Algorithm Description: Random Search*

Jason Brownlee
jasonb@CleverAlgorithms.com
The Clever Algorithms Project
http://www.CleverAlgorithms.com

January 29, 2010
Technical Report: CA-TR-20100129-1

Abstract

The Clever Algorithms project aims to describe a large number of Artificial Intelligence algorithms in a complete, consistent, and centralized manner. This report describes the Random Search algorithm using a standardized algorithm description template. As the first algorithm description in the project, a few modifications are suggested for the standardized description template and some bounds are suggested for the broader project ideals.

Keywords: `Algorithm, Description, Random, Search, Optimization`

1 Introduction

The Clever Algorithms project aims to describe a large number of algorithms from the fields of Computational Intelligence, Biologically Inspired Computation, and Metaheuristics in a complete, consistent and centralized manner [2]. The project requires all algorithms to be described using a standardized template that includes a fixed number of sections, each of which is motivated by the presentation of specific information about the technique [4]. This report describes the 'Random Search' algorithm using the standardized algorithm description template. Section 11 summarizes some suggested changes to the standardized description template, comments on the limits the projects ideals, and highlights some related algorithms that may be considered for inclusion in the project.

2 Name

Random Search, RS, Blind Random Search, Blind Search, Pure Random Search, PRS.

3 Taxonomy

Random search belongs to the fields of Stochastic Optimization and Global Optimization. Random search is a direct search method as it does not require derivatives to search a continuous domain. This base approach is related to techniques that provide small improvements such as Directed Random Search, and Adaptive Random Search.

4 Inspiration

N/A

5 Metaphor

N/A

6 Strategy

The strategy of Random Search is to sample solutions from across the entire search space using a uniform probability distribution. Each future sample is independent of the samples that come before it.

7 Procedure

Algorithm 1 provides a pseudo-code listing of the Random Search Algorithm for minimizing a cost function.

Algorithm 1: Pseudo Code Listing for the Random Search Algorithm.

```
Input: NumIterations, ProblemSize, SearchSpace
Output: Best
Best ← 0;
foreach $iter_i$ ∈ NumIterations do
    $candidate_i$ = RandomSolution(ProblemSize, SearchSpace);
    if Cost($candidate_i$) < Cost(Best) then
        Best ← $candidate_i$;
    end
end
return Best;
```

8 Heuristics

- Random search is minimal in that it only requires a candidate solution construction routine and a candidate solution evaluation routine, both of which may be calibrated using the approach.
- The worst case performance for Random Search for locating the optima is worse than an Enumeration of the search domain, given that Random Search has no memory and can blindly resample.
- Random Search can return a reasonable approximation of the optimal solution within a reasonable time under low problem dimensionality, although the approach does not scale well with problem size (such as the number of dimensions).
- Care must be taken with some problem domains to ensure that random candidate solution construction is unbiased
- The results of a Random Search can be used to seed another search technique, like a local search technique (such as the Hill Climbing algorithm) that can be used to locate the best solution in the neighborhood of the 'good' candidate solution.

9 Code Listing

Listing 1 provides an example of the Random Search Algorithm implemented in the Ruby Programming Language. In the example, the algorithm runs for a fixed number of iterations and returns the best candidate solution discovered. The example problem is an instance of a continuous function optimization that seeks $min f(x)$ where $f = \sum_{i=1}^{n} x_i^2$, $-5.0 \leq x_i \leq 5.0$ and $n = 2$. The optimal solution for this basin function is $(v_0, \ldots, v_{n-1}) = 0.0$.

```ruby
NUM_ITERATIONS = 100
PROBLEM_SIZE = 2
SEARCH_SPACE = Array.new(PROBLEM_SIZE) {|i| [-5, +5]}

def cost(candidate_vector)
  return candidate_vector.inject(0) {|sum, x| sum + (x ** 2.0)}
end

def random_solution(problemSize, searchSpace)
  return Array.new(problemSize) do |i|
    searchSpace[i][0] + ((searchSpace[i][1] - searchSpace[i][0]) * rand)
  end
end

def search(numIterations, problemSize, searchSpace)
  best = nil
  numIterations.times do |iter|
    candidate = {}
    candidate[:vector] = random_solution(problemSize, searchSpace)
    candidate[:cost] = cost(candidate[:vector])
    best = candidate if best.nil? or candidate[:cost] < best[:cost]
    puts " > iteration #{(iter+1)}, best: c=#{best[:cost]}, v=#{best[:vector].inspect}"
  end
  return best
end

best = search(NUM_ITERATIONS, PROBLEM_SIZE, SEARCH_SPACE)
puts "Done. Best Solution: c=#{best[:cost]}, v=#{best[:vector].inspect}"
```

Listing 1: Random Search Algorithm in the Ruby Programming Language

10 References

10.1 Primary Sources

There is no seminal specification of the Random Search algorithm, rather there are discussions of the general approach and related random search methods from the 1950's through to the 1970's. This was around the time that pattern and direct search methods were actively researched. Brooks is credited with the so-called 'pure random search' [1]. Two seminal reviews of 'random search methods' of the time include: Karnopp [5] and prhaps Kul'chitskii [6].

10.2 Learn More

For overviews of into Random Search Methods see Zhigljavsky [14], Solis and Wets [9], and also White [12] who provides an excellent review article. Spall provides a detailed overview of the field of Stochastic Optimization, including the Random Search method [10] (for example, see Chapter 2). For a shorter introduction by Spall, see [11] (specifically Section 6.2). Also see Zabinsky for another detailed review of the broader field [13].

11 Conclusions

This report introduced the Random Search technique as a baseline procedure suitable as a default technique for a difficult problem, and a minimal method suitable for calibrating solution construction and evaluation procedures. The description of this technique did not cover inspiration and metaphor as these topics were not relevant to the algorithm. The description deviated from the standardized template in that a *code listing* was provided with a brief description, instead of a *tutorial* [4]. It is believed that a minimal code listing and small context-setting description are more suitable for the Clever Algorithms project, and this change is suggested for all algorithm descriptions to be prepared for the Clever Algorithms project. It is also suggested that the same general structure and layout (such as function names) be used between the pseudo code algorithm and the source code listing to ensure consistency.

It is apparent that the Clever Algorithms project takes a very practical approach to algorithm description, not directly considering mathematical descriptions of algorithms or their expected behavior. This further highlights that the ideals of the Clever Algorithms project are in fact bounded by default [2]. Specifically, the project will describe algorithms in a complete, consistent and centralized manner, although the descriptions will be 'complete enough' for a practitioner to understand and use them. They will be 'consistent enough' to ensure continuity between algorithm descriptions, and the descriptions will be 'centralized enough' to be consumed in a book and website format. These clear bounds on the project ideals must be specified in the introductory information provided with the descriptions, such as the introduction chapter of the book.

Random Search is closely related to *Localized Random Search* (use of a step size), *Adaptive Random Search* (use of a adaptive step size), and the *Directed Random Search* (sometimes a synonym for Localized and/or Adaptive Random Search) techniques, all of which appear in the data-driven algorithm selection results, and some of which have been selected for inclusion in the Clever Algorithms project [3]. Some additional techniques uncovered during research that may be considered for inclusion the project include: *Controlled Random Search* [7], and *Creeping Random Search* [8].

12 Contribute

Found a typo or a bug in the source code? Are you an expert in this technique and know some facts that could improve the algorithm description for all? Do you want to get that warm feeling from contributing to an open source project? Do you want to see your name as an acknowledgment in print? Please contact the author via email `jasonb@CleverAlgorithms.com` or visit the project website at `http://www.CleverAlgorithms.com` and let us know how we can improve this work.

References

[1] Samuel H. Brooks. A discussion of random methods for seeking maxima. *Operations Research*, 6(2):244–251, 1958.

[2] Jason Brownlee. The clever algorithms project: Overview. Technical Report CA-TR-20100105-1, The Clever Algorithms Project http://www.CleverAlgorithms.com, January 2010.

[3] Jason Brownlee. Data-driven optimization algorithm selection. Technical Report CA-TR-20100112-1, The Clever Algorithms Project http://www.CleverAlgorithms.com, January 2010.

[4] Jason Brownlee. A template for standardized algorithm descriptions. Technical Report CA-TR-20100107-1, The Clever Algorithms Project http://www.CleverAlgorithms.com, January 2010.

[5] Dean C. Karnopp. Random search techniques for optimization problems. *Automatica*, 1(2–3):111–121, 1963.

[6] O. Yu. Kul'chitskii. Random-search algorithm for extrema in functional space under conditions of partial uncertainty. *Cybernetics and Systems Analysis*, 12(5):794–801, 1976.

[7] W. L. Price. A controlled random search procedure for global optimisation. *The Computer Journal*, 20(4):367–370, 1977.

[8] L. A. Rastrigin. The convergence of the random search method in the extremal control of a many parameter system. *Automation and Remote Control*, 24:1337–1342, 1963.

[9] F. J.. Solis and J. B. Wets. Minimization by random search techniques. *Mathematics of Operations Research*, 6:19–30, 1981.

[10] James C. Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*. John Wiley and Sons, 2003.

[11] James C. Spall. *Handbook of computational statistics: concepts and methods*, chapter 6. Stochastic Optimization, pages 169–198. Springer, 2004.

[12] R. C. White. A survey of random methods for parameter optimization. *Simulation*, 17(1):197–205, 1971.

[13] Z. B. Zabinsky. *Stochastic adaptive search for global optimization*. Kluwer Academic Publishers, 2003.

[14] A. A. Zhigljavsky. *Theory of Global Random Search*. Kluwer Academic, 1991.