Improving Hit-and-Run for Global Optimization*

ZELDA B. ZABINSKY[1], ROBERT L. SMITH[2], J. FRED McDONALD[3], H. EDWIN ROMEIJN[4] and DAVID E. KAUFMAN[5]

[1]*Industrial Engineering Program, FU-20, University of Washington, Seattle, Washington 98195, USA*; [2] *Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109-2117, USA*; [3]*Department of Mathematics and Statistics, University of Windsor, Windsor, Ontario, Canada N9B 3P4*; [4]*Department of Operations Research & Tinbergen Institute, Erasmus University Rotterdam, P.O. Box 1738, NL-3000 DR Rotterdam, The Netherlands*; *and* [5]*Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109-2117, USA*

(Received: 14 December 1990; accepted: 17 March 1992)

Abstract. Improving Hit-and-Run is a random search algorithm for global optimization that at each iteration generates a candidate point for improvement that is uniformly distributed along a randomly chosen direction within the feasible region. The candidate point is accepted as the next iterate if it offers an improvement over the current iterate. We show that for positive definite quadratic programs, the expected number of function evaluations needed to arbitrarily well approximate the optimal solution is at most $O(n^{5/2})$ where n is the dimension of the problem. Improving Hit-and-Run when applied to global optimization problems can therefore be expected to converge polynomially fast as it approaches the global optimum.

Key words. Random search, Monte Carlo optimization, algorithm complexity, global optimization.

1. Introduction

Random search algorithms offer considerable promise as efficient optimization methods for a large class of problems. Recent results [16, 28] demonstrate that it is theoretically possible for a random search algorithm to achieve a computational complexity that is, on average, linear in dimension. In this paper, we introduce and implement a new sequential random search algorithm named Improving Hit-and-Run (IHR).

Sequential random search procedures are designed to address a standard optimization problem,

$$\min_{x \in S} f(x)\,, \qquad (P)$$

where x is an n-dimensional vector, S is a convex, compact, full-dimensional subset of $\mathbf{R}^n$, and f is a real-valued continuous function defined over S. All sequential random search procedures generate a sequence of random points $\{X_j\}$ which may depend on the previous point or several of the previous points. The

*Paper presented at the II. IIASA-Workshop on Global Optimization, December 9–14, 1990, Sopron (Hungary).

Journal of Global Optimization **3**: 171–192, 1993.

concept underlying sequential step size algorithms [17, 18, 15, 23, 21, 13, 22, 25] is to generate the next random point X_{j+1} by taking a specified step in a random direction from the previous point X_j. These algorithms are based on the iterative formula, for $j = 1, 2, \ldots$

$$X_{j+1} = \begin{cases} X_j + s_j D_j & \text{if } f(X_j + s_j D_j) < f(X_j), \\ X_j & \text{otherwise}, \end{cases}$$

where D_j is the direction vector, s_j is the step size, and X_j is the point generated in the jth iteration. The direction vector is often, but not necessarily, obtained by sampling from a uniform distribution on a unit hypersphere. The method of choosing the step size is specific to each algorithm.

There is experimental evidence in the literature that suggests sequential random search algorithms are efficient for large dimensional quadratic programs. Schumer and Steiglitz [23] provide experimental evidence that the number of function evaluations increases linearly with dimension for their adaptive step size algorithm on the following three test functions: $\Sigma_{i=1}^n x_i^2$, $\Sigma_{i=1}^n x_i^4$, and $\Sigma_{i=1}^n a_i x_i^2$. They also prove that the average number of function evaluations for an optimum relative step size random search restricted to an unconstrained quadratic objective function is asymptotically linear in dimension. Schrack and Borowski [21] report experimental results on a quadratic test function, $\Sigma_{i=1}^n x_i^2$, that doubling the dimension doubles the number of function evaluations required for their random search algorithm. Solis and Wets [25] experimentally verified a linear correlation between the number of function evaluations and dimension for their own variation of the step size algorithm on a quadratic test function. They provided a justification of this linearity condition based on the tendency of these algorithms to maintain a constant probability of successful improvement.

There is also theoretical justification that sequential random search algorithms are efficient for a larger class of global optimization problems. An analysis of a random search procedure called pure adaptive search [16, 28] proves that it is theoretically possible for a sequential random search procedure to achieve linear complexity (in improving iterates) for global optimization problems. Pure adaptive search (PAS) constructs a sequence of points uniformly distributed within a corresponding sequence of nested improving regions. In Zabinsky and Smith [28], complexity is measured as the expected number of iterations needed to get arbitrarily close to the solution with a specified degree of certainty. In pure adaptive search, each iteration corresponds to an improving point. Unfortunately, as pointed out in [16, 28], pure adaptive search is difficult to implement directly due to the problem of efficiently generating a point according to a uniform distribution in a general region. However, recent research has shown that Hit-and-Run methods [4, 5, 11, 24, 2] can generate a sequence of points that asymptotically approach a uniform distribution, as shown by Smith [24]. Hit-and-Run generates a sequence of random points by providing a random direction and then providing a uniform random point in that direction. Since no better

alternative for generating uniform points is known at this time, we use Hit-and-Run to generate an improving point in the level set at each iteration. The resulting algorithm is called Improving Hit-and-Run.

We analyze the computational efficiency of Improving Hit-and-Run for a specific class of quadratic programs, and theoretically establish an $O(n^{5/2})$ upper bound on the expected number of function evaluations.

2. Improving Hit-and-Run

Improving Hit-and-Run (IHR) is designed to be easy to implement and at the same time to inherit the efficiency of pure adaptive search. As already noted, the difficulty of directly implementing pure adaptive search lies in the iterative step of efficiently generating a uniform point in the improving region. This can be achieved by executing the Hit-and-Run algorithm at each iteration, restricting the sequence of Hit-and-Run points generated within an iterative step to the improving feasible region. Since the resulting sequence is only asymptotically guaranteed to be uniform within the improving region, we must decide on how long a sequence to generate. A class of algorithms can therefore be parametrically related to the length of the corresponding Hit-and-Run sequences. At one extreme, we know that when the Hit-and-Run sequences are very long and hence provide a close approximation to sampling from a uniform distribution, we have a good approximation to pure adaptive search and can expect the number of improving points to be linear in dimension. At the other extreme, the Hit-and-Run sequence is reduced to a length of *one*. It is the latter algorithm with Hit-and-Run sequences of length one that we effectively adopt. We call the resulting algorithm Improving Hit-and-Run.

Although the sequence of points generated per iteration is insufficiently long to well approximate uniformity, the hope is that the algorithm may nonetheless inherit a polynomial complexity similar to that enjoyed by pure adaptive search. In fact, we will prove for the class of positive definite quadratic programs that the expected number of iterations is $O(n^2)$, but more importantly the expected number of *function evaluations* remains polynomial, in particular, $O(n^{5/2})$.

Improving Hit-and-Run is a sequential random search algorithm. The basic structure of IHR is to generate a random direction followed by a candidate point that is along a random step in that direction. A positive definite matrix H in the algorithm controls the direction distribution. If the matrix H is the identity matrix, then the direction distribution is uniform on a hypersphere. However in order to achieve the analytically established polynomial performance bound, H must be set to the Hessian of the quadratic objective function. In practice, for a global optimization problem H would be locally estimated as in quasi-Newton local search procedures. (See [29] for experimental results on general global optimization problems.) Note that IHR reduces to the Hide-and-Seek continuous simulated annealing algorithm with temperature 0 (see [3]).

Given a positive definite matrix H, we define the Improving Hit-and-Run algorithm as follows.

IMPROVING HIT-AND-RUN (IHR)

Step 0. Initialize $X_0 \in S$, $Y_0 = f(X_0)$, and $j = 0$.

Step 1. Generate a direction vector D_j from the multivariate normal distribution with mean 0 and covariance matrix H^{-1}.

Step 2. Generate a step size s_j uniformly from L_j, the set of feasible step sizes from the current iteration point X_j in the direction D_j, where

$$L_j = \{\lambda \in \boldsymbol{R} : X_j + \lambda D_j \in S\}.$$

If $L_j = \emptyset$, go to Step 1.

Step 3. Update the new point if it is improving, i.e. set

$$X_{j+1} = \begin{cases} X_j + s_j D_j & \text{if } f(X_j + s_j D_j) < Y_j \\ X_j & \text{otherwise} \end{cases}$$

and set $Y_{j+1} = f(X_{j+1})$.

Step 4. If the stopping criterion is met, stop. Otherwise increment j and return to Step 1.

Improving Hit-and-Run is straightforward to implement. Generating a random direction, as defined in Step 1, can be accomplished by appropriately transforming a multivariate standard normal deviate, using a decomposition of the covariance matrix H^{-1}. When $H = I$, this simplifies to computing the direction vector D_j of unit length given by,

$$\dot{D_j} = (d_1, d_2, \ldots, d_n)\left(\sum_{i=1}^{n} |d_i|^2\right)^{-1/2},$$

where d_i, $i = 1, 2, \ldots, n$ are sampled independently from a standard normal distribution, $N(0, 1)$ (see [12]). For the general case, we first find a matrix A of rank n such that $H = A'A$. One possible choice for A is obtained by diagonalizing H, i.e., by writing $H = ZDZ'$, where D is a diagonal matrix containing the eigenvalues of H, and Z is a matrix having the corresponding eigenvectors as its columns. This approach yields $A = ZD^{\frac{1}{2}}Z'$. Another possibility is to perform a Cholesky decomposition $H = LL'$, where L is a lower triangular matrix, yielding $A = L'$. If D_j is distributed according to the standard normal distribution (i.e., D_j has mean zero and covariance matrix I), then $A^{-1}D_j$ has the desired distribution, i.e., $A^{-1}D_j$ is normally distributed with mean zero and covariance matrix H^{-1}.

Generating the step size in Step 2 is straightforward as long as it is possible to find the points where the line through X_j in the direction D_j intersects S. This is particularly easy for the case where S is a set of linear constraints. The points of intersection are easily found with a modified minimum ratio test [14]. Alternatively, one can enclose a general region in a box and identify the corresponding points

of intersection, and use one dimensional acceptance-rejection along the resulting line segment until a feasible point is found.

3. Polynomial Performance

3.1. DEFINITIONS AND NOTATION

Before proceeding with the analysis of the algorithms computational complexity, we begin with notation and definitions.

For the optimization problem (P), let (x_*, y_*) denote an optimal solution, where

$$y_* = \min_{x \in S} f(x)$$

and

$$f(x_*) = y_* .$$

The value x_* need not be unique. It will also be convenient to define the maximum,

$$y^* = \max_{x \in S} f(x) .$$

Let P_y denote the *level set* of the problem (P) at objective function value y,

$$P_y = \{x \in S : f(x) < y\}$$

for $y_* < y \leq y^*$.

Let $\{X_j\}$, $X_j \in S$, $j = 0, 1, 2, \ldots$ be the sequence of *sample points* generated by Improving Hit-and-Run on (P), with the subscript j denoting iteration count. Notice that each iteration corresponds to one sample point, and one function evaluation. Let $\{Y_j\}$, $y_* \leq Y_j \leq y^*$, $j = 0, 1, 2, \ldots$ be the corresponding sequence of *sample values* generated by the algorithm on (P), $Y_j = f(X_j)$. Improving Hit-and-Run is defined so that the current sample point is only updated ($X_{j+1} = X_j + s_j D_j$) when it improves and repeated otherwise ($X_{j+1} = X_j$), so that $Y_{j+1} \geq Y_j$ for $j = 0, 1, 2, \ldots$

The sequences $\{X_j\}$ and $\{Y_j\}$ are random variables since the directions and step sizes are random. The *distribution of improvement* is defined to be the probability that the jth objective function value Y_j is at most y. This is the probability that the jth sample point lies within the level set P_y, so that

$$P(Y_j < y) = P(X_j \in P_y)$$

for $j = 0, 1, 2, \ldots$ and $y_* < y \leq y^*$. The *conditional distribution of improvement*, denoted

$$P(Y_{j+1} < y \mid Y_j = w)$$

for $j = 0, 1, 2 \ldots$ and $y_* < y$, $w \leq y^*$ is defined to be the probability of obtaining

an objective function value of at most y in a single iteration, starting from sample value w.

It will be convenient for the analysis to identify the sequence of *record values* and their corresponding *improving points*. Let j_k denote the subscript corresponding to the kth improving point. Thus the point X_{j_k} is the kth record point.

The analysis will be performed on a class of mathematical programs with "elliptical" level sets. This class of programs includes positive definite quadratic programs as a special case. We define an *elliptical program* to be the mathematical program (P), where $f(x)$ can be expressed as

$$f(x) = h(r) \quad \text{with} \quad r = \|x - x_*\|_A ,$$

where A is an $n \times n$-matrix of full rank, and h is strictly monotonically increasing for $r \geq 0$. The norm $\|\cdot\|_A$ is defined for any n-dimensional point z as

$$\|z\|_A = \|Az\| ,$$

where $\|\cdot\|$ denotes the standard Euclidean norm. Note that $\|\cdot\|_I = \|\cdot\|$. In this class of programs, we assume x_* is interior to S. Furthermore, for this class of programs we use the IHR algorithm with H chosen to be equal to $A'A$.

An elliptical program can be interpreted geometrically as a problem with level sets that are elliptical in shape, and nested about the optimum. The function $h(r)$ can be interpreted as a means of layering the level sets. In the special case where $A = I$ the level sets become spherical in shape, and the variable r can be geometrically interpreted as the radius of the level set. We will call the special case of $A = I$ a *spherical program*. Notice that an elliptical program is convex if and only if $h(r)$ is convex in r. The class of elliptical programs also includes cases where $h(r)$ is nonconvex in r.

Much of the analysis is developed for spherical programs and then extended to elliptical programs using the A matrix. For spherical programs, we use the geometric interpretation of the radius of the level set and define $\{R_k\}$, $k = 0, 1, 2, \ldots$ to be the sequence of *radii* associated with the level sets of the *improving* points generated by Improving Hit-and-Run, i.e., $R_k = \|X_{j_k} - x_*\|_I$, where X_{j_k} is the k^{th} record point. Also, $R_k = h^{-1}(f(X_{j_k})) = h^{-1}(Y_{j_k})$. Notice that the sequence $\{R_k\}$ is defined only for record points, and thus is strictly decreasing, i.e., $R_k > R_{k+1}$ for $k = 0, 1, 2, \ldots$

To measure computational complexity we define a *sample point count*, $N(r)$, as the number of points sampled to achieve $Y_j \leq h(r)$ for $0 < r \leq q$. The sample point count is equal to the count of iterations and equivalently the count of function evaluations, since each iteration involves exactly one additional function evaluation. There is very little overhead other than the function evaluation associated with each iteration, so this is a very accurate indication of total computing time. In the next section we prove our main result that the expected value of $N(r)$ is at most $O(n^{5/2})$.

For the analysis we also define an *improving point count*, $K(r)$, as the number

of improving points needed to achieve $Y_{j_k} \leq h(r)$ for $0 < r \leq q$. As an intermediate result, we prove that the expected value of $K(r)$ is at most quadratic in dimension.

3.2. COMPLEXITY ANALYSIS

We now turn to an analysis of the complexity of Improving Hit-and-Run. To simplify the presentation of the analysis, we assume without loss of generality that $Y_0 = y^*$, and S equals the level set associated with X_0. We also assume for the spherical program that $R_0 = q$.

We will proceed by first showing that we can restrict our attention to the case where $A = I$, i.e., a spherical program where the level sets are concentric spheres instead of ellipsoids. To facilitate the proof of this result, we transform an elliptical program (P) with corresponding matrix A into a spherical program $(\tilde{P})$ by defining a transformed point $\tilde{x} = Ax$, and

$$\tilde{S} = \{\tilde{x} \in \boldsymbol{R}^n : A^{-1}\tilde{x} \in S\}$$

and

$$\tilde{f} : \tilde{S} \to \boldsymbol{R}$$

$$\tilde{f}(\tilde{x}) = f(A^{-1}\tilde{x}) .$$

Denote the transformed problem of minimizing $\tilde{f}$ over $\tilde{S}$ by $(\tilde{P})$, and note that

$$\min_{x \in S} f(x) = \min_{\tilde{x} \in \tilde{S}} \tilde{f}(\tilde{x}) .$$

THEOREM 3.1. *Performing the IHR algorithm on the elliptical problem* (P)

$$\min_{x \in S} f(x)$$

using $H = A'A$ *is equivalent* (*under the identification* $\tilde{x} \leftrightarrow Ax$) *to performing the IHR algorithm on the problem* $(\tilde{P})$

$$\min_{\tilde{x} \in \tilde{S}} \tilde{f}(\tilde{x})$$

using $H = I$.

Proof. See Appendix.

It follows from Theorem 3.1 that in the remainder of the complexity analysis, we need only consider the class of spherical programs. Since the number of points generated is invariant, all complexity results that will be obtained for this class also hold for the more general case of elliptical programs.

We next determine the conditional distribution of improvement for Improving Hit-and-Run on a spherical program. The conditional probability of making a specified improvement on a single iteration depends on the position of the current point,

$$P(Y_{j+1} < y \mid Y_j = w) = E[P(Y_{j+1} < y \mid X_j, Y_j = w)]$$
$$= E[P(X_{j+1} \in P_y \mid X_j, Y_j = w)].$$

The probability within the last expectation can be expressed as an integral in terms of the random direction generated from the point x, and the ratio of the length of improving line segment(s) in that direction,

$$P(X_{j+1} \in P_y \mid X_j = x, Y_j = w) = \int_{\partial D} \| L_{P_y}(d, x) \| / \| L_S(d, x) \| dF_{\partial D}(d) \quad (1)$$

where ∂D is the boundary of the unit sphere, and $F_D(\cdot)$ is the cumulative distribution function for the normalized random direction vector. Note that the normalized direction vector has a uniform distribution on the boundary of the unit sphere (see [12]). The expression $\| L_{P_y}(d, x) \|$ is the combined lengths of the line segments formed by the intersection of the level set P_y with the line in direction d originating at x. In words, the probability of achieving an objective function value of at most y, starting at point x with $f(x) = w$, is the probability of landing within P_y given direction d, i.e., $\| L_{P_y}(d, x) \| / \| L_S(d, x) \|$, integrated over all feasible improving directions.

For general mathematical programs, the conditional probability of improvement depends on the exact location of X_j, and makes it difficult to derive a general expression. However, for the class of spherical programs, we can analytically derive the conditional probability of improvement.

LEMMA 3.2. *For any spherical program* (P), *the conditional probability of improvement on the next sample point for IHR is given by*

$$P(Y_{j+1} < h(s) \mid Y_j = h(r), Y_0 = h(q)) \equiv p(s; r, q)$$
$$= \left(\frac{r}{q}\right)\left(\frac{s}{r}\right)^n \frac{1}{n} F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{s^2}{q^2}, \frac{s^2}{r^2}\right)$$

for $0 < s \leq r \leq q$, *and for* $j = 0, 1, 2, \ldots$, *where* $F_1(a, b, c, d; x, y)$ *is a generalized hypergeometric function (see* [1] *or* [10]).

Proof. See Appendix.

Notice that the special case where $s = r$, denoted $p(r; r, q)$, is the probability that a sample point is improving given the current point is at $Y_j = h(r)$ and simplifies to (using [10] pp. 1055, equation 10):

$$p(r; r, q) = \left(\frac{r}{q}\right) \frac{1}{\gamma(n)} F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; \frac{r^2}{q^2}\right) \quad (2)$$

where

$$\gamma(n) = \frac{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}$$

for $0 < r \leq q$, and for $j = 0, 1, 2, \ldots$, where $F(a,b;c;z)$ is Gauss' hypergeometric function (see [1] or [10]). We now express the expected number of sample points, and hence the expected number of function evaluations, in terms of the expected number of improving points.

THEOREM 3.3. *For any spherical program (P), the expected number of IHR sample points needed to achieve an objective function value of $h(r)$ or better is bounded by*

$$E[N(r)] \leq \frac{E[K(r)]}{p(r;r,q)}$$
$$\leq \frac{q}{r}\gamma(n)E[K(r)]$$

for $0 < r \leq q$ and where

$$\gamma(n) = \frac{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}$$
$$= O(\sqrt{n}) .$$

Proof. See Appendix.

We now analyze the performance of improving points of IHR. We start by evaluating a conditional probability for improving points and then develop a bound on the expected number of improving points $E[K(r)]$ in terms of dimension.

LEMMA 3.4. *For any spherical program (P), the conditional probability distribution of improving points for IHR is given by*

$$P(R_{k+1} < s \mid R_k = r, R_0 = q) = \frac{p(s;r,q)}{p(r;r,q)}$$
$$\geq \frac{(s/r)^n}{n}$$

for $0 < s \leq r \leq q$, and for $k = 0, 1, 2, \ldots$.

Proof. See Appendix.

THEOREM 3.5. *For any spherical program (P), the expected number of IHR improving points needed to achieve an objective function value of $h(r)$ or better is bounded by*

$$E[K(r)] \leq nE[K(r)^{PAS}]$$
$$= O(n^2)$$

for $0 < r \leq q$ *where* $K(r)^{PAS}$ *is the corresponding expected number of improving points for PAS.*

Proof. See Appendix.

The culminating result of polynomial complexity follows from combining Theorems 3.3 and 3.5 with Theorem 3.1.

COROLLARY 3.6. *For any elliptical program* (P), *the expected number of IHR sample points needed to achieve an objective function value of* $h(r)$ *or better is bounded by*

$$E[n(r)] \leq \frac{q}{r}\gamma(n)nE[K(r)^{PAS}]$$
$$= O(n^{5/2})$$

for $0 < r \leq q$.

Proof. From Theorem 3.3 we have, for $0 < r \leq q$,

$$E[N(r)] \leq \frac{q}{r}\gamma(n)E[K(r)]$$

and from Theorem 3.5

$$\leq \frac{q}{r}\gamma(n)nE[K(r)^{PAS}]$$

which yields

$$= O(n^{5/2}) .$$ ■

4. Discussion

Improving Hit-and-Run has been shown to have search effort that is polynomially bounded in dimension for the class of elliptical programs. This complexity is attainable for strictly convex quadratic programs by choosing H to be the Hessian of the objective function. Although this class is small, we can (in principle) construct an algorithm with *asymptotically* the same polynomial complexity if the objective function is twice continuously differentiable at the global minimum x_*, and if the Hessian H_* at that point, is strictly positive definite. In that case, we can, for values of x sufficiently close to x_*; approximate the objective function by

$$f(x) \approx f_* + \frac{1}{2}(x - x_*)'H_*(x - x_*)$$

by using the second order Taylor approximation at the global minimum. The approximating function is now a strictly convex quadratic function, for which the complexity results in this paper apply. However, we should know H_* in advance to implement the algorithm.

Alternatively, if f is twice continuously differentiable everywhere, we can, at

iteration j, approximate the objective function by

$$f(x) \approx f(X_j) + \nabla f(X_j)'(x - X_j) + \frac{1}{2}(x - X_j)'H(X_j)(x - X_j),$$

where $\nabla f(X_j)$ denotes the gradient at X_j, and $H(X_j)$ denotes the Hessian at X_j. This suggests that, in every iteration, we should use the Hessian at the iteration point in the IHR algorithm. If this scheme is computationally too expensive, or if $H(X_j)$ is not positive definite at some iteration point, one should turn to approximation methods for the Hessian. For example, DFP and BFGS approximation schemes are used in quasi-Newton local search algorithms (see e.g. [20]).

The original hope for Improving Hit-and-Run was that its computational complexity would be comparable to pure adaptive search. If extremely long sequences of Hit-and-Run were used to approximate PAS then we would expect a linear bound in dimension on improving points at a comparably high cost of obtaining an improving point. For IHR, which has extremely short sequences of Hit-and-Run (length one), the expected number of improving points has a quadratic bound which although worse than PAS is associated with a comparably low cost of obtaining an improving point. In fact, the total expected number of function evaluations is only $O(n^{5/2})$. Thus IHR can be viewed as an optimizing version of Hit-and-Run, and the complexity results provide support for the value of short Hit-and-Run sequences within an optimizing framework.

Acknowledgements

This work was supported in part by NATO Collaborative Research Grant 0119/89. The work of Z.B. Zabinsky was partially supported by the Graduate School Research Fund at the University of Washington. The work of J.F. McDonald was partially supported by the Natural Sciences and Engineering Research Council of Canada under Grant A4625.

Appendix

A. PROOFS FOR THE COMPLEXITY ANALYSIS

THEOREM 3.1. *Performing the IHR algorithm on the elliptical problem* (P)

$$\min_{x \in S} f(x)$$

using $H = A'A$ *is equivalent to performing the IHR algorithm on the problem* $(\tilde{P})$

$$\min_{\tilde{x} \in \tilde{S}} \tilde{f}(\tilde{x})$$

using $H = I$.

Proof. We show that the following two methods are equivalent. Given $X_j \in S$,

the first method is to generate X_{j+1} using IHR as defined in the text on the original problem (P) with $H = A'A$. The second method is to transform the given X_j into $\tilde{X}_j = AX_j$, perform IHR on the transformed problem $(\tilde{P})$ with $H = I$, and then transform the point $\tilde{X}_{j+1}$ back into the original space, $X'_{j+1} = A^{-1}\tilde{X}_{j+1}$. We now show that X_{j+1} is stochastically equivalent to X'_{j+1}.

Method 2:

Step 1'. Set $\tilde{X}_j = AX_j$. Generate a direction vector $\tilde{D}_j$ from the normal distribution with mean 0 and covariance matrix I.

Step 2'. Generate a step size $\tilde{s}_j$ uniformly from $\tilde{L}_j$, the set of feasible step sizes from the current iteration point $\tilde{X}_j$ in the direction $\tilde{D}_j$, i.e.,

$$\tilde{L}_j = \{\lambda \in \boldsymbol{R} : \tilde{X}_j + \lambda \tilde{D}_j \in \tilde{S}\}.$$

If $\tilde{L}_j = \emptyset$, go to Step 1.

Step 3'. Update the new point if it is improving:

$$\tilde{X}_{j+1} = \begin{cases} \tilde{X}_j + \tilde{s}_j \tilde{D}_j & \text{if } \tilde{f}(\tilde{X}_j + \tilde{s}_j \tilde{D}_j) < \tilde{f}(\tilde{X}_j) \\ \tilde{X}_j & \text{otherwise} \end{cases}$$

and set $\tilde{Y}_{j+1} = \tilde{f}(\tilde{X}_{j+1})$. Set $X'_{j+1} = A^{-1}\tilde{X}_{j+1}$.

First note that AD_j, where D_j is as in Step 1 of IHR, has the same distribution as $\tilde{D}_j$ in Step 1' of Method 2. Using this, we get

$$\begin{aligned} L_j &= \{\lambda \in \boldsymbol{R} : X_j + \lambda D_j \in S\} \\ &= \{\lambda \in \boldsymbol{R} : A^{-1}\tilde{X}_j + \lambda A^{-1}\tilde{D}_j \in S\} \\ &= \{\lambda \in \boldsymbol{R} : A^{-1}(\tilde{X}_j + \lambda \tilde{D}_j) \in S\} \\ &= \{\lambda \in \boldsymbol{R} : \tilde{X}_j + \lambda \tilde{D}_j \in \tilde{S}\} \\ &= \tilde{L}_j . \end{aligned}$$

Thus s_j from Step 2 and $\tilde{s}_j$ from Step 2' are realizations from the same distribution, i.e., $s_j \overset{d}{=} \tilde{s}_j$. Note that

$$\begin{aligned} f(X_j + s_j D_j) &\overset{d}{=} f(A^{-1}\tilde{X}_j + \tilde{s}_j A^{-1}\tilde{D}_j) \\ &= f(A^{-1}(\tilde{X}_j + \tilde{s}_j \tilde{D}_j)) \\ &= \tilde{f}(\tilde{X}_j + \tilde{s}_j \tilde{D}_j) \end{aligned}$$

and hence $\tilde{f}(\tilde{X}_{j+1}) \overset{d}{=} f(X_{j+1})$. Thus the probability of not finding an improvement is the same for both methods, and so if the new point is not improving, then $X'_{j+1} \overset{d}{=} X_{j+1}$. Now look at the distributions of the new iteration points given that an improvement occurs:

$$
\begin{aligned}
X'_{j+1} &= A^{-1}\tilde{X}_{j+1} \\
&= A^{-1}(\tilde{X}_j + \tilde{s}_j\tilde{D}_j) \\
&= A^{-1}\tilde{X}_j + \tilde{s}_j A^{-1}\tilde{D}_j \\
&\overset{d}{=} X_j + s_j D_j \\
&= X_{j+1}
\end{aligned}
$$

and again $X'_{j+1} \overset{d}{=} X_{j+1}$. Thus performing IHR on (P) with $H = A'A$ is equivalent in distribution to the second method of performing IHR on $(\tilde{P})$ with $H = I$. ■

LEMMA 3.2 *For any spherical program* (P), *the conditional probability of improvement on the next sample point for IHR can be expressed as*

$$
\begin{aligned}
P(Y_{j+1} < h(s) \mid Y_j = h(r),\ Y_0 = h(q)) &\equiv p(s;\, r,\, q) \\
&= \left(\frac{r}{q}\right)\left(\frac{s}{r}\right)^n \frac{1}{n} F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{s^2}{q^2}, \frac{s^2}{r^2}\right)
\end{aligned}
$$

for $0 < s \leq r \leq q$, *and for* $j = 0, 1, 2, \ldots,$ *where* $F_1(a,b,c,d;x,y)$ *is a generalized hypergeometric function* (*see* [1] *or* [10]).

Proof. We derive the analytical expression for $P\,(Y_{j+1} < h(s) \mid Y_j = h(r),\ Y_0 = h(q))$ in terms of the radii of the level sets, r. In order to take full advantage of the symmetry of the problem, we shall use a spherical coordinate system $(\rho, \theta, \phi_i,$ $i = 1, \ldots, n-2)$ to compute the integral required to evaluate $P(Y_{j+1} < h(s) \mid Y_j = h(r),\ Y_0 = h(q)) \equiv p(s;r,q)$. We let the origin of the spherical coordinates be at X_j, where the radius of the corresponding level set is r. Let the positive x_1 axis run through the center of the level sets. Figure 1 illustrates a cross section corresponding to fixed ϕ_i, $i = 1, \ldots, n-2$ of a level set of radius r within the feasible region of radius q and containing a level set of radius s. Appendix B includes a brief summary of spherical coordinates.

The integral that we need to evaluate corresponds to equation 1 in the text, only now we use the spherical coordinate system. An important term in the integral is the proportion of the line intersecting the s-sphere to the line intersecting the q-sphere, which we write as $\dfrac{\delta(r,s,\theta)}{\delta(r,q,\theta)}$ (see Figure 1), where $\delta(r,s,\theta)$ is the length of a line segment emanating in a direction θ from a point X_j on a sphere of radius r that intersects with a sphere of radius s. Let θ_0 be the angle that corresponds to the line segment that is tangent to the inner sphere of radius s. Then,

$$\sin\theta_0 = s/r\,.$$

Using spherical coordinates, the equation of the inner sphere of radius s is given by

$$\rho^2 - 2r\rho\cos\theta + r^2 = s^2\,,$$

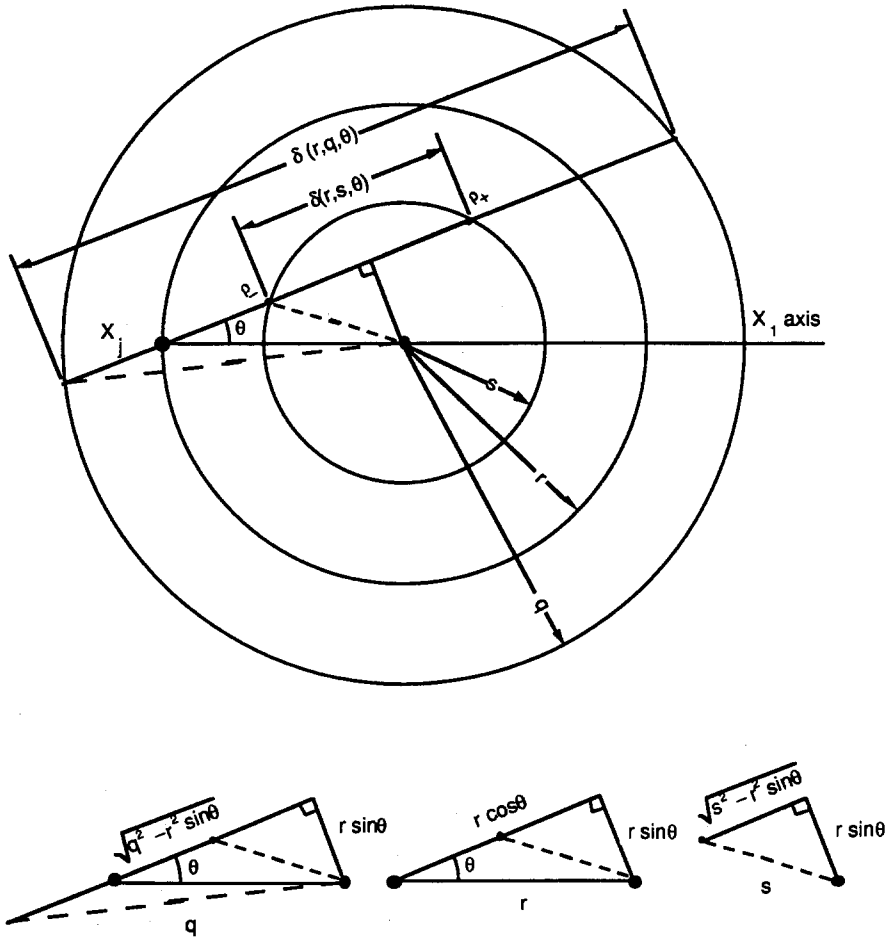

Fig. 1.

or (refer to Figure 1),

$$\rho_{\pm} = r\cos\theta \pm \sqrt{s^2 - r^2\sin^2\theta}$$

for $0 \leq \theta \leq \theta_0$. Now, in the range $0 \leq \theta \leq \theta_0$ the length of the line segment that lies in the sphere of radius s is independent of the ϕ_i because of symmetry, and can be expressed as

$$\delta(r, s, \theta) = \rho_+ - \rho_- = 2\sqrt{s^2 - r^2\sin^2\theta}\,.$$

Similarly we get

$$\delta(r, q, \theta) = 2\sqrt{q^2 - r^2\sin^2\theta}\,.$$

It now follows easily that (refer to Appendix B for details)

$$p(s; r, q) = \frac{2\int_0^{\theta_0}\int_0^{\pi}\cdots\int_0^{\pi}\int_0^{2\pi}\omega(\theta, \phi_1, \ldots, \phi_{n-2})\,\frac{\delta(r, s, \theta)}{\delta(r, q, \theta)}\,d\theta\,d\phi_1\ldots d\phi_{n-2}}{2\int_0^{\pi/2}\int_0^{\pi}\cdots\int_0^{\pi}\int_0^{2\pi}\omega(\theta, \phi_1, \ldots, \phi_{n-2})\,d\theta\,d\phi_1\ldots d\phi_{n-2}}$$

where

$$\omega(\theta, \phi_1, \ldots, \phi_{n-2}) = \sin^{n-2}(\theta)\prod_{k=1}^{n-3}\sin^{n-2-k}(\phi_k)$$

for $n \geq 4$ (as in [26], pp. 227–228), and

$$\frac{\delta(r, s, \theta)}{\delta(r, q, \theta)} = \frac{\sqrt{s^2 - r^2\sin^2\theta}}{\sqrt{q^2 - r^2\sin^2\theta}}.$$

After simplifying, we have

$$p(s; r, q) = \frac{\int_0^{\theta_0}\frac{\sqrt{s^2 - r^2\sin^2\theta}}{\sqrt{q^2 - r^2\sin^2\theta}}\sin^{n-2}(\theta)\,d\theta}{\int_0^{\pi/2}\sin^{n-2}(\theta)\,d\theta}$$

and since for $w, z > 0$ the Beta function can be written as ([1] pp. 258),

$$\frac{1}{2}B(z, w) = \frac{1}{2}\frac{\Gamma(z)\Gamma(w)}{\Gamma(z+w)} = \int_0^{\pi/2}\sin^{2z-1}(t)\cos^{2w-1}(t)\,dt$$

we can write

$$p(s; r, q) = \frac{2}{B\left(\frac{n-1}{2}, \frac{1}{2}\right)}\int_0^{\theta_0}\frac{\sqrt{\frac{s^2}{r^2} - \sin^2\theta}}{\sqrt{\frac{q^2}{r^2} - \sin^2\theta}}\sin^{n-2}(\theta)\,d\theta.$$

It is easily shown that this expression is also valid for $n = 2$ and 3.

We now concentrate on the integral and go through a series of substitutions in order to get the final expression. First we let $a = s^2/r^2$ and $b = q^2/r^2$ to get

$$\int_0^{\theta_0}\frac{\sqrt{\frac{s^2}{r^2} - \sin^2\theta}}{\sqrt{\frac{q^2}{r^2} - \sin^2\theta}}\sin^{n-2}(\theta)\,d\theta = \int_0^{\theta_0}\left(\frac{a - \sin^2\theta}{b - \sin^2\theta}\right)^{1/2}\sin^{n-2}(\theta)\,d\theta$$

and then we change variables using $\sin^2\theta = at$ to get

$$= \int_0^1\left(\frac{a - at}{b - at}\right)^{1/2}\frac{(at)^{\frac{n-2}{2}}\,a\,dt}{2(at)^{1/2}(1 - at)^{1/2}}$$

$$= \frac{a^{\frac{n}{2}}}{2} \int_0^1 \left(\frac{1-t}{(b-at)(1-at)} \right)^{1/2} t^{\frac{n-3}{2}} \, dt$$

$$= \frac{a^{\frac{n-2}{2}}}{2} \int_0^1 \left(\frac{1-t}{(b/a-t)(1/a-t)} \right)^{1/2} t^{\frac{n-3}{2}} \, dt$$

which, using 3.211 of [10] pp. 287, reduces to

$$= \frac{a^{\frac{n}{2}}}{2b^{1/2}} B\left(\frac{3}{2}, \frac{n-1}{2}\right) F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{a}{b}, a\right).$$

Noticing that

$$\frac{B\left(\frac{3}{2}, \frac{n-1}{2}\right)}{B\left(\frac{n-1}{2}, \frac{1}{2}\right)} = \frac{1}{n}$$

we can write

$$p(s; r, q) = \frac{a^{n/2}}{b^{1/2}} \left(\frac{1}{n}\right) F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{a}{b}, a\right)$$

and returning to the original variables

$$= \left(\frac{r}{q}\right)\left(\frac{s}{r}\right)^n \frac{1}{n} F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{s^2}{q^2}, \frac{s^2}{r^2}\right). \quad \blacksquare$$

THEOREM 3.3. *For any spherical program (P), the expected number of IHR sample points needed to achieve an objective function value of $h(r)$ or better is bounded by*

$$E[N(r)] \leq \frac{E[K(r)]}{p(r; r, q)}$$

$$\leq \frac{q}{r} \gamma(n) E[K(r)]$$

for $0 < r \leq q$ and where

$$\gamma(n) = \frac{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}$$

$$= O(\sqrt{n}).$$

Proof. We first express $N(r)$ as a sum,

$$N(r) = \sum_{m=1}^{K(r)} N_m,$$

where N_m is defined to be the number of sample points between R_{m-1} and R_m, for

$m = 1, 2, \ldots$ Thus N_m is the number of sample points between consecutive improving points (including the sample point corresponding to R_m). Now we take the expectation of both sides to get,

$$E[N(r)] = E\left[\sum_{m=1}^{K(r)} N_m\right]$$

$$= E\left[E\left[\sum_{m=1}^{K(r)} N_m \,|\, K(r), R_0, \ldots, R_{K(r)-1}\right]\right].$$

The conditional expectation can be rewritten, for $r_0 > r_1 > \cdots r_{k-1} > r$, and $k \geq 1$:

$$E\left[\sum_{m=1}^{K(r)} N_m \,|\, K(r) = k, R_0 = r_0, \ldots, R_{k-1} = r_{k-1}\right]$$

$$= E\left[\sum_{m=1}^{k} N_m \,|\, K(r) = k, R_0 = r_0, \ldots, R_{k-1} = r_{k-1}\right]$$

$$= E\left[\sum_{m=1}^{k} N_m \,|\, R_0 = r_0, \ldots, R_{k-1} = r_{k-1}, R_k \leq r\right]$$

because the k-th improving point is the first point inside the r-sphere

$$= \sum_{m=1}^{k} E[N_m \,|\, R_0 = r_0, \ldots, R_{k-1} = r_{k-1}, R_k \leq r]$$

by linearity of conditional expectation

$$= \sum_{m=1}^{k} E[N_m \,|\, R_{m-1} = r_{m-1}]$$

since N_m is conditionally independent of R_j $(j \neq m-1)$ when given R_{m-1}

$$= \sum_{m=1}^{k} \frac{1}{p(r_{m-1}; r_{m-1}, q)}$$

because the number of sample points to achieve the m-th improving point, given $R_{m-1} = r_{m-1}$, has a geometric distribution with parameter $p(r_{m-1}; r_{m-1}, q)$

$$\leq \sum_{m=1}^{k} \frac{1}{p(r; r, q)}$$

since $p(r;r,q) \leq p(s;s,q)$ for $s \geq r$

$$= \frac{k}{p(r; r, q)}.$$

We now can show the first statement in the theorem,

$$E[N(r)] \leq E\left[\frac{K(r)}{p(r; r, q)}\right]$$

$$= \frac{E[K(r)]}{p(r; r, q)}.$$

The second statement follows easily from the expression for $p(r;r,q)$ as given in the text in equation (2),

$$E[N(r)] \leq \frac{q}{r}\,\gamma(n) F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; \frac{r^2}{q^2}\right) E[K(r)]$$

where $\gamma(n) = \dfrac{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}$. Noting that $F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; \frac{r^2}{q^2}\right) \leq 1$, we have

$$E[N(r)] \leq \frac{q}{r}\,\gamma(n) E[K(r)]\,.$$

Now for large n, we have

$$\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} \approx \sqrt{n/2}$$

(see [6]), yielding $\gamma(n) = O(\sqrt{n})$. ∎

LEMMA 3.4. *For any spherical program* (P), *the conditional probability for improving points for IHR can be expressed as*

$$P(R_{k+1} < s \mid R_k = r, R_0 = q) = \frac{p(s; r, q)}{p(r; r, q)} \geq \frac{(s/r)^n}{n}$$

for $0 < s \leq r \leq q$, *and for* $k = 0, 1, 2, \ldots$

Proof.

$$\begin{aligned} P(R_{k+1} < s \mid R_k = r, R_0 = q) &= P(Y_{j+1} < h(s) \mid Y_j = h(r), Y_0 = h(q)) \\ &\quad + P(Y_{j+2} < h(s), Y_{j+1} \geq h(r) \mid Y_j = h(r), \\ &\quad Y_0 = h(q)) \quad + \cdots \\ &= p(s; r, q) + p(s; r, q)(1 - p(r; r, q)) + \cdots \\ &= p(s; r, q) \sum_{i=0}^{\infty} (1 - p(r; r, q))^i \\ &= \frac{p(s; r, q)}{p(r; r, q)}\,. \end{aligned}$$

From Lemma 3.2 and equation (2) we have

$$p(s; r, q) = \left(\frac{r}{q}\right)\left(\frac{s}{r}\right)^n \frac{1}{n} F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{s^2}{q^2}, \frac{s^2}{r^2}\right)$$

and

$$p(r;\, r,\, q) = \left(\frac{r}{q}\right) \frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)} F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; \frac{r^2}{q^2}\right).$$

It is easy to see from definitions that $F(a,b;c;z)$ is increasing in z for $z>0$ when $a,b,c>0$. Also, $F_1(a,b,c,d;x,y)$ is increasing in both x and y for $x,\, y>0$ when $a,b,c,d>0$. Thus,

$$F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; \frac{r^2}{q^2}\right) \leqslant F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; 1\right)$$

$$= \frac{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}$$

from [10] 9.122.1 pp. 1042 for $0<r/q\leqslant 1$. Similarly,

$$F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{s^2}{q^2}, \frac{s^2}{r^2}\right) \geqslant F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; 0, 0\right)$$

$$= 1\,.$$

Hence

$$\frac{p(s;\, r,\, q)}{p(r;\, r,\, q)} = \left(\frac{s}{r}\right)^n \frac{1}{n} \frac{F_1\left(\frac{n-1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{n+2}{2}; \frac{s^2}{q^2}, \frac{s^2}{r^2}\right)}{F\left(\frac{1}{2}, \frac{n-1}{2}; \frac{n+1}{2}; \frac{r^2}{q^2}\right)} \frac{\Gamma\left(\frac{n+1}{2}\right)\Gamma\left(\frac{1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}$$

$$\geqslant \frac{(s/r)^n}{n}\,. \qquad \blacksquare$$

THEOREM 3.5. *For any spherical program* (P), *the expected number of IHR improving points needed to achieve an objective function value of* $h(r)$ *or better is bounded by*

$$E[K(r)] \leqslant nE[K(r)^{PAS}]$$

$$= O(n^2)$$

for $0<r\leqslant q$.

Proof. From Lemma 3.4 we have a bound on the conditional probability for improving points for IHR, for $0<s\leqslant r\leqslant q$

$$P(R_{k+1}^{IHR} < s \mid R_k^{IHR} = r) \geqslant \frac{(s/r)^n}{n}\,.$$

Similarly, for pure adaptive search (PAS) we have [28],

$$P(R_{k+1}^{PAS} < s \mid R_k^{PAS} = r) = (s/r)^n .$$

Now we define the following intermediate algorithm, called A.

ALGORITHM A

Step 0. Initialize $X_0 \in S$, and $k = 0$.

Step 1. With probability $1/n$, choose X_{k+1} uniformly from the set

$$\{x : x \in S, \text{ and } f(x) < f(X_k)\} .$$

Otherwise, set $X_{k+1} = X_k$.

Step 2. Increment k, and return to Step 1.

Algorithm A performs a PAS step with probability $1/n$. Thus for $0 < s \leq r \leq q$ we have,

$$P(R_{k+1}^{A} < s \mid R_k^{A} = r) = \frac{(s/r)^n}{n}$$

and further,

$$P(R_{k+1}^{A} < s \mid R_k^{A} = r) \leq P(R_{k+1}^{IHR} < s \mid R_k^{IHR} = r) .$$

We have defined $E[K(r)]$ to be the expected number of improving points needed to achieve $R_k \leq r$, and we now extend the definition for the three algorithms and add a superscript. We now have $E[K(r)^{IHR}]$, $E[K(r)^{PAS}]$, and $E[K(r)^{A}]$. With the above comments, we have the following,

$$\begin{aligned} E[K(r)^{IHR}] &\leq E[K(r)^{A}] \\ &= nE[K(r)^{PAS}] \\ &= O(n^2) . \end{aligned}$$

■

B. SPHERICAL COORDINATES

Here we summarize the spherical coordinates in n dimensions as given in [26], pp. 227–228. We only include those details that are necessary for the proof of Lemma 3.2. Let $x_1, x_2, \ldots, x_n$ denote the Cartesian coordinates and let ρ, θ, $\phi_1, \ldots, \phi_{n-2}$ denote the spherical coordinates. For $n > 3$, the relationships are given by

$$\begin{aligned} x_1 &= \rho \cos\theta \\ x_2 &= \rho \sin\theta \cos\phi_1 \\ x_3 &= \rho \sin\theta \sin\phi_1 \cos\phi_2 \\ &\vdots \\ x_{n-1} &= \rho \sin\theta \sin\phi_1 \sin\phi_2 \cdots \sin\phi_{n-3} \cos\phi_{n-2} \\ x_n &= \rho \sin\theta \sin\phi_1 \sin\phi_2 \cdots \sin\phi_{n-3} \sin\phi_{n-2} . \end{aligned}$$

Note that $\rho = \|x\|$ and that θ is the angle between x and the positive x_1 axis. In order to cover the whole space $-\infty < x_1 < +\infty$ once, we need $0 \leq r < +\infty$; $0 \leq \theta \leq \pi$; $0 \leq \phi_i \leq \pi$ for $i = 1, \ldots, n-3$; and $-\pi \leq \phi_{n-2} \leq \pi$. The surface element of the unit sphere is given by

$$d\omega = \omega(\theta, \phi_1, \ldots, \phi_{n-2})\, d\theta\, d\phi_1 \ldots d\phi_{n-2},$$

where,

$$\omega(\theta, \phi_1, \ldots, \phi_{n-2}) = \sin^{n-2}(\theta) \prod_{k=1}^{n-3} \sin^{n-2-k}(\phi_k)$$

for $n \geq 3$.

References

1. M. Abramowitz and I. A. Stegun, eds. (1961), *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables* (National Bureau of Standards, Applied Mathematics Series 55, June 1964).
2. C. J. P. Bélisle, H. E. Romeijn, and R. L. Smith (1993), Hit-and-Run Algorithms for Generating Multivariate Distributions, to appear in *Mathematics of Operations Research*.
3. C. J. P. Bélisle, H. E. Romeijn, and R. L. Smith (1990), Hide-and-Seek: A Simulated Annealing Algorithm for Global Optimization, Technical Report 90-25, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, September 1990.
4. H. C. P. Berbee, C. G. E. Boender, A. H. G. Rinnooy Kan, C. L. Scheffer, R. L. Smith, and J. Telgen (1987), Hit-and-Run Algorithms for the Identification of Nonredundant Linear Inequalities, *Mathematical Programming* **37** 184–207.
5. A. Boneh (1983), A Probabilistic Algorithm for Identifying Redundancy by a Random Feasible Point Generator (RFPG), in M. H. Karwan, V. Lotfi, J. Telgen, and S. Zionts, eds., *Redundancy in Mathematical Programming* (Springer-Verlag, Berlin).
6. H. Cramér (1946), *Mathematical Methods of Statistics* (Princeton University Press).
7. L. C. W. Dixon and G. P. Szegö, eds. (1975), *Towards Global Optimization* (North-Holland, Amsterdam).
8. L. C. W. Dixon and G. P. Szegö, eds. (1978), *Towards Global Optimization 2* (North-Holland, Amsterdam).
9. W. Feller (1971), *An Introduction to Probabiltiy Theory and Its Applications*, Volume 2, 2nd Edition (John Wiley and Sons, New York).
10. I. S. Gradshteyn and I. M. Ryzhik, eds. (1980), translated by Alan Jeffrey, *Table of Integrals, Series, and Products* (Academic Press, New York).
11. M. H. Karwan, V. Lotfi, J. Telgen, and S. Zionts, eds. (1983), *Redundancy in Mathematical Programming* (Springer-Verlag, Berlin), 108–134.
12. D. E. Knuth (1969), *The Art of Computer Programming*, Vol. 2 (Addison-Wesley, Reading, Massachusetts), p. 116.
13. J. P. Lawrence III and K. Steiglitz (1972), Randomized Pattern Search, *IEEE Transactions On Computers* **C-21**, 382–385.
14. K. G. Murty (1983), *Linear Programming* (John Wiley and Sons, New York).
15. V. A. Mutseniyeks and L. Rastrigin (1964), Extremal Control of Continuous Multi-Parameter Systems by the Method of Random Search, *Engineering Cybernetics* **1**, 82–90.
16. N. R. Patel, R. L. Smith, and Z. B. Zabinsky (1988), Pure Adaptive Search In Monte Carlo Optimization, *Mathematical Programming* **43**, 317–328.
17. L. A. Rastrigin (1960), Extremal Control by the Method of Random Scanning, *Automation and Remote Control* **21**, 891–896.
18. L. A. Rastrigin (1963), The Convergence of the Random Method in the Extremal Control of a Many-Parameter System, *Automation and Remote Control* **24**, 1337–1342.
19. S. M. Ross (1983), *Stochastic Processes* (John Wiley and Sons, New York).

20. L. E. Scales (1985) *Introduction to Non-Linear Optimization* (Macmillan).
21. G. Schrack and N. Borowski (1972), An Experimental Comparison of Three Random Searches, in F. Lootsma, ed., *Numerical Methods for Nonlinear Optimization* (Academic Press, London), pp. 137–147.
22. G. Schrack and M. Choit (1976), Optimized Relative Step Size Random Searches, *Mathematical Programming* **10**, 270–276.
23. M. A. Schumer and K. Steiglitz (1968), Adaptive Step Size Random Search, *IEEE Transactions On Automatic Control* **AC-13**, 270–276.
24. R. L. Smith (1984), Efficient Monte Carlo Procedures for Generating Points Uniformly Distributed over Bounded Regions, *Operations Research* **32**, 1296–1308.
25. F. J. Solis and R. J.-B. Wets (1981), Minimization by Random Search Techniques, *Mathematics of Operations Research* **6**, 19–30.
26. A. Sommerfeld (1949), *Partial Differential Equations in Physics* (Academic Press, New York).
27. Z. B. Zabinsky (1985), *Computational Complexity of Adaptive Algorithms in Monte Carlo Optimization* (Ph.D. Dissertation from The University of Michigan, Ann Arbor MI).
28. Z. B. Zabinsky and R. L. Smith (1992), Pure Adaptive Search in Global Optimization, *Mathematical Programming* **53**, 323–338.
29. Z. B. Zabinsky, D. L. Graesser, M. E. Tuttle, and G. I. Kim (1992), Global Optimization of Composite Laminates Using Improving Hit-and-Run, in C. A. Floudas and P. M. Pardalos, eds., *Recent Advances in Global Optimization*, Princeton University Press.