# A Performance Analysis of Evolutionary Pattern Search with Generalized Mutation Steps

**William E. Hart**
Applied Mathematics Department
Sandia National Laboratories
P.O. Box 5800, MS 1110
Albuquerque, NM 87185 USA
wehart@cs.sandia.gov

**Keith O. Hunter**
Applied Mathematics Department
Sandia National Laboratories
P.O. Box 5800, MS 1110
Albuquerque, NM 87185 USA
kohunte@cs.sandia.gov

**Abstract- Evolutionary pattern search algorithms (EPSAs) are a class of evolutionary algorithms (EAs) that have stationary-point convergence guarantees on a broad class of nonconvex continuous problems. We have analyzed the empirical performance of EPSAs. This paper revisits that analysis and extends it to a more general model of mutation. We evaluate experimentally how the choice of the set of mutation offsets affects optimization performance for EPSAs. In addition, we compare EPSAs to self-adaptive EAs with respect to robustness and rate of optimization. All experiments employ a suite of test functions representing a range of modality and number of multiple minima.**

## 1 Introduction

Evolutionary pattern search algorithms (EPSAs) are distinguished from other evolutionary algorithms (EAs) by the convergence theory which proves that they almost surely converge to a stationary point of any continuously differentiable function [8, 6, 5]. Convergence proofs have been developed for other evolutionary algorithms for convex functions [2, 1]. However, the convergence theory for evolutionary pattern search provides the first assurance of a stationary-point convergence for multimodal, nonconvex problems.

EPSAs are also distinguished from most other EAs applied to continuous search problems by the manner in which mutation is applied. EPSAs perform mutation by adding a scaled integer vector to an individual, and these mutation vectors are selected from a finite set of possible mutation offsets. By contrast, methods like evolutionary programming [4] and evolution strategies [11] typically add a vector of offsets that is normally distributed in each dimension.

The initial analysis of EPSAs [6, 5] considered a set of mutation offsets defined by the unit vectors along each coordinate axis, $e_i$ and $-e_i$. More recently, we have extended the analysis of EPSAs to allow a broader class of mutation offsets [8]. Specifically, the set of mutation offsets is simply required to form a positive basis of the search domain.

In addition to extending the class of EPSAs, this analysis allows the step length parameter for mutation to be reduced after $n + 1$ mutation steps instead of the $2n$ mutation steps required by the former convergence theory. Since this analysis is asymptotic, these new EPSAs may not necessarily converge more quickly. However, our prior work with EPSAs [7] suggests that the performance of EPSAs can be limited by the rate at which they reduce the mutation step length.

This paper presents an experimental analysis that evaluates the relative importance of several algorithmic factors for EPSAs: the set of mutation offsets, the use of a crossover operator, the expansion factor for the mutation step length, and the manner in which mutation is selected stochastically. Our experiments confirm that EPSAs converge more quickly when fewer mutation steps need to be considered before reducing the mutation step length. Using crossover uniformly produced superior solutions at the cost of requiring substantially more function evaluations to terminate. Finally, our experiments compare the performance of EPSAs with EP, and indicate that EPSAs are competitive with EPs.

## 2 Background

### 2.1 Self-Adaptive Evolutionary Algorithms

Evolutionary programming (EP) and evolution strategies (ES) are standard paradigms for applying evolutionary methods to continuous optimization problems [4, 11]. EP and ES are similar in many respects. These EAs do not always rely on recombination to perform a global search of the search domain. In both EAs, mutation is performed by adding normally distributed random variables to each dimension of an individual. Furthermore, the standard deviation of these normal deviates is typically modified by a self-adaptive mechanism. This mechanism can be viewed as a separate encoding of the mutation standard deviation along with the search parameters.

Figure 1 shows pseudo-code for a canonical EP or ES that uses self-adaptation. $N(0,1)$ is a normally distributed variable with standard deviation 1, and $N(0,\sigma)$ is a vector of normally distributed random variables with standard deviation $\sigma_i$. The function **selection** selects individuals from the previous population (possibly creating a multiset) that are used to perform additional search, and the function **compose** forms the next population using the newly generated points and the previous population. This code uses a log normal update to $\sigma_i$; on a test set of six standard functions, Saravanan,

(1) Given initial step length vectors $\{\sigma_1^0, \ldots, \sigma_N^0\}$
(2) Uniformly select $X_0 = \{x_1^0, \ldots, x_N^0\}$, $x_i^0 \in \mathbf{R}^n$
(3) $x_0^* = \arg\min\{f(x_1^0), \ldots, f(x_N^0)\}$
(4) Repeat $t = 0, 1, \ldots$
(5) $\quad \hat{X} = \text{selection}(X_t)$
(6) $\quad$ For $i = 1 : N$
(7) $\quad\quad \nu = N(0,1)$
(8) $\quad\quad$ For $j = 1 : n$
(9) $\quad\quad\quad \sigma_i^{t+1}(j) = \sigma_i^t(j) * \exp(\tau'\nu + \tau N(0,1))$
(10) $\quad\quad\quad \hat{x}_i(j) = \hat{x}_i(j) + N(0, \sigma_i^{t+1}(j))$
(11) $\quad X_{t+1} = \text{compose}(X_t, \hat{X})$
(12) $\quad x_{t+1}^* = \arg\min\{f(x_t^*), f(x_1^{t+1}), \ldots, f(x_N^{t+1})\}$
(13) Until some stopping criterion is satisfied

Figure 1: **A canonical EP or ES using self-adaptation.**

Fogel, and Nelson [10] find that this method is generally preferable to the additive update that was initially proposed for EP. This update uses the constants $\tau = \left(\sqrt{2\sqrt{n}}\right)^{-1}$ and $\tau' = \left(\sqrt{2n}\right)^{-1}$ [10]. The stopping rules used for EP and ES methods typically rely on measures of the rate of improvement or population statistics that evaluate whether or not the population has converged to a single point [4, 11].

### 2.2 Evolutionary Pattern Search Algorithms (EPSA)

Figure 2 shows pseudo-code for a class of simple EPSAs. These methods share many of the common features of standard EAs like EP and ES. Mild conditions are placed upon the **selection** and **compose** functions to ensure that (a) the best point in the population has a nonzero chance of being selected in each generation and (b) the best point in the population is always kept in subsequent populations. The **crossover** function is also restricted to generate a point such that $\text{crossover}(x, y) \in \{x_1, y_1\} \times \{x_2, y_2\} \times \ldots \times \{x_n, y_n\}$, which is consistent with most standard crossover operators (e.g. two-point crossover). The call to $\text{uint}(j)$ uniformly generates an integer from 1 to $j$.

EPSAs differ from self-adaptive EAs like the EP in Figure 1 in that the step length parameter is controlled explicitly. EPSAs use a single step length parameter for all dimensions. The step length parameter may be expanded if an improving step is generated from a mutation step off of the current best point. Also, the step length may be contracted if all mutation steps about the current best point have worse fitness than the current best point.

This method of explicitly controlling the step length for mutation enables a stationary point convergence theory for EPSAs. This convergence theory guarantees that for a continuously differentiable function the sequence of best solutions in each generation, $\{x_k^*\}$, has the property that

$$P\left(\liminf_{k\to\infty} \| \bigtriangledown f(x_k^*)\| = 0\right) = 1,$$

where $\bigtriangledown f(x)$ is the gradient of $f$ at $x$ [6, 5]. Although this is a stationary-point convergence theory, experience with direct search methods suggests that EPSAs can be successfully applied to a wide range of optimization problems [9, 13, 14]. Our previous empirical evaluation of EPSAs [7] indicates that they can perform a nonlocal optimization of the search domain.

We can replace step 13 in Figure 2 with
(13.a) For $i = 1 : N$
(13.b) $\quad y = \hat{x}_i$
(13.c) $\quad$ For $j = 1 : k$
(13.d) $\quad\quad$ If $(\text{unif}() < \mu)$ then
(13.e) $\quad\quad\quad \hat{x}_i = \hat{x}_i + \Delta t \cdot s_j$
(13.f) $\quad$ If $(\exists j \in 1 : k$ s.t. $\hat{x}_i == y + s_j)$ $\eta_j = 1$

This performs a random selection of the mutation steps, potentially applying multiple mutation steps. This is roughly equivalent to the *binomial* mutation [7] that is commonly used in genetic algorithms (see Section 3). Suppose that $S = S_{\text{std}} = \{e_1, -e_1, \ldots, e_n, -e_n\}$, where $e_i$ is the unit vector in the $i$th dimension. The method used to implement mutation in Figure 2 is equivalent to the *multinomial* mutation operator described by Hart [7], which mutates a single dimension at a time.

## 3 EPSA Design

Although the definition of EPSAs is broad enough to encompass a wide range of algorithmic options, the convergence analysis for these methods provides little insight into what types of algorithmic designs will provide the best empirical performance. We have previously examined algorithmic factors likely to impact the empirical performance of EPSAs, like the choice of the mutation operator and the effect of crossover [7].

Here we address two additional factors. First we evaluate the utility of allowing the step length to be expanded or contracted. In our previous experimental analysis the step length was only contracted, but the convergence theory allows for step length expansion as well. We consider a simple step length expansion policy: if a mutation step from the best point yields an improvement then expand the step length. Expansion allows an EPSA to follow a descent direction more rapidly. However, this may also limit the rate at which an EPSA converges to a stationary point. Consequently, it is unclear whether this algorithmic factor is beneficial.

Secondly we reconsider the set of mutation steps used in the EPSA. The generalized convergence theory requires that the set of mutation offsets form a positive basis of the search domain. The *positive span* of a set of vectors $\{a_1, \ldots, a_r\}$ is the cone $\{a \in \mathbf{R}^n \mid a = c_1 a_1 + \ldots + c_r a_r, c_i \geq 0 \quad \forall i\}$. The set $\{a_1, \ldots, a_r\}$ is *positive independent* if none of the $a_i$'s is a positive combination of the others. A *positive basis* is a positive independent set whose positive span is $\mathbf{R}^n$. A positive basis has at least $n+1$ vectors and at most $2n$ vectors.

Figure 3 illustrates two sets of mutation offsets. Figure 3a depicts the *standard mutation offsets*. This set of offsets uses

(1) Given $\Delta_0 \in \mathbf{Q}^{>0}$
(2) Given $S = \{s_1, \ldots, s_m\}$, where $s_i \in \mathbf{Z}^n$ and $S$ forms a positive basis.
(3) Let $\eta = \{0\}^m$
(4) Uniformly select $X_0 = \{x_1^0, \ldots, x_N^0\}$, $x_i^0 \in \mathbf{Q}^n$
(5) $x_0^* = \arg\min\{f(x_1^0), \ldots, f(x_N^0)\}$
(6) Repeat $t = 0, 1, \ldots$
(7) $\quad \overline{X} = \text{selection}(X_t)$
(8) $\quad$ For $i = 1 : N$
(9) $\quad\quad$ If $(\text{unif}() < \chi)$ then
(10) $\quad\quad\quad \hat{x}_i = \text{crossover}(\overline{x}_{\text{uint}(N)}, \overline{x}_{\text{uint}(N)})$
(11) $\quad\quad$ Else
(12) $\quad\quad\quad \hat{x}_i = \overline{x}_{\text{uint}(N)}$
(13.a) For $i = 1 : N$
(13.b) $\quad$ If $(\text{unif}() < \mu)$ then
(13.c) $\quad\quad j = \text{uint}(m)$
(13.d) $\quad\quad$ If $(\hat{x}_i == x_{t-1}^*)\ \eta_j = 1$
(13.e) $\quad\quad \hat{x}_i = \hat{x}_i + \Delta_t \cdot s_j$
(14) $\quad X_{t+1} = \text{compose}(X_t, \hat{X})$
(15) $\quad x_{t+1}^* = \arg\min\{f(x_1^{t+1}), \ldots, f(x_N^{t+1})\}$
(16) $\quad$ If $(f(x_{t+1}^*) < f(x_t^*))$
(17) $\quad\quad$ If $(\exists s \in S \text{ s.t. } x_{t+1}^* = x_t^* + s)\ \Delta_{t+1} = \Delta_t * 2$
(18) $\quad\quad \eta = \{0\}^m$
(19) $\quad$ ElseIf $(|\eta| == m)$
(20) $\quad\quad \eta = \{0\}^m$
(21) $\quad\quad \Delta_{t+1} = \Delta_t / 2$
(22) $\quad$ Else
(23) $\quad\quad \Delta_{t+1} = \Delta_t$
(24) Until $(\Delta_{t+1} < \Delta_{lb})$

Figure 2: **A simple EPSA using multinomial mutation.**

$S = S_{\text{std}} = \{e_1, -e_1, \ldots, e_n, -e_n\}$. Hence, the mutation steps are parallel to coordinate axes. $S_{\text{std}}$ contains $2n$ mutation offsets, so it forms a maximal positive basis. Figure 3b depicts a set of mutation offsets that form a minimal positive basis. The $n+1$ mutation offsets are defined by vectors from the centroid of a regular simplex to each of its corners. This set of mutation offsets consists of axes that are separated by an angle of 120 degrees. The regular simplex is an equilateral triangle in two dimensions, a tetrahedron in three dimensions, and so on. In $n$ dimensions *regular simplex mutation offsets* can be derived using the method defined by Spendley, Hext, and Himsworth [12].

Finally, we consider the impact of the manner in which mutation offsets are chosen. We have previously identified two different ways of randomly selecting mutation offsets. The multinomial method selects a single mutation offset uniformly at random. For the standard set of mutation offsets, this corresponds to mutating a single dimension. For the regular set of mutation offsets, this corresponds to adding a vector to the point.

The binomial method randomly selects each mutation offset with a fixed probability, and the final mutation offset is the vector sum of these mutation offsets. For the standard set of mutation offsets this requires checks to prevent the selection of pairs of mutation offsets that add to zero. More generally, a binomial mutation operator must avoid selecting a set of mutation offsets that add to a zero vector. In the case of the regular simplex mutation offsets, this can only happen if all the offsets are selected.

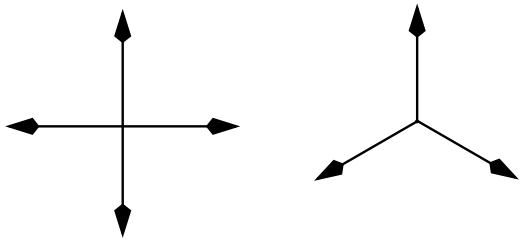


Figure 3: Illustrations of sets of mutation offsets: (a) standard mutation offsets and (b) regular simplex mutation offsets.

The manner in which offsets are chosen may affect the EPSA's ability to follow a descent direction. Because the binomial method generates a larger number of mutation offsets, using it will result in a less focused search. Because the multinomial method facilitates a more focused search, it is reasonable to expect that it leads to a more localized search. We

have previously shown that use of the multinomial method increases the rate at which mutation steps are adapted, which was confirmed by the faster convergence observed in preliminary experiments [7]. We revisit the selection of mutated components in this work in order to study the interaction between the set of mutation offsets employed and this factor.

## 4 Experimental Evaluation

### 4.1 Methods

Experiments were run with the EPSA varying 4 different algorithmic factors. These factors were (1) a crossover rate of 0% or 80% for two-point crossover, (2) standard or regular simplex mutation offsets, (3) step length expansion allowed or not allowed, and (4) binomial or multinomial selection of mutated components. For both EP and EPSA tests, we used a population size of 50, and for each problem the number of trials was 50. When the EPSAs used multinomial mutation, the mutation operator was always applied, and when the EPSAs used binomial mutation, the mutation operator was applied at a rate of 10%. Thus the expected number of mutations was the same for both methods. The initial step length for EPSAs was set to 20, and they were terminated when the mutation step length fell below a threshold of $10^{-8}$.

We set up the EP step length parameters to correspond to the step lengths for the EPSA. Specifically, the initial vectors $\sigma_j^0$ were selected so that the expected distance of mutation was equal to 20. Consider the $j$th point in the initial population. An EP/ES performs mutation in each dimension by adding the vector of offsets $N(0,\sigma)$, and in the initial population the vector $\sigma$ is typically initialized to a vector of constant values, so $\sigma_j^0(i) = \bar{\sigma}$.

The sum of the squares of $n$ normally distributed random variables is a chi-squared variate [3], so

$$\sum_{i=1}^{n} N(0,\bar{\sigma})^2 \equiv \bar{\sigma}^2 \sum_{i=1}^{n} N(0,1)^2 \equiv \bar{\sigma}^2 \chi^2(n).$$

The positive square root of the chi-squared variate $\chi^2(n)$ has expectation $2^{1/2}\Gamma[(n+1)/2]/\Gamma[n/2]$, where $\Gamma$ is the gamma function [3]. Thus we have

$$E\left(\sqrt{\sum_{i=1}^{n} N(0,\bar{\sigma})}\right) = \bar{\sigma}2^{1/2}\Gamma[(n+1)/2]/\Gamma[n/2].$$

For large $n$, this expectation approaches $\sqrt{n}\,\bar{\sigma}$. For the value $n = 10$, the expectation is approximately $3.084\bar{\sigma}$.

Thus we set $\bar{\sigma} = 6.485$ to make the expected initial step length for EP equal to 20. For the EP we also bounded the $\sigma_i$ values below by $10^{-8}/\sqrt{10}$, which kept the step length above $10^{-8}$. This makes the comparison between EPSA and EP fair by not allowing the EP to shrink its step length below the step length of the EPSA. The EP was terminated after 700,000 function evaluations, and performance comparisons between EPSA and EP were made based upon the termination point for the EPSA.

We used five well-understood test functions in our experiments: Griewank, Ackley, Rastrigin, and a simple quadratic. These test functions were rescaled so their domain was $[-100,100]^{10}$. Experiments were also run with a quadratic function, F24, that was rescaled along each coordinate axis: $F24(x) = 10\sum_{i=1}^{n}(i+1)^2 x_i^2$. This problem is also used over the domain $[-100,100]^{10}$.

Our experimental analysis considers two performance metrics for EPSA: the number of function evaluations until they terminate and the value of the best solution found. To provide a consistent method of comparison across different test functions, we ranked these performance metrics for different combinations of algorithmic factors. For example: to evaluate the impact of mutation offsets and mutation selection methods, we considered all combinations of the other algorithmic factors. For each combination of factors we ranked the results of all combinations of mutation offsets and mutation selection methods. These relative ranks provide a metric for evaluating the impact of only those factors that are being compared.

### 4.2 Results

#### 4.2.1 Effects of Crossover and Step Expansion

Figures 4 and 5 illustrate how the use of crossover uniformly increased the number of function evaluations required for convergence of the EPSA while providing better solutions. We examined the data further to identify interactions between crossover and mutation offsets. It seemed that crossover was most successful when used in conjunction with the regular simplex mutation offsets, typically producing the best-ranked solutions overall. Among the worst ranked results overall were those that used the regular mutation offset without crossover. However, neither of these interactions was consistent in all cases.

Using expansion consistently resulted in more function evaluations to convergence. However, there appeared to be little correlation between expansion and relative ranking of final solutions. Thus using expansion in EPSAs does appear to be helpful.

#### 4.2.2 Mutation Effects

Figures 7 and 8 illustrate the effect of varying the set of mutation offsets with the method of selecting mutation offsets. EPSAs generally terminated sooner when using multinomial selection of mutated components with the regular simplex mutation offsets. However, there is less of a distinct trend when we consider the relative rank of the final solutions. For the Rastrigin problem, using the regular mutation offsets was better, probably reflecting the strong coordinate bias in this problem. Also, the use of the regular simplex mutation offsets with binomial mutation seems to help the EPSA find the

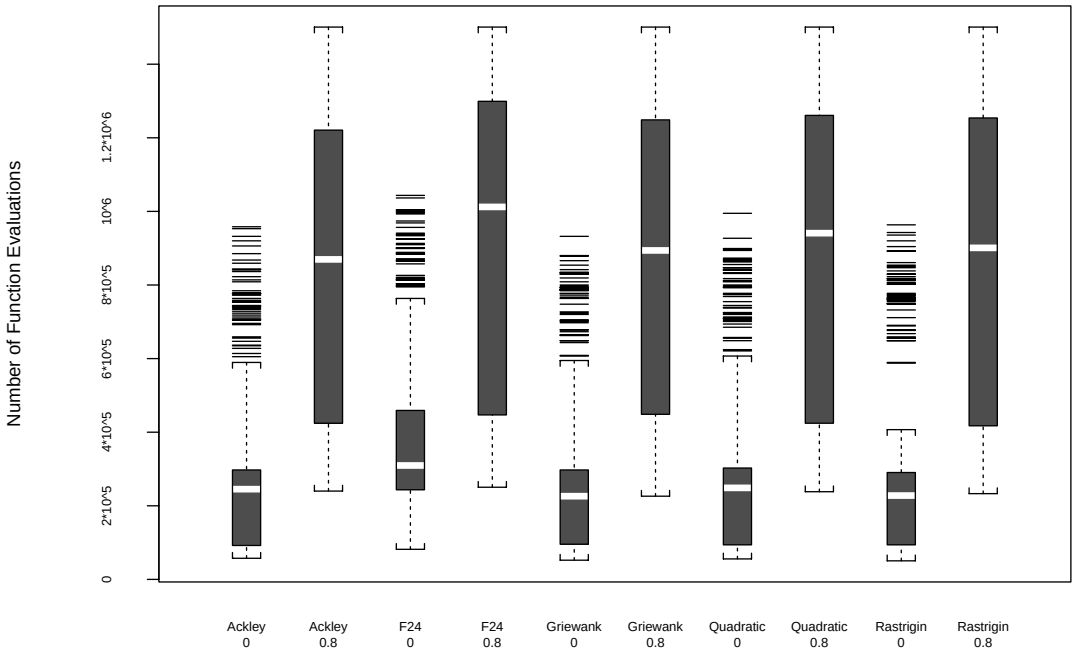


Figure 4: Number of function evaluations to convergence with and without crossover. Name of the test function appears above the crossover rate for plot.

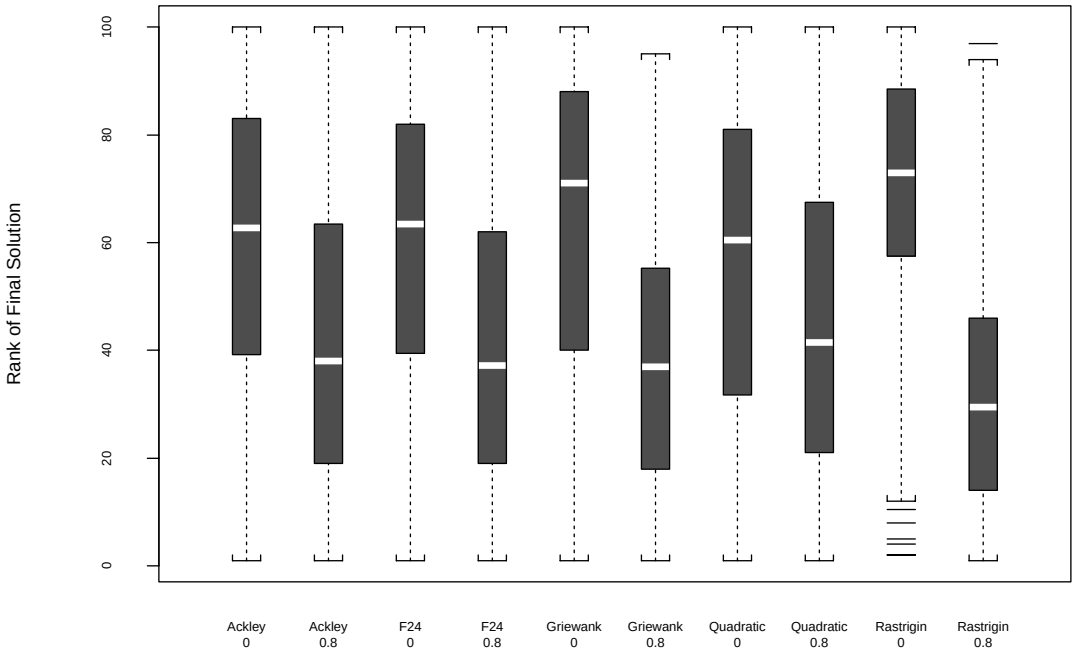


Figure 5: Relative rank of final solutions with and without crossover. Name of the test function appears above the crossover rate for plot.

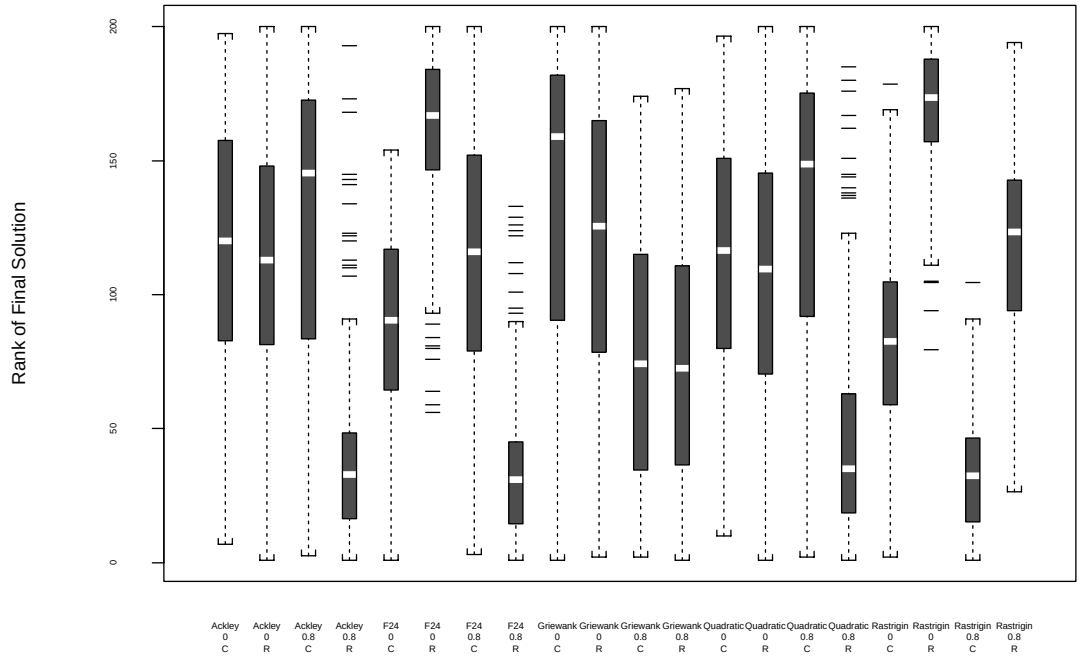


Figure 6: Relative rank of final solutions by mutation offsets with and without crossover. Name of the test function appears above the crossover rate with 1 below for regular mutation offsets and 0 below for standard mutation offsets.

best solutions in three of the five test cases, although not necessarily consistently so (e.g. see F24).

#### 4.2.3 Comparison with EP

Figure 9 shows the relative quality of the final solutions reached by EPSAs and EPs. We compare the final values from the EPSA using the regular simplex mutation offsets, no expansion, multinomial selection of mutation offsets, and using both crossover and noncrossover. We compare these results with the best results obtained by an EP. We consider the EP's results at three stopping points: (a) at the median number of function evaluations for the EPSA without crossover, (b) at the median number of function evaluations for the EPSA with crossover, and (c) at the maximum number of allowed function evaluations (700,000). Thus we can make direct comparisons between the EP to the EPSAs with and without crossover, as well as consider whether running the EP longer would ultimately find better solutions. In these results, the EPSA consistently outperforms the EP, even when the EP is allowed to run for substantially longer than the EPSA.

## 5 Conclusions

Our experiments confirm our previous observation that an EPSA using crossover can find better solutions, but at the expense of slower convergence [7]. Further, it is clear from these experiments that EPSAs converge more quickly if they can reduce the mutation step length while considering fewer mutation steps. These results suggest that EPSAs using the regular set of mutation offsets and the multinomial selection of mutated components are the most effective design for EPSA. These EPSA examine the fewest number of mutation offsets before reducing the step length, and in most of our problems this type of EPSA found solutions that are as good as any of the other EPSAs. Finally, the comparison between this EPSA and a canonical EP indicates that better solutions can be obtained by EPSAs with fewer function evaluations.

This algorithmic analysis can be extended in several ways. First, we need to consider other algorithmic factors, such as the way in which competitive selection is performed, to evaluate their effect on EPSAs. Further, we need to perform a broader comparison of EP and EPSAs on real applications to evaluate whether the performance difference here is robust. Similarly, these methods need to be compared on domains for which the near optimal points are not centrally located to evaluate the general effectiveness of crossover. We have also begun to extend these results to include an analysis of the robustness of convergence for EP and EPSAs, as well as their applicability to bound-constrained problems.

## Acknowledgements

We thank Kumar Chellapilla for the use of his EP code in this work. This work was supported by Sandia National Laboratories. Sandia is a multiprogram laboratory operated by Sandia corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

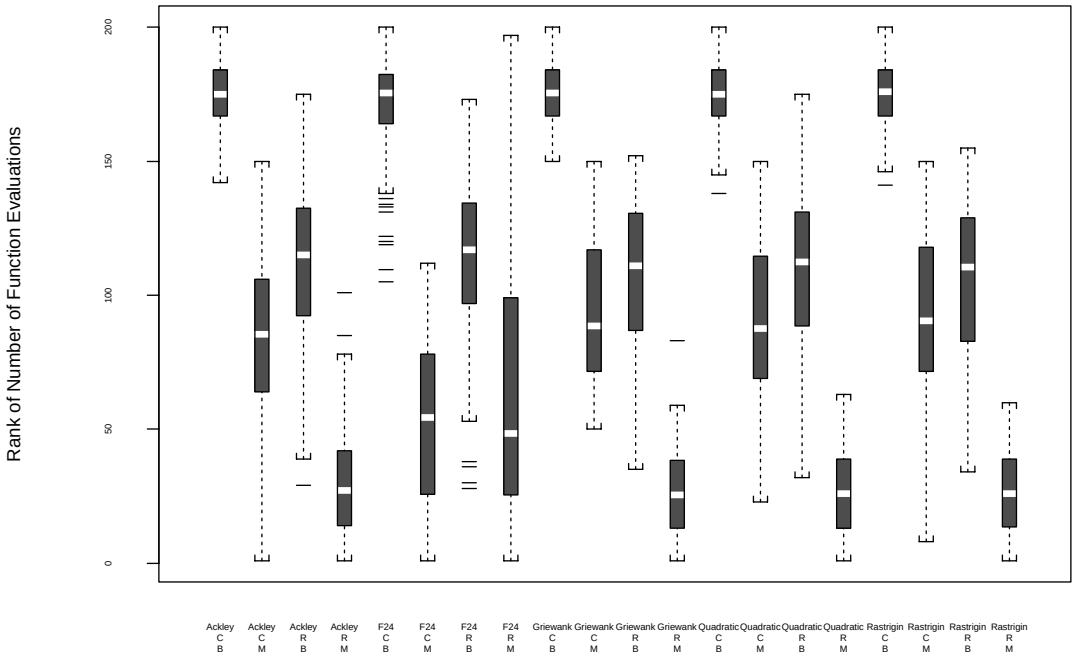


Figure 7: Relative rank of the number of function evaluations by combination of mutation offsets and selection of mutated components.

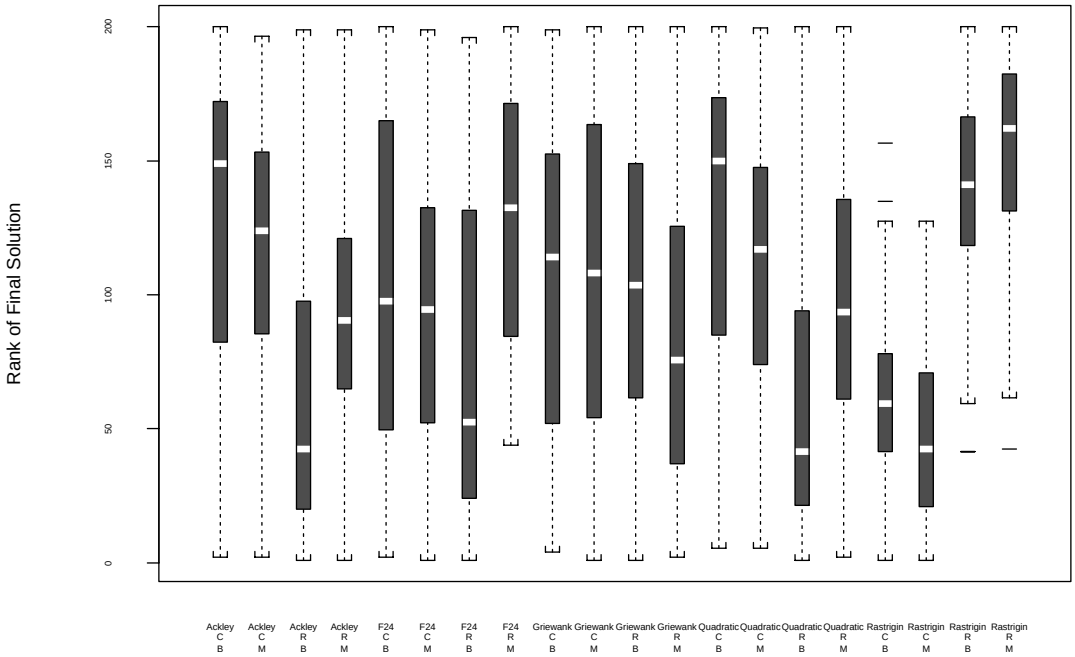


Figure 8: Relative rank of final solutions by combination of mutation offset and selection of mutated components.

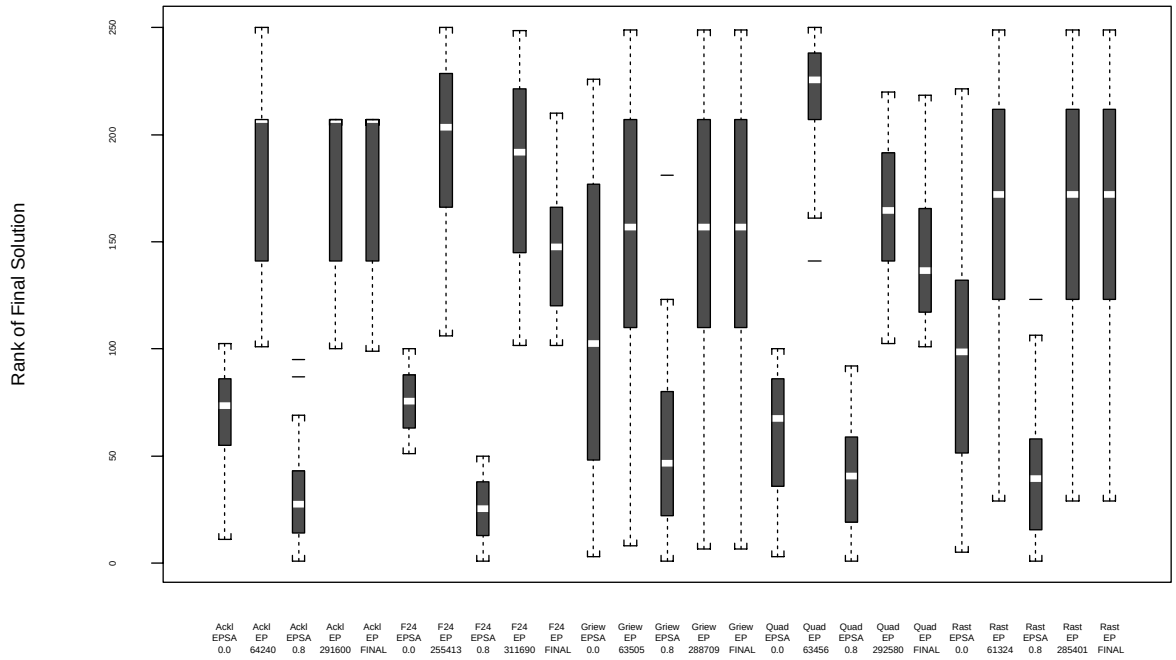


Figure 9: Relative performance of EP and EPSA. The function name appears above name of the optimization method. Bottom row consists of crossover rate (for EPSA plots) or number of function evaluations (for EP plots). FINAL denotes the EP terminated on maximum number of function evaluations. Comparisons between EPSA and EP results for the same number of function evaluations can be made by considering pairs of consecutive EPSA and EP results.

## Bibliography


[1] T. Bäck, G. Rudolph, and H.-P. Schwefel. Evolutionary programming and evolution strategies: Similarities and differences. In *Proc. of Second Annual Conf. on Evolutionary Programming*, pages 11–22, 1993.

[2] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[3] M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions, Second Ed.* John Wiley and Sons, Inc., New York, 1993.

[4] D. B. Fogel. *Evolutionary Computation*. IEEE Press, Piscataway, NJ, 1995.

[5] W. E. Hart. A generalized stationary point convergence theory for evolutionary algorithms. In T. Baeck, editor, *Proc 7th Intl Conf on Genetic Algorithms*, pages 127–134, San Francisco, CA, 1997. Morgan Kaufmann.

[6] W. E. Hart. A stationary point convergence theory for evolutionary algorithms. In R. K. Belew and M. D. Vose, editors, *Foundations of Genetic Algorithms 4*, pages 325–342, San Fransico, CA, 1997. Morgan Kaufmann Publishers, Inc.

[7] W. E. Hart. On the application of evolutionary pattern search algorithms. In *Proc Evolutionary Programming VII*, pages 303–312, New York, 1998. Springer-Berlin.

[8] W. E. Hart. A convergence analysis of unconstrained and bound constrained evolutionary pattern search. 1999. (submitted).

[9] J. Meza and M. L. Martinez. Direct search methods for the molecular conformation problem. *Journal of Computational Chemistry*, 15(6):627–632, 1994.

[10] N. Saravanan, D. B. Fogel, and K. M. Nelson. A comparison of methods for self-adaptation in evolutionary algorithms. *BioSystems*, 36:157–166, 1995.

[11] H.-P. Schwefel. *Evolution and Optimum Seeking*. John Wiley & Sons, New York, 1995.

[12] W. Spendley, G. R. Hext, and F. R. Himsworth. Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics*, 4(4):441–461, Nov 1962.

[13] V. Torczon, May 1996. Personal Communication.

[14] M. Wright. Direct search methods: Once scorned, now respected. In *Proc 1995 Dundee Biennial Conf in Numerical Analysis*, pages 191–208, Harlow, United Kingdom, 1996. Addison Wesley Longman.