Random Restarts in Global Optimization

by

X. Hu, R. Shonkwiler, and M. C. Spruill

School of Mathematics

Georgia Institute of Technology

Atlanta, GA 30332

Keywords:

Monte Carlo Methods, Multistart Methods, Markov Chains, Parallelization.

AMS Subject Classification: 65K10.

Abbreviated Title: Random Restarts in Global Optimization

Mail proofs to:
R. Shonkwiler
School of Mathematics
Georgia Institute of Technology
Atlanta, GA 30332

January 18, 1994

Abstract

In this article we study stochastic multistart methods for global optimization, which combine local search with random initialization, and their parallel implementations. It is shown that in a minimax sense the optimal restart distribution is uniform. We further establish the rate of decrease of the ensemble probability that the global minimum has not been found by the n^{th} iteration. Turning to parallelization issues, we show that under independent identical processing (iip), exponential speedup in the time to hit the goal bin normally results. Our numerical studies are in close agreement with these findings.

1. Introduction

In this paper random restart methods for finding a global optimum are analyzed. Let f be a real valued function on the domain D. In the event that f is sufficiently smooth, powerful methods can be employed for identifying local minimizers (or maximizers). Often these methods utilize local derivative or gradient information to identify a sequence of points $x_0, x_1, \ldots \to x^*$ on which the function values $v_i = f(x_i)$ decrease until a local minimum is determined within some tolerance. If f has numerous local minima, the problem of locating a global minimizer can be difficult and success depends on the choice of a starting point x_0. The methods encompassed in this study combine the "gradient" algorithm G with random initialization; the search proceeds by selecting at random an initial starting point x, running the algorithm until a local minimizer $G(x)$ has been found, and repeating this process independently and with the same probability distribution for selecting the starting point until meeting some specified conditions for stopping.

Features of these problems investigated here include efficacy of parallel processing for random restart methods, choice of the restart probability distributions, and rate of decrease of the ensemble probabilities of having not found the minimizer by termination. Analysis proceeds on two fronts: discrete and continuous models. Our findings can be summarized as follows. First, the uniform probability measure is good for restarting. When it is used, the rate of decrease of the ensemble probabilities of not having found the minimum by the time of termination, depend upon the probable "size" of the region containing the global minimum. This rate is exponential when the size is bounded below. Second, the "size" of the region containing the global minimum also influences the speedup available by parallel processing in terms of numbers of restarts. Speedup is an exponentially increasing function in the number of processors when the region containing the minimum is small and the number of processors is moderate. Third, analysis in terms of number of restarts is useful also as an indicator of the time till minimization when the amount of time taken to run the algorithm is approximately constant for each starting point. When times are variable and the number of possible states at each restart is large in comparison with the number of local minima of the function being minimized, the speedup is always an exponentially increasing function in the number of processors for moderate numbers of processors. Fourth, speedup can be superlinear even for large numbers of processors.

A recent survey of literature on global stochastic optimization can be found in Schoen, (1990). What we have called random restart methods are included there as the simplest instances of what Schoen calls multistart methods. Solis and Wets, (1981) study convergence for more complicated random restart methods in which the probability distribution for choosing the next starting point can depend on the evolution of the search. Boender and Rinnooy Kan, (1987) study stopping rules. Our studies do not address these issues. Törn and Zelinskas, (1989) present in their survey some numerical results on parallelization for a collection of methods and end by recommending future study of Monte Carlo and geometric rather than algorithmic parallelization. Our investigation of random restarts is in one such recommended area, multiple CPU Monte Carlo parallelization, and continues the work of Shonkwiler and Van Vleck, (1992) using the same techniques.

To describe details of our findings some terminology is required. Let $\mathbb{F}$ be a set of functions any member of which one may potentially be called upon to minimize, all defined on a common domain D and each having a global minimum in D. The algorithm G, which to every $f \in \mathbb{F}$ and $x \in D$ yields a point $G(f,x)$ of D, is assumed to satisfy

$$\text{(i)} \qquad f(G(f,x)) \leq f(x)$$

and, fixing f,

$$\text{(ii)} \qquad G^2(x) = G(x).$$

For a fixed f, the relation r on $D \times D$ defined by xry if $G(x) = G(y)$ is an equivalence relation so that there is a partition $\mathbb{B}(f)$ of D into subsets B with the property that x and

y are in $B \in \mathbb{B}(f)$ if and only if $G(x) = G(y)$. The properties (i)–(ii) ensure that G maps each basin B into itself.

The union $M(f)$ of these sets B satisfying $f(G(x)) = \min_{y \in D} f(y)$ for $x \in B$ will be called the *min–bin*. The random variable E defined as the number of the restart on which the initial point of the algorithm is first chosen in M shall be known as the minimization epoch. For example, if the starting point is chosen in the min–bin then $E = 0$, if the starting point is in the complement M^c but the restarting point is in M then $E = 1$ and so forth. Let T denote the time at which the initial point of the algorithm is first chosen in M.

Detailed proofs of the results which follow can be found in the appendix.

2. Choice of a Probability Measure for Restart

In this section arguments in favor of using the uniform measure on the state space are presented. The sense in which this is a good measure is this; if a random restart method is to be used to find a global minimum of a function f, and if this function is unknown (pointwise evaluations can be made of f and possibly some of its derivatives or other functionals, but the general behavior of f on D is not known) and could be any one from a sufficiently large collection of functions, then by using a measure other than the uniform the worst case is worse than for a uniform. Let D be a finite set and suppose that members of the family of functions $\mathbb{F}$ are all defined on D. The measure of performance of the restart distribution $\mathbb{P}$ is

$$\mathbb{E}[E(f)] = \frac{1 - \theta(f, \mathbb{P})}{\theta(f, \mathbb{P})},$$

where $\theta = \mathbb{P}[M(f)]$ is the probability assigned to the min bin of f by the restart distribution $\mathbb{P}$. If the family $\mathbb{F}$ is sufficiently large then the uniform measure will minimize the maximum of these values over the functions in $\mathbb{F}$. One condition which guarantees the family is sufficiently large is the following.

Condition (A). For each $x \in D$, $\mathbb{F}$ contains a function which achieves its minimum uniquely at x.

Lemma 2.1. *If $\mathbb{F}$ satisfies (A) then, letting subscript* 0 *refer to the uniform distribution,*

$$\inf_{\mathbb{P}} \sup_{f\in\mathbb{F}} \mathbb{E}[E(f)] = \sup_{f\in\mathbb{F}} \mathbb{E}_0[E(f)]$$

Proof. . Since

$$\sup_{f\in\mathbb{F}} \mathbb{E}_0[E(f)] = \sup_{M(f)} \frac{(1-\mathbb{P}_0[M(f)])}{\mathbb{P}_0[M(f)]},$$

the latter is just

$$\sup_{M(f)} \frac{(1-n(M(f)))/n(D)}{n(M(f))/n(D)},$$

and the function $\frac{(1-x)}{x}$ is continuous and decreasing on $[a,1]$ for any $a > 0$, the left hand side is $n(D)-1$ by condition (A). On the other hand, for any other measure $\mathbb{P}$, $\sup_{f\in\mathbb{F}} \mathbb{E}[E(f)] = \sup_{M(f)} \frac{1-\mathbb{P}[M(f)]}{\mathbb{P}[M(f)]} = (\min_{x\in D}\mathbb{P}(x))^{-1} - 1 > n(D) - 1$.

The condition (A) is not very restrictive. For example, in the case of a discretized continuum, if the collection $\mathbb{F}$ contains the functions

$$\left\{ f(x) = \sum_{i=1}^{k} a_i(x_i - c_i)^2 : \mathbf{a} \in R^k, \mathbf{c} \in D \right\},$$

then condition (A) is satisfied. In a TSP with k nodes in the plane, for example, D consists of all possible tours and the functions $f \in \mathbb{F}$ have values which depend on the locations of the nodes. Clearly, by appropriate placement of the nodes, any tour can be made to be the unique minimum.

Throughout the remainder of the paper, including the section in which D is modeled as a continuum, it is assumed that restarting is always done using the uniform measure. Furthermore, it is done independently of all other information so that the restart sequence forms an iid sequence of D-valued random variables.

3. Analysis of Some Discrete Models

In this section efficacy of running parallel processors at a fixed but arbitrary $f \in \mathbb{F}$ is investigated for some models based on a finite discrete set D. We study the benefits, measured by the decrease in expected time to hit the goal, of employing m independent identical processors, each running the same local algorithm in a multistart way. The speed at which the global optimum is found is of course influenced by the particular local algorithm employed; but it is not this aspect which is studied here. In our study the algorithm is fixed and can be any one in a broad class of algorithms. It is the benefits of parallelization alone which we attempt here to assess.

Since any one of the processors finding the global minimum results in success, it is clear because of independence that for any algorithm the expected time to attain the global minimum will be decreased by increasing the number of processors. It is, however, not clear exactly what quantification of this decrease is most illuminating (see Bertsekis and Tsitsiklis, (1989) for some related issues in defining speedup). For this reason we begin this section by a brief discussion of our choice of the measure of speedup and its relationship to some other possible choices.

If first we allow ourselves some license to capture the idea without paying careful attention to details, a simple characterization of our choice of measure is that it measures the decrease in expected time to turn a local algorithm into a global algorithm as a function of the number of processors employed. The reason we have chosen our measure is two-fold; the measure is natural and closely related to the traditional comparison in terms of time to goal and is more easily computed than the latter.

To be specific, our discussion begins with the simplest discrete model, the atomic model employed implicitly in section 2 in which the only quantity analyzed is the epoch of minimization. Setting $\theta = \mathbb{P}_0[M(f)]$, the probability of starting in the goal bin, if X is the random hitting time of the global minimum using a single processor, X_m is that for m independent processors, E is the random hitting epoch of the min bin for a single and E_m that for m independent processors then in this simple model $X = E + 1$, $X_m = E_m + 1$

and denoting by SGB (speedup to hit the goal bin) our measure of speedup is

$$SGB = \frac{\mathbb{E}(E)}{\mathbb{E}(E_m)} = \frac{1-(1-\theta)^m}{\theta(1-\theta)^{m-1}},$$

see (3.1). The more traditional measure of speedup in terms of the hitting times X is

$$SG = \frac{\mathbb{E}(X)}{\mathbb{E}(X_m)} = \frac{1}{\theta}(1-(1-\theta)^m).$$

We note first that the two measures are equivalent since one is a monotonic function of the other. This can be seen easily from the expression of yet another reasonable measure, the relative improvement of the expected hitting times

$$R_m = \frac{\mathbb{E}(X)-\mathbb{E}(X_m)}{\mathbb{E}(X)},$$

for

$$R_m = 1 - SG^{-1} = \frac{\mathbb{E}(E)-\mathbb{E}(E_m)}{\mathbb{E}(X)} = \frac{\mathbb{E}(E)}{\mathbb{E}(E)+1}\frac{\mathbb{E}(E)-\mathbb{E}(E_m)}{\mathbb{E}(E)} = c(1-SGB^{-1}).$$

All of these measures are equivalent. Incidentally, the constant c will be close to 1 in non-trivial problems.

Second we notice that exponential speedup, a phrase which reoccurs in this paper, is always inappropriate to the traditional measure SG; no matter how quickly the parallelization works to get the process into the goal bin, there is always a last step so that $X_m \geq 1$. Thus SG is a bounded function of m with $SG \leq \mathbb{E}[X]$. In contrast, SGB can be exponential (see the formulas above) in m.

Turning to another facet, the ease of computation, we illustrate by citing the multi-step discrete model realizing that the same issues apply in the continuum model. In this model T is the random time to hit the goal bin so if X is as above and T_m and X_m are the minimum times as above then

$$SGB = \frac{\mathbb{E}(T)}{\mathbb{E}(T_m)}$$

while the traditional is as above

$$SG = \frac{\mathbb{E}(X)}{\mathbb{E}(X_m)}.$$

In this model $X = T + Y$, where Y is a random variable. However there is now no simple relationship between $X_m$and T_m since one processor may reach the goal bin before all others but some other may enter the goal bin at a later time ahead of all earlier arrivals in line to the goal. It is still true that $\mathbb{E}[X_m] \le \mathbb{E}[T_m] + \mathbb{E}[Y]$ so we have the bounds

$$\frac{\mathbb{E}(T)+\mathbb{E}(Y)}{\mathbb{E}(T_m)} \ge \frac{\mathbb{E}(X)}{\mathbb{E}(X_m)} \ge \frac{\mathbb{E}(T)+\mathbb{E}(Y)}{\mathbb{E}(T_m)+\mathbb{E}(Y)}.$$

It follows that

$$SGB^{-1}c + K \ge SG^{-1} \ge SGB^{-1}c,$$

where $c = (1+\frac{\mathbb{E}(Y)}{\mathbb{E}(T)})^{-1}$ and $K = \frac{\mathbb{E}(Y)}{\mathbb{E}(Y)+\mathbb{E}(T)}$. One would expect in any non-trivial problem that K is close to 0 and c is close to 1.

Atomic Model

The simplest possible discrete model is the "atomic" model employed implicitly in Section 2. In this analysis the only quantity analyzed is the epoch of minimization. Setting $\theta = \mathbb{P}_0[M(f)]$, denoting the number of processors by m, and the epoch at which the first of the independently running processors restarts in the min–bin by E_m, elementary calculations yield

$$\text{speedup} = \frac{\mathbb{E}[E]}{\mathbb{E}[E_m]} = \frac{1-(1-\theta)^m}{\theta(1-\theta)^{m-1}}. \tag{3.1}$$

Some numerical values of the speedup of parallel processing follow.

Atomic Speedup

$m\backslash\theta$	0.001	0.01	0.1	0.2
10	10.05	10.47	16.80	33.25
50	51.25	64.63	1737.29	2.80×10^5
100	105.12	171.47	3.39×10^5	19.6×10^9
1000	1717.92	2.29×10^6	-	-

Table 1

If it is true that $T = (E+1)k$, where k is the iteration number or time, then these figures are indicative of the savings in time due to employing m independent processors. In this case, it has been implicitly assumed either that the process of restarting takes a minuscule time compared to the time k taken to run the algorithm G or that all the processors are started simultaneously and execute according to the same schedule. The numbers are especially intriguing in the context of the latter case. Is the apparently enormous speedup essentially wasted because many processors are finding the min–bin on the start? When $\theta = .01$ and $m = 50$, for example, the number to start in the min–bin has approximately a Poisson distribution with mean 1/2 so the probability that at least one starts there is only 0.4. It appears that in this case the answer is no. Other cases are given in the following table.

Probability of at Least One Start in the Min–bin

$m \backslash \theta$	0.001	0.01	0.1	0.2
10	0.01	0.10	0.63	0.86
50	0.05	0.39	0.99	1.00
100	0.10	0.63	1.00	
1000	0.63	1.00		

Table 2

Multi-step Discrete Model

In the next portion of this section we shall be concerned not only with the epoch E but also with the total time taken to search the individual basins encountered up to that point. For that purpose we introduce the "multi-step" model. The function f creates a partition $\mathbb{B}(f)$ consisting of b basins. We shall also assume that in each basin there is a unique minimum and that a single basin, B_1, contains the global minimum. Denote the states in bin i by (i, j), $1 \leq j \leq n(i)$, where $(i, 1)$ is the local minimum in basin i and $n(i)$ is the number of such states, We shall arrange the states of the non-goal bins

$(2,1),\dots(2,n(2)),(3,1),\dots,(b,n(b))$ in such a way that $n(2)\ge n(3)\ge\cdots\ge n(b)$. Using the uniform measure, the state–transition matrix for a single processor is $P=[A_{ij}]$, $i=1,\dots,b$, $j=1,\dots,b$, where A_{ii} is the $n(i)\times n(i)$ sub–matrix

$$A_{ii}=\begin{pmatrix} \frac{1}{N} & \frac{1}{N} & \cdots & \frac{1}{N} & \frac{1}{N} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ & & \cdots & & \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}$$

for $i=1,\dots,b$ and the $n(i)\times n(j)$ sub–matrix A_{ij} for $i\neq j$ is given by

$$A_{ij}=\begin{pmatrix} \frac{1}{N} & \frac{1}{N} & \cdots & \frac{1}{N} & \frac{1}{N} \\ 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ & & \cdots & & \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

Here $N=\sum_{i=1}^{b}n(i)$ and in the notation above, $\theta=n(1)/N$.

Shonkwiler and Van Vleck, (1992) analyze

$$\text{speedup}=\frac{\mathbb{E}[T]}{\mathbb{E}[T_m]},$$

where $\mathbb{E}[T_m]$ is the expected minimum time until the first of m independent processors restarts in the min–bin. In their analysis the deleted transition matrix, $\hat{P}$, obtained by deletion of the first $n(1)$ rows and columns of P is introduced and plays a key role. They prove that the speedup is given by

$$\text{speedup}=s^{m-1}\frac{1-\lambda^m}{1-\lambda}+O(1-\lambda^m) \tag{3.2}$$

as $\lambda\to1$. In this, λ is the Perron–Frobenius eigenvalue of the deleted transition matrix $\hat{P}$, ω is the corresponding left eigenvector normalized so that $\omega'1=1$, χ is the corresponding right eigenvector normalized so that $\omega'\chi=1$, and

$$s^{-1}=\hat{\alpha}'\chi,$$

where $\hat{\alpha}$ is the common deleted initial probability starting vector of all processors. In our case this is the $(N - n(1)) \times 1$ column vector $\hat{\alpha}' = \frac{1}{N}1'$. As an example, for the atomic model in the first part of this section, it can be shown that $\lambda = 1 - \theta$ and $s^{-1} = 1 - \theta$ so that one has exactly

$$\text{speedup} = s^{m-1}\frac{1-\lambda^m}{1-\lambda}.$$

We consider next the analysis of the Perron–Frobenius eigenvalue of $\hat{P}$ in the multi–step model.

Theorem 3.1. *The Perron–Frobenius eigenvalue of $\hat{P}$ is λ, where $\gamma = \lambda^{-1}$ is the unique solution larger than 1 to $f(\gamma) = 0$ for f given by*

$$f(\eta) = \frac{1}{N}\sum_{i=2}^{b}\eta^{n(i)+1} - \left(\frac{1}{N}(b-1)+1\right)\eta + 1. \tag{3.3}$$

The associated right eigenvectors are scalar multiples of

$$v_0' = (1, \gamma, \gamma^2, \ldots, \gamma^{n(2)-1}, 1, \gamma, \gamma^2, \ldots, \gamma^{n(3)-1}, 1, \ldots, \gamma^{n(b)-1}),$$

and the associated left eigenvectors are scalar multiples of

$$\begin{aligned} w' = (&1-\gamma^{n(2)}, 1-\gamma^{n(2)-1}, \ldots, 1-\gamma^{n(2)-2}, \ldots, 1-\gamma \\ &1-\gamma^{n(3)}, 1-\gamma^{n(3)-1}, \ldots, 1-\gamma, \ldots, 1-\gamma^{n(b)}, \ldots, 1-\gamma) \end{aligned}$$

The next theorem shows that in the multistep model, $s > 1$ always.

Theorem 3.2. *For the multi–step model,*

$$s = \frac{\left(\sum_{j=2}^{b} n(j)\lambda^{-(n(j)+1)} - N\right)}{n(1)} \geq \lambda^{-1} > 1$$

where λ is the Perron–Frobenius eigenvalue.

The observation that $s > 1$ in this model indicates that speedup is approximately an exponentially increasing function in m (for small to moderate m) when N is large or θ is small. This can be seen from (3.2) and an examination of the row sums of $\hat{P}$. Indeed, using familiar facts about the Perron–Frobenius eigenvalue it follows that $\lambda > 1 - \theta$. It is also true that $\lambda \to 1$ as $N \to \infty$ even if θ is not close to 1 as can be seen from the fact that $\gamma = \lambda^{-1}$ solves $f(\gamma) = 0$ where f is defined in (3.3).

How general is this model? One feature assumed at the outset is that there is a unique basin containing the global minimum, but this assumption may be relaxed and the results still apply to the more general case if $n(1)$ is replaced by the appropriate sum of the number of states in the min–bin. Another feature of the multistep formulation restricts its generality. It is assumed implicitly by the structure of the transition matrix of the Markov chain that the basins consist of single paths of varying lengths to local minima and the basins are disjoint. In many problems there will be more than one path to a local minimum so these features are inconsistent. However, if there are relatively few local minima in comparison to the total number N of possible starting points then the calculations made above assuming disjoint bins should be indicative of parallel speedup.

In the next section we investigate a continuous time model which does not force this path structure on the basins and allows a separation of basin size and time taken to search the basin.

4. A Continuum Model

In this section an attempt is made to shed further light on the efficacy of parallel processing by modeling the search times as continuous random variables. Generally, super–linear speedup in the number of processors m results when basins can have "long" search times.

Even though the discrete multi–step model is rather general, as noted above, multiple paths leading to the settling point of a basin complicate the construction and analysis of the transition matrix. Moreover that approach does not allow an easy quantitative insight into one important aspect of the problem of efficacy of parallel processing, by which we mean the influence of the distribution of the time taken to search non–goal basins. As in §1 let $\mathbb{B}(f) = \{B_1, \ldots, B_b\}$ denote the basins of f. Let Y_j denote the random time till termination given that the initial point is in the j^{th} basin, that is Y_j is the time to search the j^{th} basin. One would expect that if long search times Y_j occur with reasonably large probabilities, then the benefits of using $m > 1$ processors should increase more rapidly than if all the non–goal basins were rapidly searched. Hence we introduce the continuum model.

The Case of a Single Processor

Let θ_i, $i \geq 1$, be the probability that a starting point in the i^{th} basin is chosen when the starting points are chosen according to a uniform distribution on the space D to be searched. By the length of the j^{th} epoch we shall mean the time Y_j spent in searching the j^{th} chosen basin. As before, let E denote the number of the restart at which an initial point is first chosen in the min–bin. One observes that

$$\mathbb{P}[E = n] = (1-\theta)^n \theta$$

for $n = 0, 1, 2, \ldots$, where we assume without loss of generality $\theta_1 = \theta = \mathbb{P}_0[M(f)]$. Also, if T is the random time spent until the first start in the min–bin, then

$$\mathbb{E}(T) = \sum_{n \geq 0} \mathbb{E}[T|E = n]\mathbb{P}[E = n].$$

Now letting R_j denote the random time spent searching during the j^{th} epoch one has for $n \geq 1$

$$\mathbb{E}[T|E = n] = \mathbb{E}\left[\sum_{j=1}^{n} R_j | E = n\right] = \sum_{j=1}^{n}\sum_{i=2}^{b} \mathbb{E}[Y_i]\frac{\theta_i}{(1-\theta)},$$

since the conditional distribution of R given the basin is not the first is $\mathbb{P}[R \leq t|$ bin 1 not chosen$] = \sum_{i=2}^{b} \mathbb{P}[Y_i \leq t]\mathbb{P}[$ bin i is chosen$|i \neq 1] =$

$$\sum_{i=2}^{b} \mathbb{P}[Y_i \leq t] \frac{\theta_i}{(1-\theta)}.$$

Therefore

$$\mathbb{E}(T) = \sum_{n \geq 1} n \sum_{i=2}^{b} \mathbb{E}[Y_i] \frac{\theta_i}{(1-\theta)} (1-\theta)^n \theta = \sum_{i=2}^{b} \mathbb{E}[Y_i] \frac{\theta_i}{(1-\theta)} \frac{1-\theta}{\theta}.$$

So

$$\mathbb{E}(T) = \frac{\sum_{i=2}^{b} \mu_i \theta_i}{\theta},$$

where $\mu_i = \mathbb{E}[Y_i]$.

The Case of m Processors

In the case of m processors let $U_1 < U_2 < \ldots$ denote the times at which at least one of the m processors randomly restarts. We shall assume that the search times are continuous random variables so that the probability that more than one processor restarts at the same instant is zero. Letting E be as above, the time for at least one of the m processors to find the min–bin has the same probability distribution as U_E where $U_0 = 0$ and

$$U_E = U_E - U_{E-1} + \cdots + U_2 - U_1 + U_1 - U_0 + U_0.$$

Noting that

$$\mathbb{E}[U_{j+1} - U_j] \leq \mathbb{E}\left[\min_{1 \leq i \leq m} R_i\right]$$

it follows that

$$\mathbb{E}[U_E] \leq \sum_{n \geq 1} n \mathbb{E}\left[\min_{1 \leq i \leq m} R_i\right] (1-\theta)^n \theta = \mathbb{E}\left[\min_{1 \leq i \leq m} R_i\right] (1-\theta)/\theta.$$

With T_m be the random minimum time taken for m processors to first start in the min–bin, we have

$$\mathbb{E}[T_m] \leq \mathbb{E}\left[\min_{1 \leq i \leq m} R_i\right] (1-\theta)/\theta.$$

Generally the explicit computation of $E[\min_{1\leq i\leq m} R_i]$ is impossible. It is however possible to come to some conclusions regarding the role of multiprocessing in speedup for m large. Consulting the results available concerning the convergence of normalized minima from general probability distributions and the convergence of their expected values, the two results we shall need are as follows.

The first, Theorem 2.1.5 from Galambos, (1978), states that if there is a constant $\gamma > 0$ such that for all $x > 0$

$$\lim_{t\to-\infty} \frac{F^*(tx)}{F^*(t)} = x^{-\gamma}$$

where $F^*(x) = F(-1/x)$ for $x < 0$, then, setting $M_n = \min_{1\leq i\leq n} R_i$, for all $x > 0$

$$\lim_{n\to\infty} \mathbb{P}[M_n/d_n \leq x] = 1 - e^{-x^\gamma},$$

where

$$d_n = F^{-1}(1/n).$$

The other is due to Pickands, (1968) who proves that under the conditions above,

$$\mathbb{E}[M_n]/d_n \to \int_0^\infty \gamma x^\gamma e^{-x\gamma} dx = \Gamma(1 + 1/\gamma)$$

Where Γ is the gamma function.

Example 4.1. (Alternative to the multi-step). Suppose the search time Y_i of the i^{th} basin is uniform and proportional to the size of the bin so that for all $i = 2, \ldots, b$

$$F_i(tN\theta_i) = P[Y_i \leq tN\theta_i] = \begin{cases} 0 & \text{if } t \leq 0 \\ t & \text{if } 0 < t < 1 \\ 1 & \text{if } t \geq 1 \end{cases}$$

One can verify that $\gamma = 1$ for this problem and $d_n = F^{-1}(1/n) = N\theta/(b-1)n$, so $(b-1)n\mathbb{E}[M_n]/N\theta \to \Gamma(1+1) = 1! = 1$.

The speedup for m processors is

$$\frac{\mathbb{E}[T]}{\mathbb{E}[T_m]} \geq \left(\frac{\sum_{i=2}^{b} \mu_i \theta_i}{\theta}\right) \Big/ \left(\mathbb{E}\left[\min_{1\leq i\leq m} R_i\right](1-\theta)/\theta\right)$$
$$= \left(\sum_{i=2}^{b} N\theta_i^2\right) \Big/ \left(2(1-\theta)\mathbb{E}[M_m]\right).$$

Now let $\epsilon \in (0,1)$ be given and m be so large that $\mathbb{E}[M_m] < (1+\epsilon)N\theta/(b-1)m$. Then

$$\frac{\mathbb{E}[T]}{\mathbb{E}[T_m]} \geq m(b-1)\left(\sum_{i=2}^{b} \theta_i^2\right) \Big/ (2(1+\epsilon)\theta(1-\theta)).$$

The speedup is therefore at least linear in m, the number of processors, for m large.

Example 4.2. Suppose $\mathbb{P}[Y_i > t] = \exp\{-\alpha_i t^{\beta} i\}$, $1 \leq i \leq b$, so that the Y's are Weibull random variables. An analysis like the one in Example 1 shows that $\gamma = \min_{1\leq i\leq b} \beta_i$ and that for the choice of

$$d_n = \left(\frac{\Theta}{n\sum_{i\in J}\theta_i\alpha_i}\right)^{1/\gamma},$$

where $\Theta = \sum_{i\in J}\theta_i$ and $J = \{i \in \{2,3,\ldots,b\} : \beta_i = \gamma\}$, one has

$$\mathbb{E}[M_n]/d_n \to \Gamma(1+1/\gamma).$$

It follows that for m sufficiently large

$$\frac{\mathbb{E}[T]}{\mathbb{E}[T_m]} \geq \frac{\left(\sum_{j=2}^{b}\mu_j\theta_j\right)}{(1-\theta)\Gamma(1+1/\gamma)(1+\epsilon)}\left(\frac{\sum_{i\in J}\theta_i\alpha_i}{\Theta}\right)^{1/\gamma} m^{1/\gamma}.$$

The means the μ_i are $\mu_i = \alpha^{-1/\beta_i}\Gamma(1+1/\beta_i)$. Hence speedup can be superlinear.

5. Ensemble Analysis

We close with an analysis of the ensemble probability of having not found the global minimum by the n^{th} epoch in problems with the property that, even if the global minimum has been found by termination of the procedure, it is not possible to recognize it as such. In this analysis we return to the "atomic" viewpoint used in §3.

Denoting as usual $\theta = \mathbb{P}_0[M(f)]$, in the absence of knowledge of the particular f which the method will be called upon to minimize, the parameter θ can be regarded as a random variable with a prior probability distribution π. Some properties of the ensemble probabilities of finding the minimum of f by the n^{th} epoch as a function of the prior distribution π on θ are studied here.

First, two examples.

Example 5.1. Suppose θ has the prior Beta distribution on (0,1) whose density is ($a > 0, b > 0$)

$$\pi(\theta) = \begin{cases} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1} & \text{if } \theta \in (0,1) \\ 0 & \text{otherwise} \end{cases}$$

The conditional probability that the minimum has not been found by the end of epoch $n-1$ is $\mathbb{P}[F_n^c|\theta] = (1-\theta)^n$ so the probability of not finding the minimum by the end of that epoch is

$$\int (1-\theta)^n \pi(\theta) d\theta = \frac{\Gamma(\alpha+\beta)\Gamma(n+\beta)}{\Gamma(n+\alpha+\beta)\Gamma(\beta)}.$$

Using Stirling's approximation, this is asymptotic to

$$\frac{\Gamma(\alpha+\beta)}{n^{\alpha}\Gamma(\beta)}.$$

In particular, for the uniform measure on θ the probability $\mathbb{P}[F_n^c]$ that the minimum has not been found by the end of epoch $n-1$ is $1/(n+1)$.

Example 5.2. Suppose the prior density on θ is uniform on the interval $(a, 1)$. The probability that the minimum has not been found by the end of epoch $n-1$ is

$$\begin{aligned} \mathbb{P}[F_n^c] &= \int (1-\theta)^n \pi(\theta) d\theta = \int_a^1 (1-\theta)^n d\theta/(1-a) \\ &= (-(1-\theta)^{n+1}/(n+1)(1-a))|_a^1 = (1-a)^n/(n+1). \end{aligned}$$

The probability that the minimum has not been found by the end of epoch $n-1$ is therefore approaching zero geometrically.

In the two examples, the rate of convergence of the ensemble probability of having not yet found the minimum depended only upon the form of the prior density around 0. This is as one would guess; if the functions in $\mathbb{F}$ can have arbitrarily small values of θ then it is to be expected that the location of the minimum by the methods studied here would require large numbers of restarts, and the probability of requiring these large numbers would depend upon the likelihood of such a function being the one to be minimized. The following two lemmas give precise expression to this behavior for arbitrary prior densities π.

Lemma 5.1. *If $\pi(\theta) = 0$ on $(0, a)$ then $\mathbb{P}[F_n^c] = O((1-a)^n)$. Thus, if it is not possible to have the global minimum located in a region of arbitrarily small measure then the rate of convergence to zero of the probability of not finding the minimum by the n^{th} iteration is decreasing to zero geometrically .*

In the following let $\alpha \geq 0$.

Lemma 5.2. *If $\liminf_{\theta \to 0} \pi(\theta)/\theta^\alpha \geq L > 0$ then $\liminf_{n \to \infty} \mathbb{P}[F_n^c] n^{\alpha+1} > 0$.*

Lemma 5.3. *If $\limsup_{\theta \to 0} \pi(\theta)/\theta^\alpha \leq L < \infty$ then for $\eta \in (0, 1)$, $\limsup_{n \to \infty} \mathbb{P}[F_n^c] n^{\alpha+1-\eta} = 0$.*

Lemmas 5.2 and 5.3 imply that if the prior puts mass close to zero in the manner described by

$$|\pi(\theta)/\theta^\alpha - L| \to 0$$

as $\theta \to 0$, then the rate of convergence of the probability of having not found the minimum by the n^{th} iteration can not be geometric; it converges to zero at least as fast as $n^{-(1+\alpha-\eta)}$ for any $\eta > 0$ but no faster than $n^{-(1+\alpha)}$.

The results in the case of parallel processing are easily obtained; for m independent processors replace the n in the formulas above by mn.

6. Numerical Results

In this section the results of some simulations are compared with theoretical calculations based on the models. We have chosen to analyze two one-dimensional problems. The first, test function (C) shown in fig. 1, is given by

$$f(x) = 0.1x + 1 - \cos(\frac{60x}{30+x}), \qquad 0 \leq x < 27.$$

The global minimum for this function is 0 and occurs at $x = 0$. The min bin is the smallest of the basins.

Test Function C

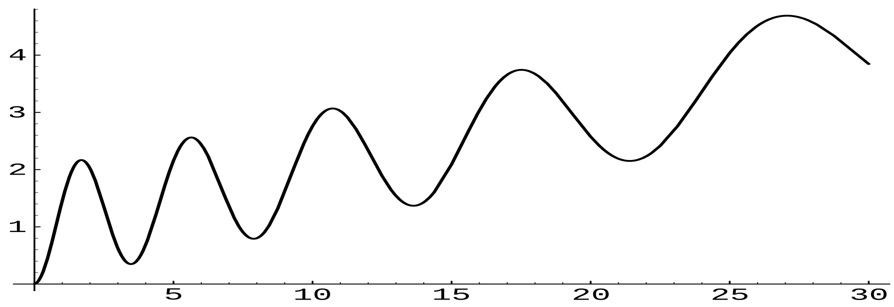

fig. 1

Our second problem, test function (S) shown in fig. 2, is an example of a Shekel function and is the function f_6 in (Törn,p177). The function is given by

$$f(x) = -\sum_{i=1}^{10} \frac{1}{(k_i(x-a_i))^2 + c_i}, \qquad 0 \leq x \leq 10,$$

where the parameters a_i, c_i, and k_i are given in Table 3.

Test Function S

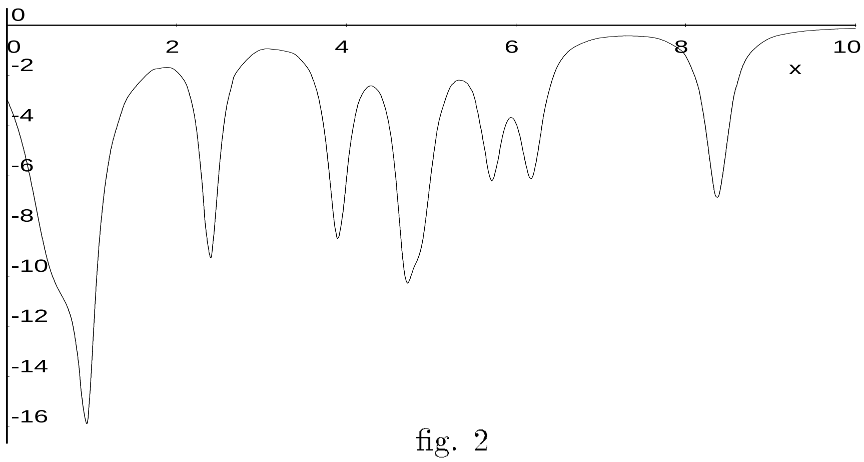

fig. 2

Shekel Function Parameters

a	k	c
4.696	2.871	0.149
4.885	2.328	0.166
0.800	1.111	0.175
.4986	1.263	0.183
3.901	2.399	0.128
2.395	2.629	0.117
0.945	2.853	0.115
8.371	2.344	0.148
6.181	2.592	0.188
5.713	2.929	0.198

Table 3

In this problem the global minimum is -15.875 and occurs at $x = 0.933$. This time the min bin is the largest.

6.1 Procedure for Generating and Analyzing the Numerical Data.

For the gradient algorithm G in these problems we used Newton's Method for Unconstrained Minimization (see Dennis and Schnabel, (1983)) with the modification that no Newton's step was allowed to exceed a pre-calculated maximum, MAXSTEP. This value was based on the reciprocal of the maximum curvature of the graph and assured that every downhill sequence remained in its restart bin.

For each trial, the number of iterations required to find the goal basin was observed. The raw data consists of 1000 such trials and was first processed to determine the complementary hitting time distribution (chd). Thus for each k, $chd(k)$ is the fraction of runs requiring k or more iterations. Theoretically, the chd is asymptotically geometric,

$$chd(k) \approx \frac{1}{s}\lambda^{k-1}.$$

Therefore a $\log(chd)$ vs $k-1$ plot should tend to a straight line with slope $\log(\lambda)$. This plot was indeed observed to be approximately affine in its central region and least squares was used to calculate its slope and hence λ, see fig. 3 for function (C) and fig. 4 for function (S).

Log CHD Plot for Test Function C

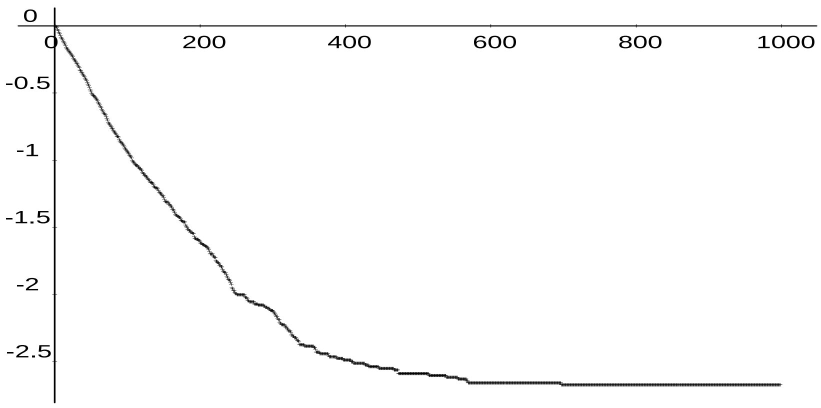

fig. 3

Log CHD Plot for Test Function S

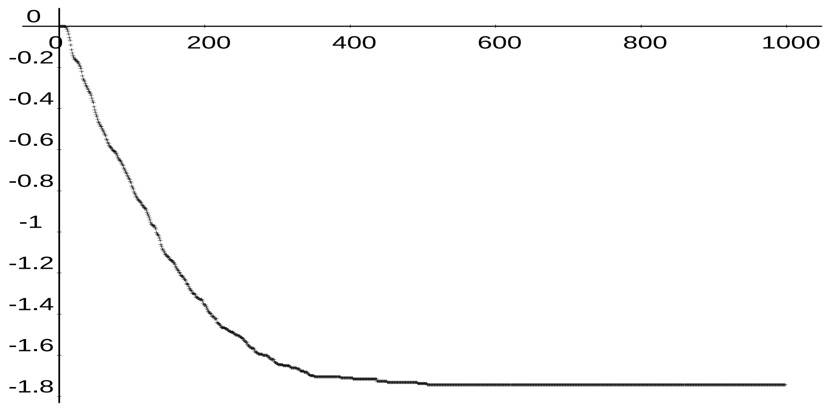

fig. 4

The *chd* plot is not suitable for determining the acceleration factor s however as its intercept is too sensitive to the choice of affine region selected. Instead the s-factor determination is made from the speedup plot, S vs m. Speedup data for m parallel processes were obtained by the *in-code parallel* technique. That is, m separate trial solutions are maintained in each iteration of a single processor algorithm. The first of these to reach the min–bin flags a stop to the others. The hitting time is noted, and a new run initiated. The speedup plot for function (C) is given in fig. 5 and that for function (S) in fig. 6. In these figures, linear speedup is shown for comparison.

Speedup Plot for Test Function C

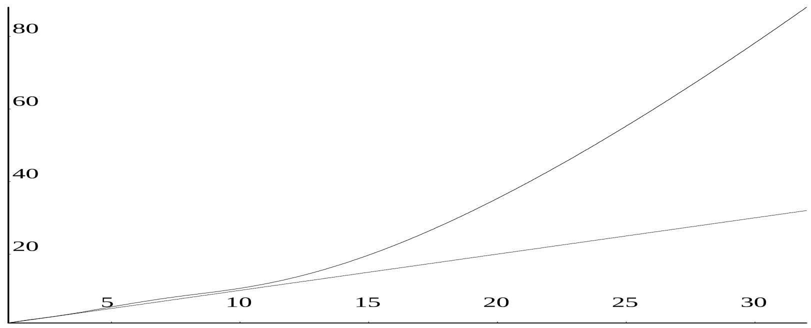

fig. 5

Speedup Plot for Test Function S

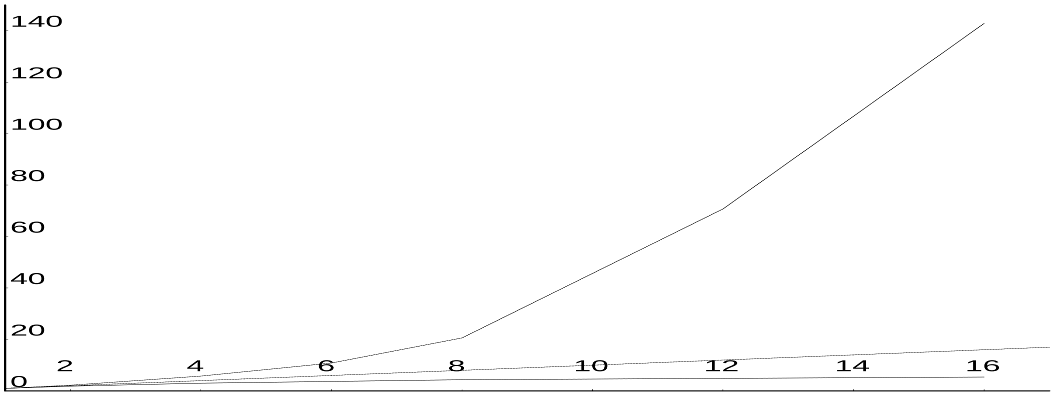

fig. 6

Using the approximate equality

$$S = s^{m-1}\frac{1-\lambda^m}{1-\lambda}$$

and the value of λ previously determined, it is an easy matter to estimate s from the speedup plot.

Before turning to the specific results, we give next an asymptotic result for the determination of $\gamma = 1/\lambda$ and s as the number of points per basin increases without bound while the relative number of points per basin remains fixed.

Theorem 6.1. *If* $\lim_{N\to\infty}\frac{n(i)}{N} = \theta_i$ *for* $i = 1,\ldots,b$ *then denoting by* $\gamma(\mathbf{n})$ *the* $\gamma > 1$ *satisfying* $f(\gamma) = 0$ *for the particular configuration* $n(1),\ldots,n(b)$ *and* f *given in* (3.3), *and by* c *the unique positive real number satisfying the equation*

$$\sum_{i=2}^{b} e^{c\theta_i} - (b-1) - c = 0 \tag{6.1}$$

one has

(i) $\gamma(\mathbf{n}) = 1 + \frac{c}{N} + o(N^{-1})$ *and*

(ii) $\lim_{N\to\infty} s(\mathbf{n}) = \frac{\sum_{i=2}^{b}\theta_i\exp\{c\theta_i\}-1}{\theta_1}$.

6.2 Test Results

For test function (C), λ and s were estimated in three different ways: (1) empirically by the numerical procedure described above, (2) asymptotically according to Theorem 6.1, and (3) by approximate discretization as we now describe. A plot of the objective function fig. 1 was hand discretized every $\Delta x = 0.25$ units and the resulting number of points per basin counted. For example, the second "basin" B_1 consists of 8 points representing the monotone decreasing segment of the function between 1.68 and 3.47. Gradient descent by a Newton's Method algorithm would descend to 3.47 if started anywhere in the interval. The discretized domain is only an approximation to the continuum since Newton's Method will not necessarily select these 8 points. On the other hand, the algorithm will select some discrete subset of points in such a descent, the specific ones selected is not important, only the number of steps taken for any given starting point. From Theorem 6.1, so long as the relative bin sizes are maintained by the discretization, the discretized results approximate the exact ones.

The resulting goal deleted matrix is 101 by 101 with 7 non-goal basins. Its spectral properties can be easily calculated from which λ and s are obtained. The results for all three methods are shown in Table 4.

Test function (S) was analyzed only by the first two methods, empirically and asymptotically with the results shown in Table 4.

λ and s Estimates by Three Methods

Function	Empirical	Asymptotic	Discretization
C	$\lambda = 0.991$, $s = 1.049$	$\lambda = 0.993$, $s = 1.049$	$\lambda = 0.991$, $s = 1.032$
S	$\lambda = 0.992$, $s = 1.147$	$\lambda = 0.994$, $s = 1.179$	–

Table 4

7. Appendix of Proofs

Proof of Theorem 3.1: The series of lemmas below prove Theorem 3.1.

Lemma 7.1. *The polynomial* (3.3) *has a smallest* $\gamma \in (1, \infty)$ *such that* $f(\gamma) = 0$.

Proof. Observe that f is a polynomial of degree > 2 with a positive coefficient of the largest power so that $f(\eta) \to +\infty$ as $\eta \to +\infty$. Also observe that $f(1) = 0$. To prove the lemma it suffices to prove that $f'(1) < 0$. Calculating, one has

$$f'(\eta) = \frac{1}{N} \sum_{i=2}^{b} (n(i) + 1)\eta^{n(i)} - \frac{1}{N}(b - 1) - 1 \tag{7.1}$$

so that

$$\begin{aligned} f'(1) &= N^{-1}(N - n(1) + b - 1) - N^{-1}(b - 1) - 1 \\ &= N^{-1}(N - n(1)) - 1 = 1 - \frac{n(1)}{N} - 1 = -\frac{n(1)}{N} < 0. \end{aligned}$$

□

It is not difficult to see that γ is in fact the unique solution larger than 1 (see Lemma 7.5).

Lemma 7.2. *If* $\gamma > 0$ *then the vector*

$$v' = (1, \gamma, \gamma^2, \ldots, \gamma^{n(2)-1}, 1, \gamma, \gamma^2, \ldots, \gamma^{n(3)-1}, 1, \ldots, \gamma^{n(b)-1})$$

satisfies

$$\hat{P}v = \frac{1}{\gamma}v$$

if

$$\frac{1}{N}1'v = \frac{1}{\gamma}. \tag{7.2}$$

Proof. Multiplying $\hat{P}v$, the condition is easily seen to be sufficient. □

Lemma 7.3. *The Perron–Frobenius eigenvalue of* $\hat{P}$ *is* γ^{-1}, *where the existence of* γ *was established in* Lemma 7.1; *the corresponding right eigenvector is* v_0, *where*

$$v_0' = (1, \gamma, \gamma^2, \ldots, \gamma^{n(2)-1}, 1, \gamma, \gamma^2, \ldots, \gamma^{n(3)-1}, 1, \ldots, \gamma^{n(b)-1}).$$

Proof. By Lemma 7.2 it will be shown that v_0 is an eigenvector if

$$\frac{1}{\gamma} = \frac{1}{N} 1' v_0 = \frac{1}{N} \left(\sum_{i=2}^{b} \frac{1 - \gamma^{n(i)}}{1 - \gamma} \right).$$

To see that it does just note that

$$\frac{1}{\gamma} - \frac{1}{N} \left(\sum_{i=2}^{b} \frac{1 - \gamma^{n(i)}}{1 - \gamma} \right) = \frac{1}{\gamma(1-\gamma)} \left[1 - \gamma - \gamma \frac{1}{N}(b-1) + \frac{1}{N} \sum_{i=2}^{b} \gamma^{n(i)+1} \right].$$

By Lemma 7.1 γ solves (7.3), is positive, and $v_0 \gg 0$. By Theorem 2.2 of Varga, (1963) it follows that γ^{-1} is the Perron–Frobenius eigenvalue and that v_0 is a right eigenvector. $\square$

To calculate the s–factor, we next seek a corresponding left eigenvector w. Simply multiplying the asserted vector in the Theorem 3.1 and using the results above, one can prove the following.

Lemma 7.4. *If* u *is a left eigenvector of* $\hat{P}$ *corresponding to the* $P - F$ *eigenvalue then* u' *is proportional to the vector*

$$\begin{aligned} w' = &(1 - \gamma^{n(2)}, 1 - \gamma^{n(2)-1}, 1 - \gamma^{n(2)-2}, \ldots, 1 - \gamma, \\ &1 - \gamma^{n(3)}, 1 - \gamma^{n(3)-1}, \ldots, 1 - \gamma, \ldots, 1 - \gamma^{n(b)}, \ldots, 1 - \gamma). \end{aligned}$$

A formula for s useful in the proof of Theorem 3.2 follows. The left eigenvector ω is chosen to be proportional to w above with $\omega' 1 = 1$ so $\omega = cw$ and $c = (w'1)^{-1}$. The right eigenvector satisfies $\chi = a v_0$. Assuming that $\hat{\alpha} = \frac{1}{N} 1$ one has

$$s^{-1} = \hat{\alpha}' \chi = \frac{1}{N} \frac{w' 1 v_0' 1}{w' v_0}. \tag{7.3}$$

The following elementary property of functions will be utilized in the proof of Theorem 3.2.

Lemma 7.5. *Let g be a real valued function on $x \geq a$ satisfying $g(a) = 0$, $g'(a) < 0$, $g''(a) > 0$, and g'' non–decreasing on $[a, \infty)$.*

(i) there is a unique point $b > a$ such that $g(b) = 0$ and

(ii) $g'(b) \geq -g'(a)$.

Proof of Theorem 3.2: Observing that, by Lemma (7.3), $v'1 = N\lambda$, one has from (7.3) $\lambda s = w'v/w'1$ where

$$w' = \left(\sum_{u=0}^{n(2)-1} \lambda^{-u}, \sum_{u=0}^{n(2)-2} \lambda^{-u}, \ldots, 1, \ldots, \sum_{u=0}^{n(b)-1} \lambda^{-u}, \ldots, 1 \right)$$

and

$$v' = \left(1, \frac{1}{\lambda}, \ldots, \frac{1}{\lambda^{n(2)-1}}, \ldots, 1 \ldots, \frac{1}{\lambda^{n(b)-1}} \right).$$

Noting that

$$\lambda s = \left(\sum_{j=2}^{b} \sum_{i=1}^{n(j)} i\lambda^{-(i-1)} \right) \Big/ \left(\sum_{j=2}^{b} \sum_{i=1}^{n(j)} i\lambda^{(i-n(j))} \right)$$

and summing yields

$$\lambda s = \frac{\sum_{j=2}^{b} (n(j)\lambda^{-n(j)+1} - (n(j)+1)\lambda^{-n(j)+2} + \lambda^2)}{\sum_{j=2}^{b} (\lambda^{-n(j)+1} - (n(j)+1)\lambda + n(j)\lambda^2)} \tag{7.4}$$

By (3.3), for the PF eigenvalue

$$\frac{1}{N} \sum_{j=2}^{b} \lambda^{-(n(j)+1)} - \left(\frac{1}{N}(b-1) + 1 \right) + 1 = 0$$

so that

$$\sum_{j=2}^{b} \lambda^{-(n(j)+1)} = \lambda^{-1}(b - 1 + N(1-\lambda))$$

and using this, $\sum_{j=2}^{b} n(j) = N - n(1)$, and some algebraic manipulations in (7.4) we have

$$\lambda s = \frac{\lambda \left(\sum_{j=2}^{b} n(j)\lambda^{-(n(j)+1)} - N \right)}{n(1)}$$

Changing parameters to $\gamma = \lambda^{-1}$ and utilizing the derivative f' given in (7.1) one has

$$\begin{aligned} sn(1) &= \sum_{j=2}^{b} n(j)\gamma^{(n(j)+1)} - N = \gamma \sum_{j=2}^{b} n(j)\gamma^{n(j)} - N \\ &= \gamma \left[Nf'(\gamma) - \sum_{j=2}^{b} \gamma^{n(j)} + b - 1 + N \right] - N \\ &= \gamma[Nf'(\gamma) - \gamma^{-1}(Nf(\gamma) + (b-1+N)\gamma - N) + b - 1 + N] - N \end{aligned}$$

and since $f(\gamma) = 0$, upon simplification

$$sn(1) = \gamma[Nf'(\gamma) - \gamma^{-1}N] - N = \gamma N f'(\gamma).$$

Recalling some properties of f we have $f(\gamma) = 0$, $f'(1) = -\theta_1$, and

$$f''(\eta) = \frac{1}{N} \sum_{j=2}^{b} (n(j)+1)n(j)\eta^{n(j)-1}$$

which is increasing in $\eta > 1$ so that f satisfies the hypotheses of Lemma 7.5 and $f'(\gamma) > -f'(1) = \theta_1$. Therefore,

$$s = \frac{\gamma}{\theta_1} f'(\gamma) \geq \frac{\gamma}{\theta_1}\theta_1 = \gamma > 1. \qquad \square$$

Proof of Lemma 5.1: Observe that for all n

$$\mathbb{P}[F_n^c] = \int (1-\theta)^n \pi(\theta)d\theta = \int_a^1 (1-\theta)^n \pi(\theta)d\theta \leq (1-a)^n \int_a^1 \pi(\theta)d\theta = (1-a)^n. \qquad \square$$

Proof of Lemma 5.2: Let $c > 0$ be arbitrary and note that for n sufficiently large

$$\begin{aligned} \mathbb{P}[F_n^c] &= \int (1-\theta)^n \pi(\theta)d\theta \geq \int_0^{c/n} (1-\theta)^n \pi(\theta)d\theta \\ &\geq (L/2) \int_0^{c/n} (1-\theta)^n \theta^\alpha d\theta \\ &\geq (1-c/n)^n (L/2) \frac{c^{\alpha+1}}{(\alpha+1)n^{\alpha+1}} \end{aligned}$$

Therefore

$$\liminf_{n\to\infty} \mathbb{P}[F_n^c] n^{\alpha+1} \geq Lc^{\alpha+1}e^{-c}/2(\alpha+1) > 0. \qquad \square$$

Proof of Lemma 5.3: Let $\eta \in (0,1)$ be arbitrary, $\delta \in (0, \eta/(1+\alpha))$ and $a(n) = c/n^{1-\delta}$. Then

$$\begin{aligned}\mathbb{P}[F_n^c] &= \int (1-\theta)^n \pi(\theta) d\theta = \int_0^{a(n)} (1-\theta)^n \pi(\theta) d\theta + \int_{a(n)}^1 (1-\theta)^n \pi(\theta) d\theta \\ &\leq \int_0^{a(n)} (1-\theta)^n \pi(\theta) d\theta + (1-a(n))^n.\end{aligned}$$

Let $\epsilon > 0$ be arbitrary and choose θ_0 such that $0 < \theta < \theta_0$ entails $\pi(\theta)/\theta^\alpha \leq L + \epsilon$. Then since $a(n) \to 0$ with n one has for n sufficiently large

$$\begin{aligned}\mathbb{P}[F_n^c] &\leq (L+\epsilon) \int_0^{a(n)} (1-\theta)^n \theta^\alpha d\theta + (1-a(n))^n \\ &\leq \frac{L+\epsilon}{1+\alpha} (a(n))^{\alpha+1} + (1-a(n))^n.\end{aligned}$$

Since for any positive a and c and for all $b \in (0,1)$ one has $n^a(1-c/n^b)^n \to 0$ as $n \to \infty$, it follows that

$$\begin{aligned}\limsup_{n\to\infty} \mathbb{P}[F_n^c] n^{\alpha+1-\eta} &\leq \limsup_{n\to\infty} n^{\alpha+1-\eta} \left\{ \frac{L+\epsilon}{1+\alpha} (a(n))^{\alpha+1} + (1-a(n))^n \right\} \\ &= \limsup_{n\to\infty} \left\{ \frac{L+\epsilon}{1+\alpha} c^{\alpha+1} n^{\alpha+1-\eta-\alpha-1+(1+\alpha)\delta} + n^{\alpha+1-\eta}(1-c/n^{1-\delta})^n \right\} = 0. \square\end{aligned}$$

Proof of Theorem 6.1: Write $\gamma(\mathbf{n}) = 1 + \frac{x(\mathbf{n})}{N}$. We have already observed that $\lim_{N\to\infty} \frac{x(\mathbf{n})}{N} = 0$ and claim now that in fact $\lim_{N\to\infty} x(\mathbf{n}) = c > 0$. To see this it suffices to prove that $x(\mathbf{n})$ is a bounded sequence. We shall prove shortly that it is a bounded sequence, but assuming that boundedness has been proven, there would be a convergent subsequence $x(\mathbf{n}')$ to a point c' and

$$\gamma(\mathbf{n}') = 1 + \frac{c'}{N} + o(N'^{-1}).$$

Therefore from (3.3)

$$
\begin{aligned}
0 &= f(\gamma(\mathbf{n}')) \\
&= \sum_{i=2}^{b} \left[\left(1 + \frac{c'}{N} + o(N'^{-1}) \right) N' \right]^{n(i)+1)/N'} - (b-1)\left(1 + \frac{x(\mathbf{n})}{N}\right) - x(\mathbf{n}') \\
&\to \sum_{i=2}^{b} e^{c'\theta_i} - (b-1) - c'
\end{aligned}
$$

and it follows that either c' is the c satisfying (6.1) or $c' = 0$. If c' were 0 then one would have $s(n') \to -1$ which is impossible since $s(\mathbf{n}) > 1$ always (see Theorem 3.2); so $c' = c$. Since every subsequence has a further subsequence converging to c it follows that $x(\mathbf{n}) \to c$.

To conclude the proof we must prove that $x(\mathbf{n})$ is bounded. Suppose not. Then there is a subsequence $x(\mathbf{n}') \to \infty$ and for all N'

$$
0 = f(\gamma(\mathbf{n}')) = \sum_{i=2}^{b} \frac{1}{x(\mathbf{n}')} \left[\left(1 + \frac{x(\mathbf{n}')}{N'} \right)^{N'} \right]^{(n'(i)+1)/N'} - \frac{b-1}{x(\mathbf{n}')}\left(1 + \frac{x(\mathbf{n}')}{N'}\right) - 1.
$$

The following inequality is true for all $b > 1/2$ and x/N' sufficiently small:

$$
\left(1 + \frac{x}{N'}\right)^{N'+bx} \geq e^{x}.
$$

It follows that for N' sufficiently large

$$
\frac{1}{x(\mathbf{n}')}\left(1 + \frac{x(\mathbf{n}')}{N'}\right)^{N'} \geq \frac{1}{x(\mathbf{n}')}\left(\frac{e}{(1+\frac{x(\mathbf{n}')}{N'})^{b}}\right)^{x(\mathbf{n}')} \geq \frac{1}{x(\mathbf{n}')} 2^{x(\mathbf{n}')} \to \infty.
$$

But we should have

$$
\sum_{i=2}^{b} \frac{1}{x(\mathbf{n}')} \left[\left(1 + \frac{x(\mathbf{n}')}{N'} \right)^{N'} \right]^{(n'(i)+1)/N'} \to 1
$$

so this contradiction shows that $x(\mathbf{n})$ is bounded and concludes the proof of the theorem. $\square$

References

[1] Dimitri Bertsekas, John Tsitsiklis, *Parallel and Distributed Computation*, Prentice Hall, Englewood Cliffs (1989)

[2] G. Boender, and A. Rinnooy Kan, *Bayesian Stopping Rules for Multistart Optimization Methods,* Math. Programming, 37(1987), pp .59–80.

[3] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, (1983)

[4] J. Galambos, *The Asymptotic Theory of Extreme Order Statistics*, Wiley, New York, (1978)

[5] J. Pickands, *Moment Convergence of Sample Extremes,* Ann. Math. Statist., 39(1968), pp. 881-889.

[6] W. Rudin, *Real and Complex Analysis*, McGraw-Hill, New York, (1966)

[7] F. Schoen, *Stochastic Techniques for Global Optimization: A Survey of Recent Advances,* J. of Global Optimization, 1(1991), pp. 207–228.

[8] R. Shonkwiler, and E. Van Vleck, *Parallel speed-up of Monte Carlo methods for global optimization,* to appear in the Journal of Complexity.

[9] F. Solis, and R. Wets, *Minimization by Random Search Techniques,* Math. of Operations Research, 6(1981), pp. 19–30.

[10] A. Törn, and A. Zelinskas, *Global Optimization*, Springer-Verlag, New York, (1989)

[11] R. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ (1963).