Abstract

Two meta-evolutionary optimization strategies described in this paper accelerate the convergence of evolutionary programming algorithms while still retaining much of their ability to deal with multi-modal problems. The strategies, called directional mutation and recorded step in this paper, can operate independently but together they greatly enhance the ability of evolutionary programming algorithms to deal with fitness landscapes characterized by long narrow valleys. The directional mutation aspect of this combined method uses correlated meta-mutation but does not introduce a full covariance matrix. It is thus much more economical in terms of storage for problems with high dimensionality.

Step-recording is a subtle variation on conventional meta-mutational algorithms which allows desirable meta-mutations to be introduced quickly. Directional mutation, on the other hand, has analogies with conjugate gradient techniques in deterministic optimization algorithms. Together, these methods substantially improve performance on certain classes of problems, without incurring much in the way of cost on problems where they do not provide much benefit. Separately, their effect is not consistent.

The performance of these new methods on several standard problems taken from the literature is examined with a direct comparison made to conventional evolutionary algorithms. A new test problem is also introduced to highlight the difficulties inherent with long narrow valleys.

Directional Mutation for Faster Convergence

Ted Dunning
Computing Research Laboratory
New Mexico State University

25 September, 1995

1 Overview

1.1 Arguments for Meta-evolution

A number of stochastic optimization procedures have been developed since the electronic computer has made automated optimization possible. Methods which have had substantial recent development include simulated annealing, genetic algorithms and evolutionary programming. One shared property of each of these classes of algorithms is that they trade some degree of convergence speed for a decreased likelihood of avoiding locally optimal but globally suboptimal solutions.

In each of these well-known algorithms, there is a parameter which can be manipulated to affect this trade-off. In simulated annealing, this parameter is the simulated temperature, and in genetic algorithms and evolutionary programming, this parameter is the mutation rate. Typically, the temperature or mutation rate is decreased as the optimization progresses. This tactic substantially improves the rate of convergence often without significantly increasing the likelihood of finding a suboptimal solution. In special cases such as a quadratic bowl, cooling schedules can be derived which satisfy various theoretical constraints regarding the effort needed to have a given probability of finding a solution in a given amount of time, but in general this cannot be done since derivation requires detailed knowledge of properties of the function being optimized. Instead, an arbitrary and hopefully sufficiently conservative cooling schedule is typically invented and used.

An alternative to a fixed cooling schedule would be to derive the cooling schedule adaptively as the optimization algorithm learns about the fitness landscape that it is exploring. The idea that the mutation rate itself should be a parameter specific to each member of the population to be evolved is not new and has been explored in [Fog92] and [Atm]. This form of meta-evolution is attractive in that no explicit cooling schedule need be given.

1.2 Common problems

Problems whose solutions are found in long narrow valleys cause severe problems with evolutionary programming algorithm because the narrowness of the valley greatly decreases the probability of finding a solution which improves on a point which is already on the floor of the valley. These problems have

been attacked in the past by using a covariance matrix to cause mutations to be correlated as described in [Seb92]. This method is essentially similar to the conjugate gradient techniques used in conventional numerical optimization codes in that they concentrate the exploration of the fitness landscape in particular directions based on past experience.

Combining this method with meta-evolution as is done in the methods presented in this paper raises some interesting problems, however. In particular, in meta-evolution, all parameters which control mutation are included in the genome and must themselves be subject to mutation. But, if there are n real valued parameters in the genome initially, then it takes n^2 entries in a covariance matrix to describe how those n parameters should be changed. Including the covariance matrix in the genome increases its size to $n^2 + n$ real valued parameters and raises the serious question of how the mutation of the new parameters should be described. Conceivably, it might be necessary to include another, much larger, covariance matrix to describe mutation of the initial matrix. Following this logic, it is easy to envision a recursive explosion in the number of required parameters.

Including even the original n^2 in the description of each element of the entire population can be very expensive, even if meta-meta-mutation parameters are not included. Thus, it is desirable to find a more economical method for describing correlated mutation, and to find a way so that this correlated mutation description contains its own description of how it should be changed. One additional desiderata is that the meta-mutation be self similar so that the algorithm is insensitive to changes in scale. The two new strategies described in this report address this need.

2 Two New Strategies

2.1 Step Recording

In a conventional meta-mutation algorithm, the mutation rate for each member of the population is mutated independently of the state vector. This method can lead to slower convergence, especially in conjunction with directional mutation if a low probability step leads to improvement in fitness. If this happens, repeating a step like the one that caused the improvement can be advantageous. If the mutation rate (and possibly the mutation direction)

was changed independently of the fitness parameters, then similar steps will probably stay unlikely and convergence will be slow.

This situation will happen when a very fit solution in a small basin has been found due to taking a very large step. That particular solution will tend to remain in the population, but further improvement is unlikely unless subsequent small steps are taken. Even if an improved solution is found by taking a small step (which is unlikely, but it will happen eventually), it is likely that the mutation rate will still be large (which is why we found this solution in the first place), and further improvements will only come slowly as solutions with *both* better fitness and lower mutation rates are found.

With step recording, on the other hand, the mutation rate of an offspring is set to the magnitude of the distance between the parent and child solution. This means that any solutions which improve fitness by taking small steps will automatically lead to a line of progeny which tend to explore by taking small steps. Similarly, when direction mutation is combined with step recording, once a step is taken along the fitness gradient, further steps similar to that one are likely. This provides algorithms using directional mutation a sense of history in much the same manner as conjugate gradient methods use past history to improve further optimization efforts.

2.2 Directional Mutation

In order to fully characterize all of the possible correlations between n random variables, it is necessary to use roughly n^2 quantities. It does not, however, follow that a description this complete is necessary to gain the benefits of directional mutation. In particular, a covariance matrix specifies much more than just mutation biased in a particular direction. Indeed it can be argued that a covariance matrix contains much more information than can reasonably be extracted from the recent pedigree of a single member of a population.

Instead, we propose the use of a much more limited model of directional mutation in which the mutation rate has a directional component and an omnidirectional component. The total mutation is the sum of mutations derived from each of these components. The directional component of mutation is restricted to a line, while the omnidirectional component is sampled from a symmetric gaussian distribution. Together these components give a total mutation distribution which is an ellipsoidal gaussian distribution. In

a heuristic attempt to enhance convergence, the directional component is biased slightly.

If we use the notation $N(\mu, \sigma)$ to indicate a normally distributed random variable with mean μ and standard deviation σ and use $U(a, b)$ to indicate a uniformly distributed random variable taken from the half open interval $[a, b)$, then the following mutation algorithm suffices to provide a directional mutation of the vector $\mathbf{x}$

$$\lambda := N(1, 1)$$

For each x_i,

$$x_i := x_i + N(0, \sigma) + \lambda k_i$$

Here λ is a biased random variable which indicates how far to go along the direction indicated by $\mathbf{k}$, while σ provides the magnitude of the omnidirectional mutation component.

The mutation parameters $\mathbf{k}$ and σ can themselves be mutated by setting

$$\sigma := -(\sigma + |\mathbf{k}|/10) \log(-U(0, 1))$$

$$\lambda := N(1, 1)$$

and then for each k_i,

$$k_i := N(0, \sigma) + \lambda k_i$$

In this algorithm the mutation of σ is done by taking a new value from the exponential distribution with mean equal to σ augmented by a fraction of the magnitude of $\mathbf{k}$. This cross coupling between σ and $\mathbf{k}$ prevents the mutations from getting too directional. The mutation of $\mathbf{k}$ uses σ to provide diversity in direction and λ to provide diversity in magnitude in a manner identical with the way that the mutation of x_i was done.

It should be noted that the meta-mutation operation described here is self similar and orientation independent. This means that the distribution of mutation parameters after several generations in the absence of selection is invariant up to the scale and orientation of the original value. This means that the properties of the resulting meta-evolutionary algorithm are subject to analysis by renormalization methods.

To convert this algorithm to a step recording algorithm, the mutation of $\mathbf{x}$ is simplified and is done after the mutation of σ and $\mathbf{k}$. This simplified

mutation consists of setting $x_i := x_i + k_i$. The result is that $\mathbf{k}$ records the mutation step which was taken so that if this step results in an improvement, it is likely to be reused.

3 Experimental Methods

To test the efficacy of the proposed algorithms, all four combinations of conventional meta-evolution, meta-evolution with step recording, meta-evolution with directional mutation and conventional meta-evolution with both step recording and directional mutation were tested on three problems. These problems included a three dimensional symmetric quadratic bowl (function F1 from [Fog95]), a Bohachevsky multi-modal bowl problem (function F6 from the same work) and a very narrow two dimensional quadratic bowl whose axis was not aligned with either axis (labelled F9 here to avoid conflict with F1 through F8 from [Fog95]).

These test functions are described by the following functions:

$$f_1(x, y, z) = x^2 + y^2 + z^2$$

$$f_6(x, y) = x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7$$

$$f_9(x, y) = (x + y)^2 + (100y - 100x)^2$$

For this test, the evolutionary algorithm used 20 survivors each generation, each of which generated 9 progeny. After evaluating the fitness function for each member of the population, the entire population was sorted to find the best 20 members who would survive into the next generation.

Each algorithm was run 10 times and a median fitness at each generation was used to compare algorithms. All programs were limited to 50 generations or less. Generally, convergence to a solution with 10^-8 of the correct value was found within far fewer generations.

4 Results

The graph in 1 illustrates the convergence for the four algorithms for the symmetric bowl (Function F1). As can be seen, the convergence of the conventional meta-evolutionary strategy is slightly faster than for the modified

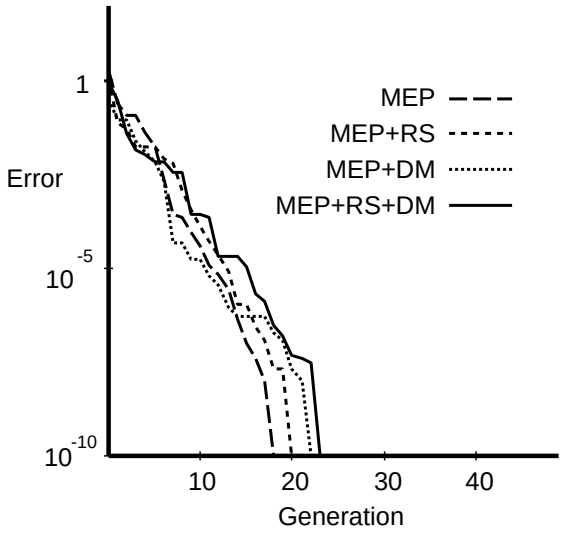

Figure 1: Convergence for Symmetric Bowl

algorithms, but the difference is not substantial. It is interesting to note that the omnidirectional mutation rate was a close approximation of square root of the remaining error. This behavior is close to the theoretical optimum cooling for this problem; that it was derived automatically by the meta-mutation was noteworthy.

The graph in 2 illustrates convergence for the Bohachevsky function. Again, the difference between the algorithms is not striking, except for the algorithm which used directional mutation without step recording. Even so, the degradation was less than a factor of two for directional mutation, and the loss in performance for the other methods was minimal.

Finally, the graph in 3 illustrates the convergence rates for the narrow quadratic bowl.

Here, all algorithms except for directional mutation with step recording have severe problems with convergence. The differences here are highly significant. The difference in the case of the non-directional algorithms is due to the fact that with symmetric mutation, if the mutation rate is much larger than the distance to the major axis of the valley, then any mutation is likely to fall outside of the narrow valley and thus does not result in any improvement. The result is that as solutions approach the major axis of the valley,

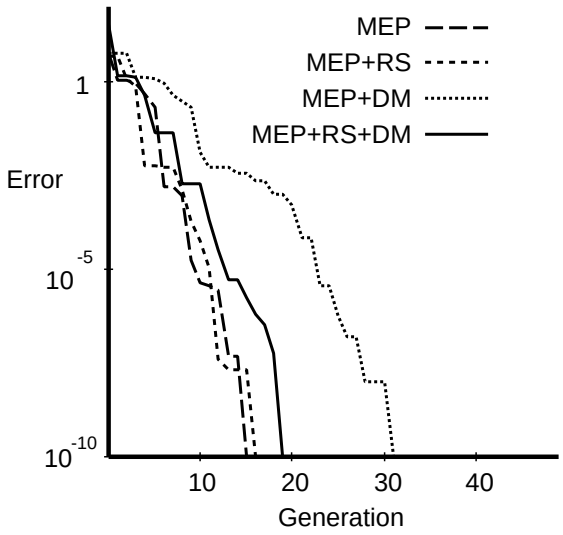

Figure 2: Convergence for Bohachevsky Function

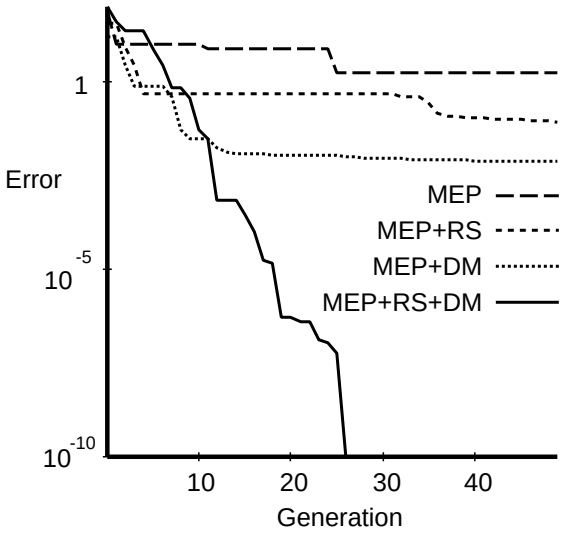

Figure 3: Convergence for Narrow Bowl

their mutation rate is decreased. Ultimately, solutions very close to the major axis are found, and the mutation rate is reduced to a very small number relative to the distance to the optimum.

It is not clear why directional mutation without step recording performs so poorly, but early experiments with different meta-mutation operators seemed to perform better, so the problem may have had more to do with the meta-mutation itself than with an inherent defect in the pure directional mutational algorithm.

The algorithm which used directional mutation and step recording performed very well. Detailed examination of evolving populations showed that populations far from the major axis of the valley quickly evolved directional mutations which took them to the major axis. Once there, the populations converted their directional mutations into omnidirectional mutations and then quickly into directional mutations oriented along the major axis of the valley. This quickly lead to bracketing of the solution at which point, the population size shrank rapidly.

It is instructive to compare these results with those from the table on page 173 of [Fog95]. The relevant parts of that table are reproduced in 1 and extended with the current results. Note that the meta-evolutionary algorithms (the new columns which are labelled MEP, MEP+RS and MEP+RS+DM) are clearly able to produce results which are comparable with previous evolutionary algorithms (the original columns which were labelled GA, DPE and EP in the original work). It should be remembered that when examining this table that comparing the various forms of evolutionary programming after such an extreme degree of convergence is not terribly meaningful.

5 Summary and Discussion

This work clearly shows that meta-evolutionary strategies can be effective in accelerating the convergence of evolutionary programming algorithms under certain conditions.

Furthermore, the directional mutation and recorded step do not significantly degrade this performance on simple problems. They can provide highly signficant improvement in convergence speed on problems which involve long narrow valleys.

The directional search algorithm presented here has a number of clear

Function	GA	DPE	EP
F1	2.8×10^{-4}	1.1×10^{-11}	3.1×10^{-66}
F6	2.629×10^{-3}	1.479×10^{-9}	5.193×10^{-96}

Function	MEP	MEP+RS	MEP+RS+DM
F1	3.3×10^{-71}	3.2×10^{-125}	1.5×10^{-51}
F6	0	0	0

Table 1: Convergence of Various Evolutionary Algorithms (GA = Genetic Algorithm, DPE = GA with Dynamic Parameter Estimation, EP = Evolutionary Programming, MEP = Meta-Evolutionary algorithms, RS = Recorded Step, DM = Directional Mutation)

advantages over carrying a full covariance matrix with each member of the population. These include lower storage requirements, lower computational overhead, and an intuitively appealing method for doing meta-mutation.

Although the algorithms describe here perform well on moderately multimodal problems such as the Bohachevsky functions, they probably trade off some of the ability to avoid locally optimal solutions in return for their ability to explore narrow valleys. This ability may need to be recovered for some problems. One way that this might be done is to allow only parent/progeny competition. This would help avoid the situation where a solution is found which is good enough to swamp the survivor pool with progeny before more obscure solutions are found. Another method for attacking this problem would be introduce cultural algorithms.

In partially or wholly decomposable problems with high dimensionality, narrow valleys can occur which are aligned with the axes rather than aligned arbitrarily. In these cases, it the cost of the directional mutation algorithm given here might be better spent by keeping a separate mutation rate for each dimension. Each of these mutation rates could be subjected to the self-similar exponential mutation described in this paper. Another option would be to keep the directional mutation, and expand the omnidirectional mutation rate to one mutation rate per parameter. Whether either of these changes would actually enhance performance is an open question.

References

[Atm] David B. Fogel; Larry J. Fogel; J. Wirt Atmar. Meta-evolutionary programming. In R.R. Chen, editor, *Proceedings of the 25th Asilomar Conference on Signals, Systems and Computers*, pages 542–545. Pacific Grove, CA.

[Fog92] David B. Fogel; Larry J. Fogel; J. Wirt Atmar; Gary B. Fogel. Hierarchic methods of evolutionary programming. In *Proc. of the First Annual Conference on Evolutionary Programming, Evolutionary Programming Society, San Diego, CA*, 1992.

[Fog95] David B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, New York, NY, 1995.

[Seb92] A.V. Sebald. On exploiting the global information generated by evolutionary programs. In *Proc. of the First Annual Conference on Evolutionary Programming, Evolutionary Programming Society, San Diego, CA*, 1992.