{-32})$
4	−3.5	19,700	−3.152	−3.022	−1.621	$107(1 \times 10^{-2})$	$165(4 \times 10^{-33})$
8	−3.5	97,189	−2.895	−2.578	−1.444	$124(4 \times 10^{-7})$	$152(8 \times 10^{-22})$
16	−3.5	12,130	−1.989	−1.319	−1.346	$154(3 \times 10^{-23})$	$129(6 \times 10^{-9})$

The last two columns show the numbers of experiments out of 180 in which DMSS outperformed the 'Equal' and 'Single' implementations: all except one are greater than 90, and thus in DMSS's favor. They are followed in parentheses by two-sided p-values, each representing the probability of as great or greater a difference between the outperformance count and the expected value 90 under the null hypothesis that DMSS is in fact identical to the 'Equal' or 'Single' implementation

Lastly, we have set the parameter $\alpha = 1/d$, that is, we have assumed the difficulty of finding an improvement to be inversely proportional to the dimension of the improving region. Asymptotically, this holds for a variety of local search methods; for instance, in Improving Hit-and-Run [24], as the search nears the global optimum, if we assume the improving region to shrink by uniform scaling, then for any given ray of approach to the optimum and any given search direction, the search line's intersection with the improving region is proportional to the improving region's scaling (while its intersection with the domain is approximately constant).

The experimental results appear in Tables 1, 2, and 3. Unsurprisingly, we have found that DMSS is generally superior to a long single run of the sequential search, at least for the problems investigated. More worthy of note is DMSS's tendency to outperform the multi-start implementation of the sequential search labelled 'Equal,' in which the same number of evaluations are redistributed equally across the same number of restarts. (Thus, for instance, if DMSS terminated after 4 runs, with 186, 235, 76 and 390 evaluations in the runs, then the total number of evaluations would be $186 + 235 + 76 + 390 = 887$; and for comparison, the 'Equal' algorithm would execute 4 runs of the sequential search, with 222, 222, 222 and 221 function evaluations.) DMSS significantly outperforms the 'Equal' and 'Single' algorithms and the p-values are reported in parentheses in the last two columns of Tables 1, 2, and 3. All three search methods suffer as dimension increases, 'Equal' most quickly and 'Single' most slowly. DMSS treads a middle ground between the two, but maintains performance superior to both in nearly every case. These numerical results indicate that DMSS has some ability to distinguish dynamically between runs which are progressing well and runs which are stalled.

7 Conclusion

We have introduced an instance of Hesitant Adaptive Search called Hesitant Adaptive Search with Power-Law Improvement Distribution, which allows us to characterize separately the frequency and the quality of improvements found by sequential search procedures. We model the search procedures dynamically by HASPLID, leading to our adaptive criteria for search

Table 2 Results on shifted-optimum sinusoidal test function, notated as Table 1

$f\colon x \in [0,\pi]^d \mapsto -2.5\prod_{j=1}^{d}\sin(x_j-\pi/2)-\prod_{j=1}^{d}\sin(5(x_j-\pi/2)),\ \epsilon=(0.005)^d,\ \delta=0.001,\ \alpha=1/d$

Dim d	Optimal value	Mean # eval.s	Mean best value found by			Outperformed	
			DMSS	Equal	Single	Equal	Single
2	−3.5	4,433	−3.454	−3.450	−1.093	85(0.5)	$173(1\times10^{-42})$
4	−3.5	35,433	−2.904	−2.944	−1.229	81(0.2)	$169(2\times10^{-37})$
8	−3.5	2,918	−2.376	−2.143	−1.303	$123(1\times10^{-6})$	$154(3\times10^{-23})$
16	−3.5	1,911	−1.174	−0.794	−1.084	$144(2\times10^{-16})$	$112(1\times10^{-3})$

Table 3 Results on Lennard–Jones cluster problem, notated as Table 1

f is the Lennard–Jones potential, $\epsilon=(0.01)^d,\ \delta=0.001,\ \alpha=1/d$

Dim d	Optimal value	Mean # eval.s	Mean best value found by			Outperformed	
			DMSS	Equal	Single	Equal	Single
18	−19.822	60,769	−9.864	−9.742	−7.191	214(0.2)	$347(5\times10^{-54})$
42	−56.816	59,079	−10.930	−10.776	−8.475	210(0.3)	$336(1\times10^{-45})$
90	−139.636	57,918	−12.378	−12.107	−10.093	$228(6\times10^{-3})$	$320(4\times10^{-35})$

Each optimization was performed 400 times; all of the counts in the final two columns are in DMSS's favor. Note that the dimension associated with the 8-, 16- and 32-atom problems is thrice the number of atoms, minus 6 (continuous symmetries were factored out). The search space restricted the first atom to the origin, the second to the line segment $[0, 1.2]\times\{(0, 0)\}$, the third to the square $[0, 1.2]^2\times\{0\}$, and all others to the axial hypercube of edge length $4d^{1/3}$ centered on (0.6, 0.6, 0.6)

restarting and termination. This is formalized in the framework called Dynamic Multistart Sequential Search. Numerical experiments demonstrate that the DMSS framework makes efficient decisions about the lengths of runs of a sequential search applied to standard test problems, and terminates appropriately.

The DMSS framework presented here ignores function values across separate runs of the sequential search: of course, it retains the best value found in any run, and within each run it counts the number of records achieved, but the run-termination criterion is independent of whether the current run is finding new overall records. This independence is convenient theoretically, but probably not optimal computationally. A modification which relates the separate runs by their function values but retains the theoretical outlook espoused by DMSS would be desirable. Such an approach may require future research into a parametric Bayesian estimation of the range distribution ρ.

Acknowledgments This research was supported in part by the National Science Foundation (USA) under Grant DMI-0244286.

References

1. Abramowitz, M., Stegun, I.: Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables. Dover, New York (1964)

2. Ali, M.M., Khompatraporn, C., Zabinsky, Z.B.: A numerical evaluation of several global optimization algorithms on selected benchmark test problems. J. Glob. Optim. **31**, 635–672 (2005)
3. Atkinson, A.: A segmented algorithm for simulated annealing. Stat. Comput. **2**, 221–230 (1992)
4. Betrò, B., Schoen, F.: Sequential stopping rules for the multistart algorithm in global optimisation. Math. Program. **38**, 271–286 (1987)
5. Boender, C., Rinnooy Kan, A.: Bayesian stopping rules for multistart global optimization methods. Math. Program. **37**, 59–80 (1987)
6. Boender, C.G.E., Romeijn, H.E.: Stochastic methods. In: Horst, R., Pardalos, P. (eds.) Handbook of Global Optimization, pp. 829–869. Kluwer, The Netherlands (1995)
7. Bulger, D.W., Wood, G.R.: Hesitant adaptive search for global optimisation. Math. Program. **81**, 89–102 (1998)
8. Dekkers, A., Aarts, E.: Global optimization and simulated annealing. Math. Program. **50**, 367–393 (1991)
9. Dür, M., Khompatraporn, C., Zabinsky, Z.B.: Solving fractional problems with dynamic multistart improving Hit-and-Run. Ann. Oper. Res. **156**, 25–44 (2007)
10. Glidewell, M., Ng, K., Hensel, E.: A combinatorial optimization approach as a pre-processor for impedance tomography. In: Proceedings of the Annual Conference of the IEEE/Engineering in Medicine and Biology Society, pp. 1–2 (1991)
11. Hajek, B.: Cooling schedules for optimal annealing. Math. Oper. Res. **18**, 311–329 (1988)
12. Khompatraporn, C.: Analysis and Development of Stopping Criteria for Stochastic Global Optimization Algorithms. Ph.D. Dissertation, University of Washington, Washington (2004)
13. Li, H., Lim, A.: A Metaheuristic for the Pickup and Delivery Problem with Time Windows. In: Proceedings of the 13th IEEE International Conference on Tools with Artificial Intelligence, pp. 160–167 (2001)
14. Locatelli, M.: Convergence of a simulated annealing algorithm for continuous global optimization. J. Glob. Optim. **18**, 219–234 (2000)
15. Lundy, M., Mees, A.: Convergence of an annealing algorithm. Math. Program. **34**, 111–124 (1986)
16. Museli, M.: A theoretical approach to restart in global optimization. J. Glob. Optim. **10**, 1–16 (1997)
17. Romeijn, H.E., Smith, R.L.: Simulated annealing and adaptive search in global optimization. Probab. Eng. Inf. Sci. **8**, 571–590 (1994)
18. Treadgold, N., Gedeon, T.: Simulated annealing and weight decay in adaptive learning: the SARPROP algorithm. IEEE Trans. Neural Netw. **9**, 662–668 (1998)
19. Theodosopoulos, T.V.: Some remarks on the optimal level of randomization in global optimization. In: Pardalos, P., Rajasekaran, S., Rolim, J. (eds.) Randomization Methods in Algorithm Design, DIMACS Series 43., pp. 303–318. American Mathematical Society, RI (1999)
20. Wales, D.J., Doye, J.P.K.: Global optimization by basin-hopping and the lowest energy structures of Lennard–Jones clusters containing up to 110 atoms. J. Phys. Chem. A **101**, 5111–5116 (1997)
21. Wood, G.R., Zabinsky, Z.B., Kristinsdóttir, B.P.: Hesitant adaptive search: the distribution of the number of iterations to convergence. Math. Program. A **89**, 479–486 (2001)
22. Zabinsky, Z.B.: Stochastic Adaptive Search for Global Optimization. Kluwer, The Netherlands (2003)
23. Zabinsky, Z.B., Smith, R.L.: Pure adaptive search in global optimization. Math. Program. **53**, 323–338 (1992)
24. Zabinsky, Z.B., Smith, R.L., McDonald, J.F., Romeijn, H.E., Kaufman, D.E.: Improving Hit-and-Run for global optimization. J. Glob. Optim. **3**, 171–192 (1993)
25. Zielinski, R.: A statistical estimate of the structure of multiextremal problems. Math. Program. **21**, 348–356 (1981)