The Dynamics of the Best Individuals in Co-evolution

Elena Popovici and Kenneth De Jong
George Mason University

October 5, 2005

Abstract.

There continues to be a growing interest in the use of co-evolutionary algorithms to solve difficult computational problems. Their performance however has varied widely from good to disappointing. The main reason for this is that co-evolutionary systems can display quite complex phenomena. Therefore, in order to efficiently use co-evolutionary algorithms for problem solving, one must have a good understanding of their dynamical behavior. To build such understanding, we have constructed a methodology for analyzing co-evolutionary dynamics based on trajectories of best-of-generation individuals. We applied this methodology to gain insights into how to tune certain algorithm parameters in order to improve performance.

1. Introduction

There continues to be a growing interest in the use of co-evolutionary algorithms to solve difficult computational problems. The results to-date have been mixed, primarily because co-evolutionary algorithms (CoEAs) can behave quite differently than the simpler and more familiar evolutionary algorithms (EAs). The goal of this research is to understand these differences in a way that improves our ability to design successful co-evolutionary computation (CoEC)[1] applications.

Perhaps the most important difference between EAs and CoEAs is the increased complexity of the CoEA dynamics. This paper describes a technique involving trajectories of best-of-generation individuals and "best-response curves" that provides important insights into the co-evolutionary dynamics that affect problem-solving performance. The usefulness of this technique is illustrated by looking at a common application area, namely, the use of cooperative co-evolutionary algorithms to solve difficult function optimization problems. First, cooperative co-evolution is applied to two familiar test functions in a straightforward manner, the results of which are ok but not great. Then, several standard EA design heuristics are used in an attempt to improve CoEA performance with mixed and somewhat surprising results. However, the use of best-of-generation trajectories and best-response curves to analyze the co-evolutionary dynamics provides the insights necessary to explain the behavior observed when applying the EA heuristics and to improve CoEA performance.

Although the paper focuses on the use of best-of-generation trajectories and best-response curves to analyze a cooperative co-evolutionary application, this technique can be used in a similar manner to better understand other co-evolutionary applications as well (e.g., competitive co-evolution).

2. Background

Historically, the two primary application areas of CoEC have been: 1) the use of a competitive co-evolutionary framework to evolve behaviors in competitive environments (Hillis, 1990; Angeline and Pollack, 1994; Rosin and Belew, 1995), and 2) the use of a cooperative co-evolutionary framework to solve difficult function optimization problems (Potter and De Jong, 1994). In both cases the additional complexity of the co-evolutionary framework poses a number of difficulties for the CoEC practitioner that go beyond those encountered in standard EC applications. This has resulted in a good deal of research on CoEC systems, the result of which is a much better understanding of how they work and how to apply them.

Perhaps the most difficult aspect of CoEC design is controlling its dynamics. As a consequence, much of the CoEC research has focused on techniques that provide a deeper understanding of CoEC from a dynamical systems perspective. One approach is to use evolutionary game theory (Hofbauer and Sigmund, 1998) as the basis for CoEC analysis (see, for example, (Ficici et al., 2000) or (Cliff and Miller, 1994)). Others have used dynamical systems theory (Alligood et al., 1996) as their analysis tool (e.g., (Bucci and Pollack, 2002; Cliff and Miller, 1995; Ficici and Pollack, 1998)). Still others have focused more on understanding the differences between EC and CoEC dynamics (such as (Luke and Wiegand, 2003; Popovici and De Jong, 2005c; Pagie and Mitchell, 2002)).

Each of these approaches provides useful, but incomplete insights into CoEC dynamics. Taken collectively, they provide a much deeper understanding of CoEC design than was available even a few years ago. But, there is still a big gap between theory and practice. The practitioner needs more than just characterization of EC dynamics. Rather he/she needs to be able to use these insights to make design decisions that improve performance. In the remainder of this paper we describe a new technique involving "best-response curves" that appears to have these properties: namely, it provides useful insights into the CoEC dynamics that can be used to improve CoEC problem-solving performance.

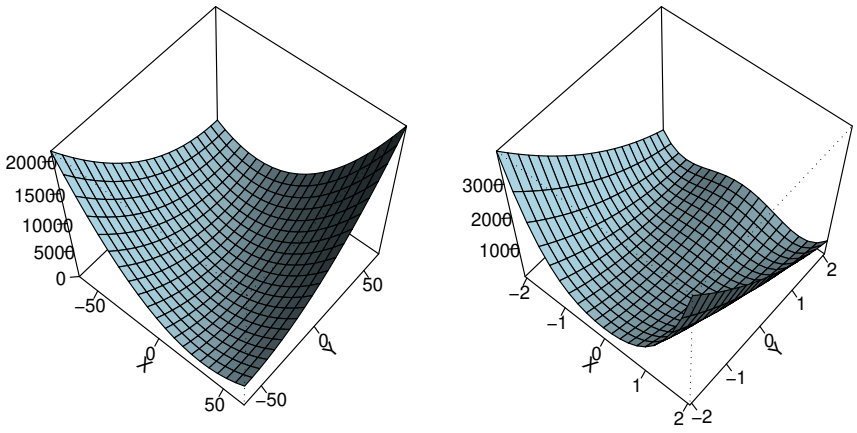

Figure 1. Left: $offAxisQuadratic$. Right: *rosenbrock*

3. An Example

For our illustrative examples we have selected two standard test functions from from the EC function optimization literature, *rosenbrock* and $offAxisQuadratic$. We reproduce them below for completeness:

$$offAxisQuadratic(x, y) = x^2 + (x + y)^2,$$
$$x, y \in [-65.536, 65.536];$$
$$rosenbrock(x, y) = 100(x^2 - y)^2 + (1 - x)^2,$$
$$x, y \in [-2.048, 2.048].$$

Figure 1 shows the two-dimensional surfaces described by the functions. Our task is to find the pairs (x, y) in which the functions reach their minimum value (zero in both cases). For $offAxisQuadratic$ this is the point $(0, 0)$ and for *rosenbrock* it is $(1, 1)$.

From a traditional EC perspective, these landscapes are similar with respect to properties such as continuity, modality, ruggedness, etc. But, as we shall see, there are important differences when using co-evolutionary algorithms to optimize them.

The standard way of attacking such problems using cooperative coevolution is to decompose the problem "naturally" by defining each argument of the function to be a subcomponent to be evolved in a separate population (Potter and De Jong, 1994). For our landscapes this results in two populations, one evolving values for the x function parameter and the other evolving values for the y parameter. We use a "vanilla" CoEA that alternates populations and uses a single-best collaboration strategy and symmetric payoff (Wiegand, 2004). In each population we use a non-overlapping-generation EA with a real-valued representation, binary tournament selection, and a Gaussian mutation

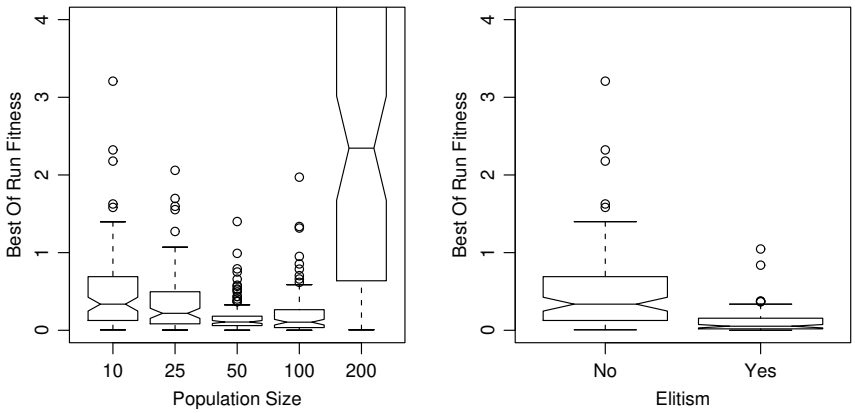

Figure 2. $offAxisQ$ – Effects of population size and elitism.

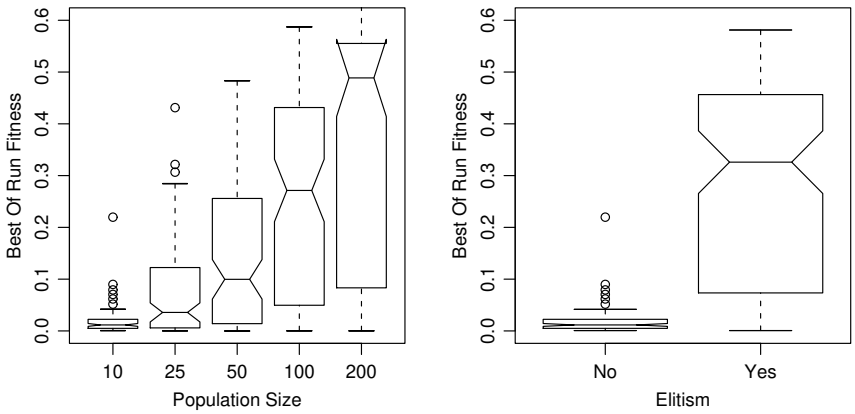

Figure 3. rosenbrock – Effects of population size and elitism.

operator with fixed sigma. More specific details of the algorithm can be found in the Notes section at the end of the article[2].

As is fairly standard with EA applications, we now go through a short tuning exercise to improve the performance of our CoEA. In this illustration we experiment with different population sizes and with the inclusion of elitism. What we obtain is a rather surprising set of results as illustrated in Figures 2 and 3. These figures show the distribution over 100 runs of best-of-run fitness values in the form of boxplots[3].

On the $offAxisQ$ landscape, introducing elitism greatly improves performance and increasing the population size has the effect of first increasing performance and then decreasing it. This is exactly what we would expect based on our experience with standard EAs. However,

on the *rosenbrock* landscape, introducing elitism resulted in significant decreases in performance, as did increasing population size! Clearly, there is something different about how our CoEA works on these two landscapes. We analyze this difference in the remainder of this paper.

4. The Importance of Co-evolutionary Dynamics

It is our belief that an important key to successful CoEC design is a better understanding of the co-evolutionary dynamics exhibited by the interplay of the co-evolving populations. We have developed a technique that helps significantly in this respect. This technique was first introduced in (Popovici and De Jong, 2004) and has been improved since then. We have already successfully used it to gain insight into the way co-evolutionary algorithms work, both in competitive (Popovici and De Jong, 2005b) and cooperative setups (Popovici and De Jong, 2005a; Popovici and De Jong, 2005c). In this paper we describe this technique and use it to extend our previous results.

4.1. Method

Dynamical systems theory has proved to be a good source of inspiration for constructing methods of analysis for co-evolutionary algorithms. In the literature the term *co-evolutionary dynamics* generally refers to population-level dynamics. For cooperative settings, (Wiegand, 2004) introduced a technique for tracking the percentage of the best individual in the population. For a particular type of competitive settings (namely, frequency based evolution) (Ficici et al., 2000) analyzed trajectories of the population state.

In this paper we use the term to refer to dynamics of *individuals* rather than population(s). Specifically, we analyze the *time trajectories of best-of-generation individuals across the search space*. There are two reasons for this. First of all, we are trying to analyze the performance of cooperative co-evolution for optimization, where the main concern is with the best individuals the algorithm produces. Second, we wanted to understand the behavior of a basic CoEA before moving to more complex ones, and the simplest one out there (that we picked for our initial experiments) uses a *single best* collaboration strategy for evaluation. The analysis of such trajectories exposed what we believe is a problem property of relevance to co-evolutionary setups, which we named *best-response curves*.

4.1.1. *Best-of-Generation Trajectories*

The fact that our basic CoEA alternates populations suggests a *line-search*-like way of operation. It therefore makes sense to plot the best individual of one generation (and therefore one population) by coupling it with its collaborator during fitness assessment (here, the best individual of the other population at the previous generation). We thus obtain one point of the search space per generation and we connect these points

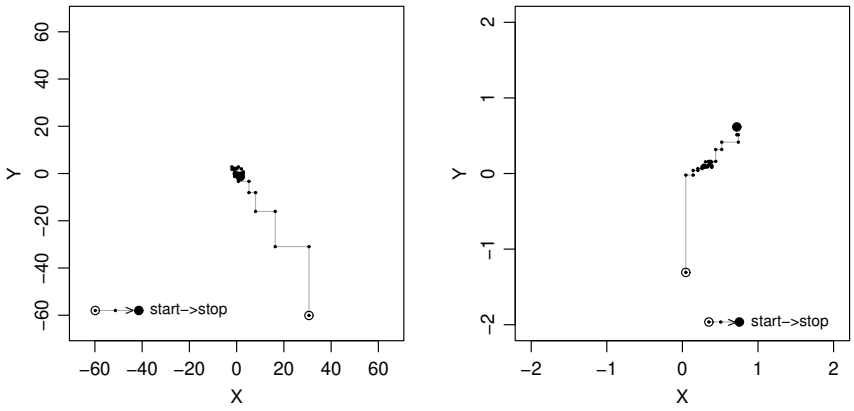

Figure 4. Best of generation trajectory for sample runs. Population size 50, no elitism. Left: $offAxisQuadratic$. Right: *rosenbrock*

chronologically. It is obvious that the lines connecting them will only be vertical (when connecting an X generation with the following Y generation) and horizontal (when connecting a Y generation with the following X generation).

An example of the result of this procedure for a single run for both $offAxisQuadratic$ and *rosenbrock* can be seen in Figure 4. The trajectory is displayed as grey lines connecting black dots (each dot representing one generation). The starting point is marked by an empty geometrical figure (in this case a circle) and the end point is marked by the same figure but filled.

Since, for each setup, we ran 100 runs, we end up with 100 such pictures. Although, as we will see later, there is value in looking at individual runs, for now we would like to get a combined view of all of them. For this purpose, we superimpose all runs of a setup on a single plot. For better visibility, we no longer draw the lines connecting the generations; instead, we use different plotting symbols for X generations and Y generations. We use the resulting images to compare the two functions. Figure 5 shows cumulative best-of-generation plots for population size 10 without elitism, population size 10 with elitism and population size 200 without elitism.

These pictures vividly show that there are some areas of the search space that strongly attract the best-of-generation individuals. Moreover, these areas have very regular shapes. In particular, for these two domains, they seem to be lines/curves. We also notice that the X best-of-generation individuals "draw" different patterns from the Y best-of-generation individuals. In the following sections we focus

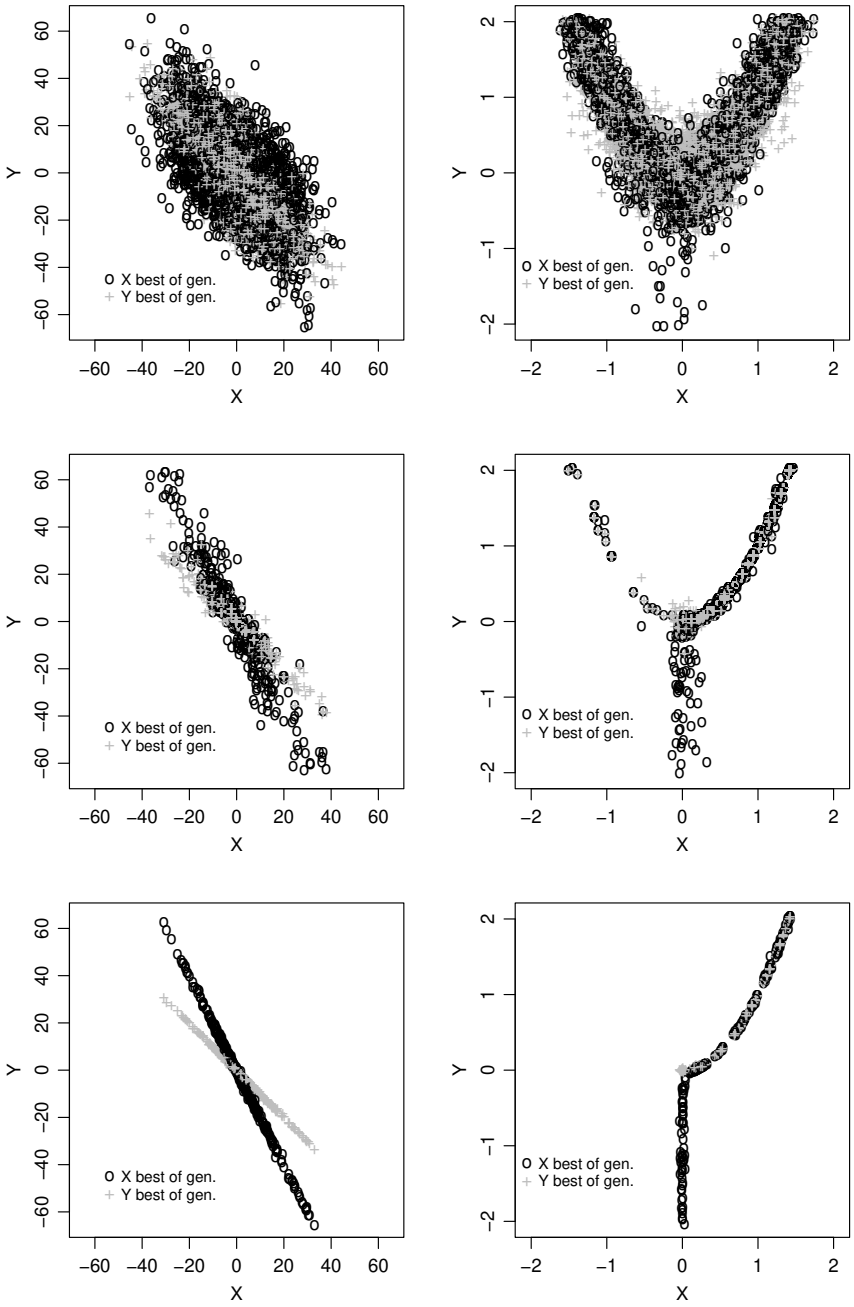

Figure 5. Cumulative best-of-generation points. Left: $offAxisQuadratic$. Right: *rosenbrock*. Top row: population size 10 without elitism; middle row: population size 10 with elitism; bottom row: population size 200 without elitism.

on understanding what these patterns are and why best-of-generation individuals are going there.

4.1.2. *Best-Response Curves*

Let us take another look at the way the algorithm works. It alternates populations and in each generation the active population is evaluated in combination with a single individual (the best) from the opposite population, then evolved and re-evaluated in combination with that same individual. This is like a one-dimensional search in a slice through the domain, where the active population is searching for an individual that gives the best (in this case minimal) function value in combination with the current best individual of the other population. In other words, the active population is trying to give the *best response* possible to the best reported by the frozen population.

Here is a more formal description. Suppose we are working with a function $f : D_X \times D_Y \rightarrow R; D_X, D_Y \subset R$. Let the current active population be the X population and the current best individual in the Y population be y_0. Then the X population is searching for an individual $x^*_{y_0}$ such that $f(x^*_{y_0}, y_0) = min_{x \in D_X} f(x, y_0)$. For every $y \in D_Y$ an x^*_y individual exists, regardless of whether the algorithm finds it or not. In general, there may be more than one, but in this paper we deal only with functions for which for every $y \in D_Y$ there is a unique x^*_y. We can therefore define a function $bestResponseX : D_Y \rightarrow D_X, bestResponseX(y) = x^*_y$. A $bestResponseY : D_X \rightarrow D_Y$ function can be similarly defined (under the same assumption of uniqueness of best responses). These functions are a property of the domain f and do not depend on any algorithm.

To obtain the formula for $bestResponseX(y)$, we need to solve the equation $\frac{\partial f(x,y)}{\partial y} = 0$ for variable y. Similarly, in order to compute $bestResponseY(x)$, we need to solve $\frac{\partial f(x,y)}{\partial x} = 0$ for variable x.

For $offAxisQuadratic$ we get: $\frac{\partial (x^2+(x+y)^2)}{\partial y} = 2(x+y) = 0 \Rightarrow y = -x$ and $\frac{\partial (x^2+(x+y)^2)}{\partial x} = 2x + 2(x+y) = 2(2x+y) = 0 \Rightarrow x = -y/2$. Thus, for $offAxisQuadratic$, the formulas for the best-response curves are: $bestResponseX(y) = -y/2$ and $bestResponseY(x) = -x$. So they are the lines of equations $x + y = 0$ and $2x + y = 0$ that intersect at point $(0, 0)$, which is where the function reaches its minimum.

For $rosenbrock$ the situation is somewhat more complicated. We have $bestResponseY(x) = \begin{cases} 2.048 & \text{if } |x| \in [\sqrt{2.048}, 2.048]; \\ x^2 & \text{if } |x| \in [0, \sqrt{2.048}]. \end{cases}$ Obtaining $bestResponseX(y)$ requires solving a cubic equation, and we do that graphically by interpolation rather than by mathematical formula.

Once again, the two best-response curves intersect in a single point, $(1, 1)$, which is also where the function reaches its minimum.

All best-response curves are shown on the top row of Figure 6. Note that for *rosenbrock* the two best-response curves seem to overlap for some part. In fact they only do so in $(1, 1)$, but for $x \in [1, \sqrt{2.048}]$ the distance between $bestResponseY(x)$ and $bestResponseX^{-1}(x)$ is less than 10^{-3}.

The two bottom rows show an intuitive geometrical view of how these curves relate to slices through the two-dimensional surfaces described by the functions. In the left column we illustrate the construction of the $bestResponseY(x)$ curve for $offAxisQuadratic$. The plot in the middle row shows five curves (plotted in different line styles) which were obtained by slicing through the two-dimensional surface of $offAxisQuadratic$ with different values for X. For each such slice-curve, we determine the Y-value for which the curve reaches its minimum (best), plot a symbol at this point and draw a vertical line to better show the Y-value. What we have done is identify $bestResponseY(x)$ for each of the five X values considered. On the bottom plot we display the points $(x_i, bestResponseY(x_i)), i \in \{1, 2, 3, 4, 5\}$ (using the same symbols as above) and how they fit on the $bestResponseY(x)$ curve. The right column shows the similar process for $bestResponseX(y)$ for $rosenbrock$.

The resemblance of the plots in Figure 5 to the best-response curves is striking. The X best-of-generation individuals were drawing pieces of the $bestResponseX(y)$ curve and the Y best-of-generation individuals were drawing pieces of the $bestResponseY(x)$ curve[4]. We begin to unravel the mystery of what our co-evolutionary algorithm is doing.

4.2. Application

We return to our initial goal, namely to understand why population size and elitism have different effects on the two functions. We also return to investigating individual runs. This time however, we combine on the same plot the best-of-generation trajectories as initially described in section 4.1.1 with the best-response curves in order to see the relationship between the two and how this relationship is affected by the various changes in parameters.

Figures 7 and 8 show sample runs for all the setups we have experimented with for both functions. Each plot shows 1, 2 or 3 sample runs from the 100 that were carried out for that particular setup. We tried to pick typical samples, while still showing the breadth of behaviors possible and at the same time keeping the images readable. Each run

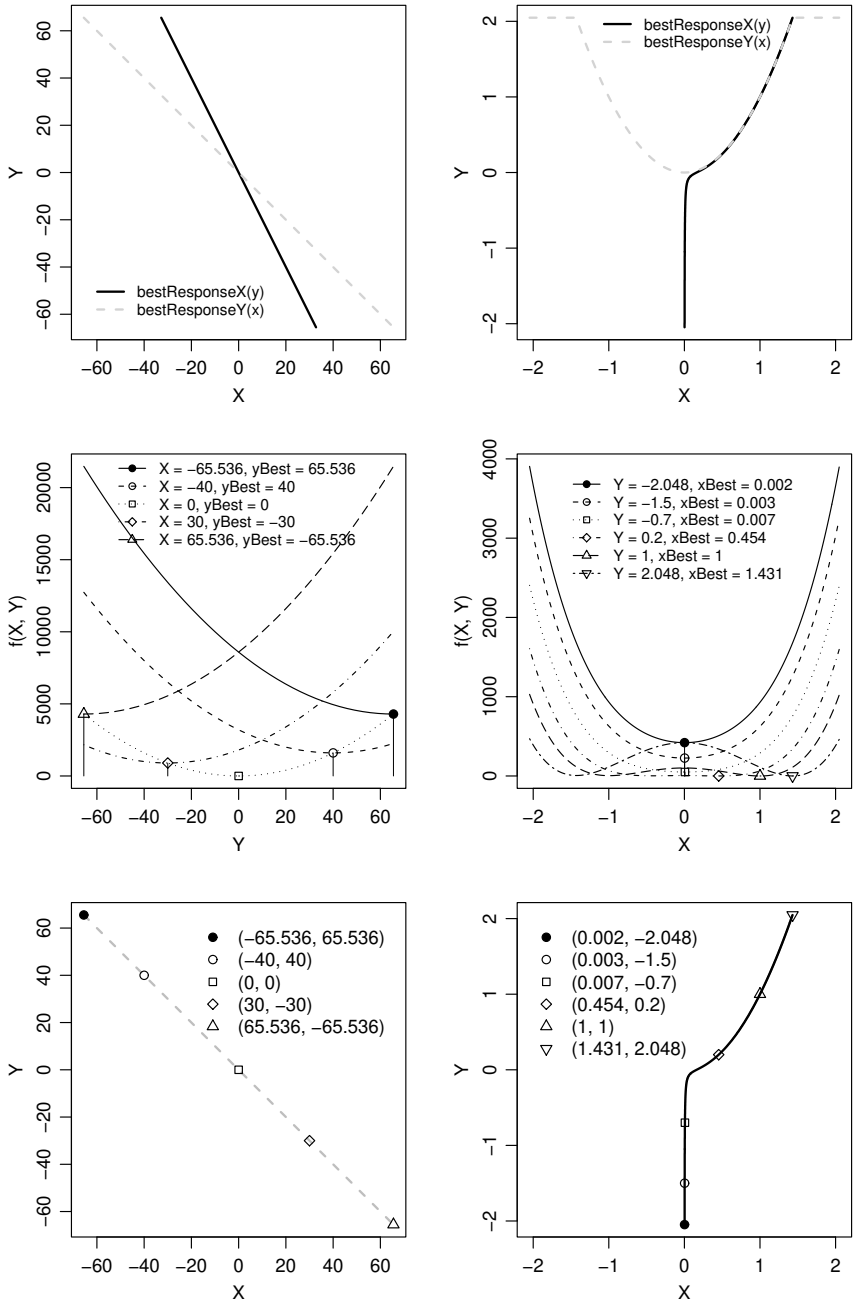

Figure 6. Best-response curves and slices. Left: $offAxisQuadratic$; top to bottom: best-response curves, slices with X, $bestResponseY(x)$ with points determined from slices. Right: *rosenbrock*; top to bottom: best-response curves, slices with Y, $bestResponseX(y)$ with points determined from slices.

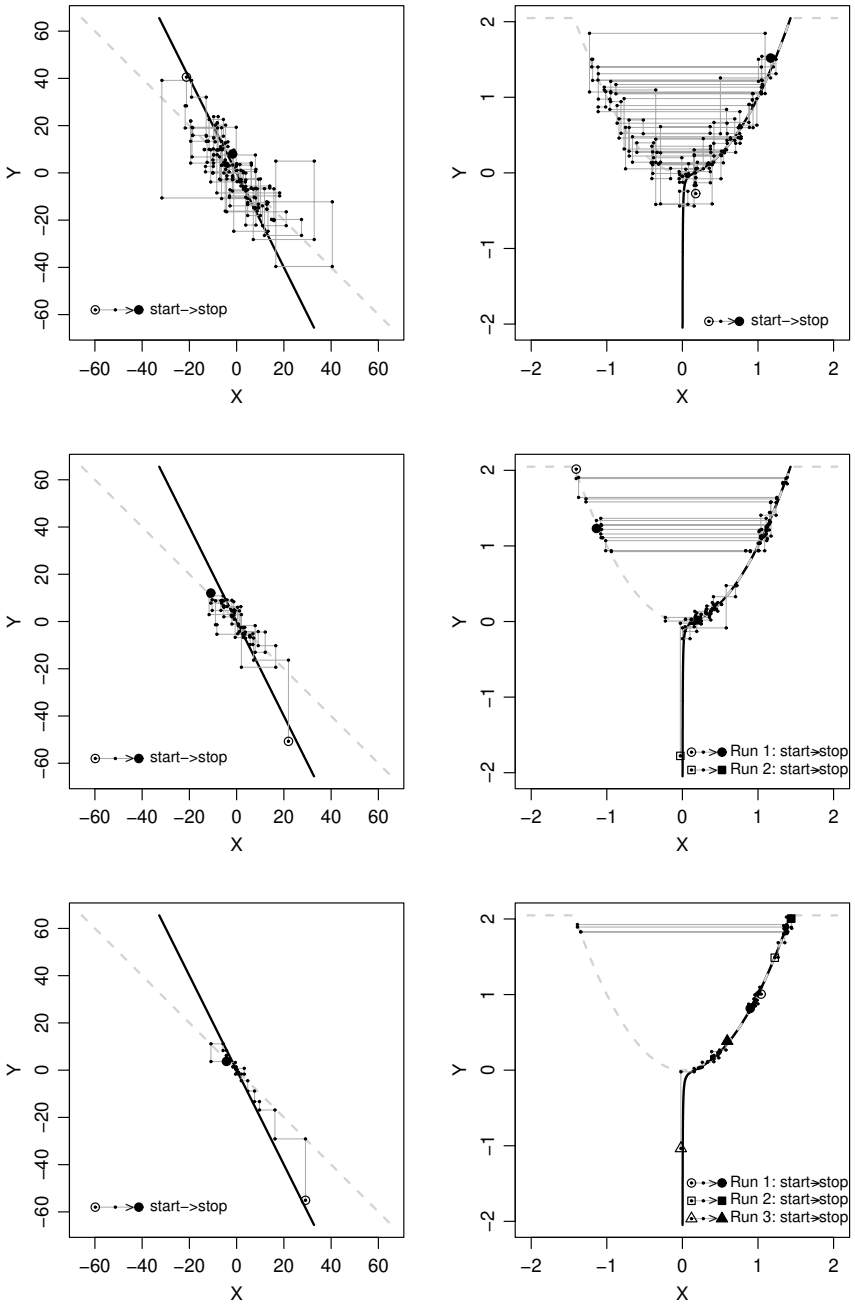

Figure 7. Best-of-generation trajectories. Left: $offAxisQuadratic$. Right: *rosenbrock*. Top row: population size 10 without elitism; middle row: population size 25 without elitism; bottom row: population size 50 without elitism.

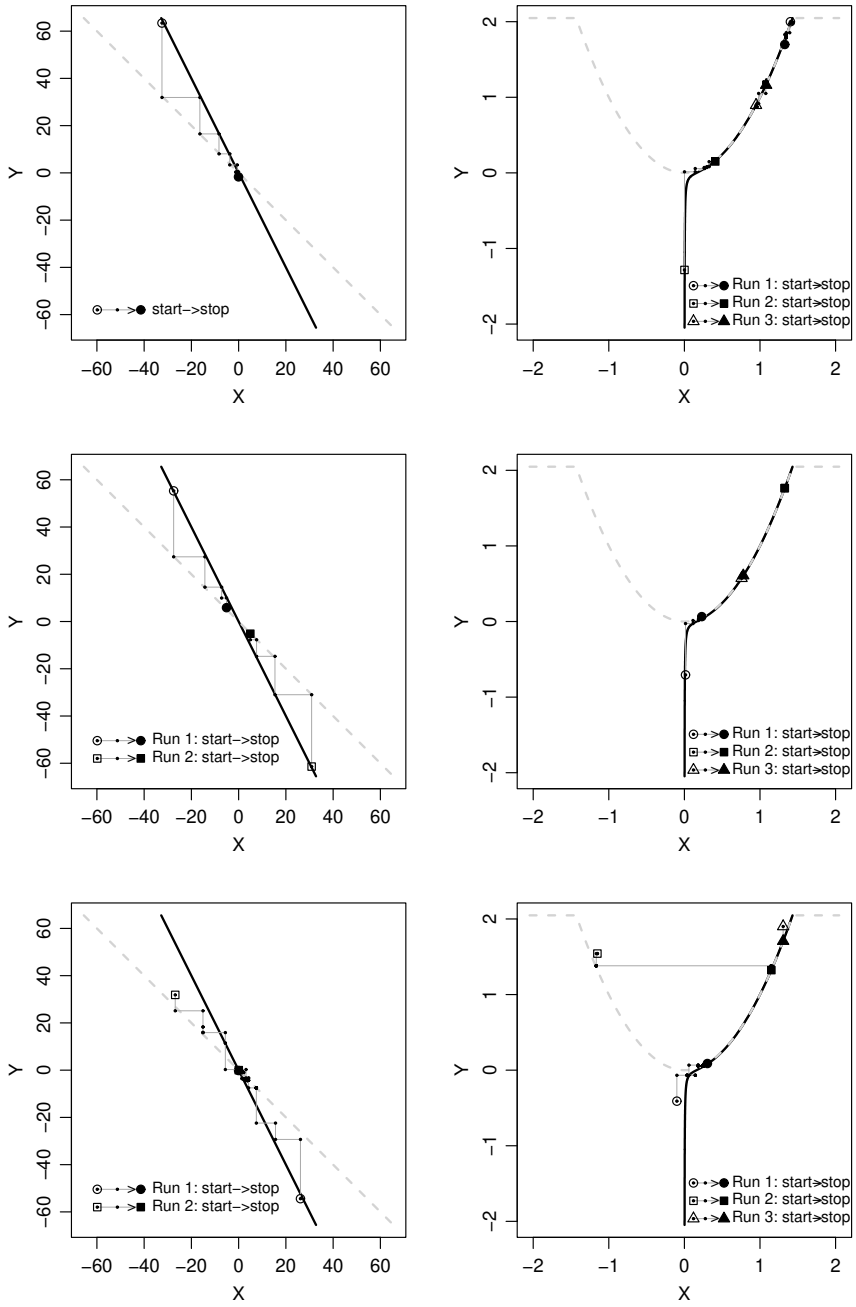

Figure 8. Best-of-generation trajectories. Left: $offAxisQuadratic$. Right: *rosenbrock*. Top row: population size 100 without elitism; middle row: population size 200 without elitism; bottom row: population size 10 with elitism.

has a different geometrical figure to denote the beginning and end of the run.

Let us first investigate the effects of increasing the population size in the absence of elitism. For both functions we see that with population size 10, the trajectories, while clearly influenced by the best-response curves, deviate from them widely. Increasing the population size causes the best-of-generation trajectories to closer and closer follow the best-response curves. This makes sense, as with a small population size, during one generation the evolutionary process is unlikely to come up with the exact best response. Increasing the population size increases the precision with which the algorithm approximates the best response.

While this underlying phenomena is the same, its effects are different on the two functions. For $offAxisQuadratic$, due to the relative positions of the best-response curves, a deterministic system exactly following them would advance like on a ladder towards the intersection point. With large population sizes, our algorithm closely approximates this behavior, while with small population sizes, the algorithm gets "distracted" from it. The intersection point is where the function reaches its minimum; for both best-response lines, the function value decreases from the ends towards the middle. We now understand the initial gradual improvement in performance shown by the boxplots on the left side of Figure 2. But why does performance start to decrease again as we further increase the population size? Remember we are using a fixed number of evaluations, so larger population size means fewer generations. There is therefore a tradeoff between the accuracy of following the best-response curves and the number of steps taken along them. If the number of steps becomes too small (e.g. population size 200 x 10 generations), the algorithm doesn't have enough time to reach the optimum, the target towards which it is so precisely moving. This can be seen in the middle-left plot of Figure 8.

For *rosenbrock*, the best-response curves are such that the trajectory of a deterministic system exactly following them would after the first or second step enter the region of almost-overlap and then be forced to take extremely small steps. Once again, the intersection point of the two best-response curves is where the function reaches its minimum. With small population size, the algorithm's best-of-generation trajectory has low accuracy in following the best-response curves, thus taking large steps rather than small ones. Additionally, the large number of steps that are available for a small population size make the trajectory be highly explorative and thus inevitably visit points close to the optimum. With increasing population size, the accuracy of approximating the best-response increases and this means smaller step size. Moreover, the number of steps decreases and these two effects combined make the

trajectory of best-of-generation individuals crawl and/or very quickly get stuck in a point close to where it (randomly) started. The algorithm will get close to the optimum only if by chance it started close to it. In this case high accuracy doesn't balance off a small number of steps, but instead works to the disadvantage of performance as well.

We now turn to elitism and analyze the bottom row of Figure 8. Since elitism does not decrease the number of generations, we would expect to see the same number of points as in the corresponding plots of population size 10 without elitism (top row of Figure 7). Instead, the elitism plots show very few points. What happens is that elitism prevents the algorithm from making a move to a best individual that is worse than the previous best. Therefore, the trajectory stays put until a new best is found. In general, increasing fitness is done by moving closer to the best-response curve of the current population (subcomponent). The algorithm thus stays focused on following the best-response curves. As mentioned in the discussion about population size effects, following the best-responses is good for $offAxisQuadratic$ and bad for $rosenbrock$, thus explaining the boxplots on the right side of Figure 3.

5. More Best-Response Effects

At this point we have some intuition about the effects that population size and elitism have on performance in the presence of certain problem features. We further investigate this by constructing a family of functions for which we can control the degree of proximity of the best-response curves by varying one parameter, which we called α and which takes values between 0 and 1.

The family of functions is defined as follows:

$$br_n^\alpha(x,y) = \begin{cases} 2y + \frac{\alpha-3}{2\alpha}(x-n) & \text{if } \alpha y < x + (\alpha-1)n; \\ 2x + \frac{\alpha-3}{2\alpha}(y-n) & \text{if } y > \alpha x + (1-\alpha)n; \\ n + \frac{x+y}{2} & \text{otherwise.} \end{cases}$$

$n \in N; \alpha \in [0,1]; x, y \in [0,n]$

and the two-dimensional surfaces described by the functions for $n = 8$ and $\alpha \in \{0, 0.25, 0.50, 0.75, 0.90, 1\}$ are shown in Figure 9.

For these functions the task is maximization and regardless of α there is a unique maximum $br_n^\alpha(n,n) = 2n$. At $\alpha = 0$ the surface is just a plane. For $\alpha \in (0,1)$ two ridges appear and they get closer and closer as α increases. At $\alpha = 1$ the two ridges merge into one.

The formulas for the best-response curves are:

$bestResponseX(y) = \alpha y + (1-\alpha)n$

$bestResponseY(x) = \alpha x + (1-\alpha)n.$

These formulas describe two lines that intersect in (n,n) (where the optimum is!). Figure 10 plots them for the same values of α as above. At $\alpha = 0$ the two best-response lines are perpendicular; as α increases, the angle between them decreases, till finally they overlap when $\alpha = 1$. The value of the function along any such line decreases from $2n$ in (n,n) as we move towards the left (i.e. towards smaller x and y values).

Our hypothesis is as follows:

- at low α, increasing population size and introducing elitism will have beneficial effects on performance; very few steps are necessary to get very close to the optimum, so high accuracy in following the best-response curves is good;
- as α increases, we will begin to see "curving" effects, as the benefits of accuracy in following the best-response curves are counterbalanced by a number of generations too small to have time to reach proximity of the optimum; the law of diminishing returns sets in;
- as α reaches 1, increasing population size and introducing elitism will decrease performance, since closely following the best-response is bad when they are extremely close or overlapping.

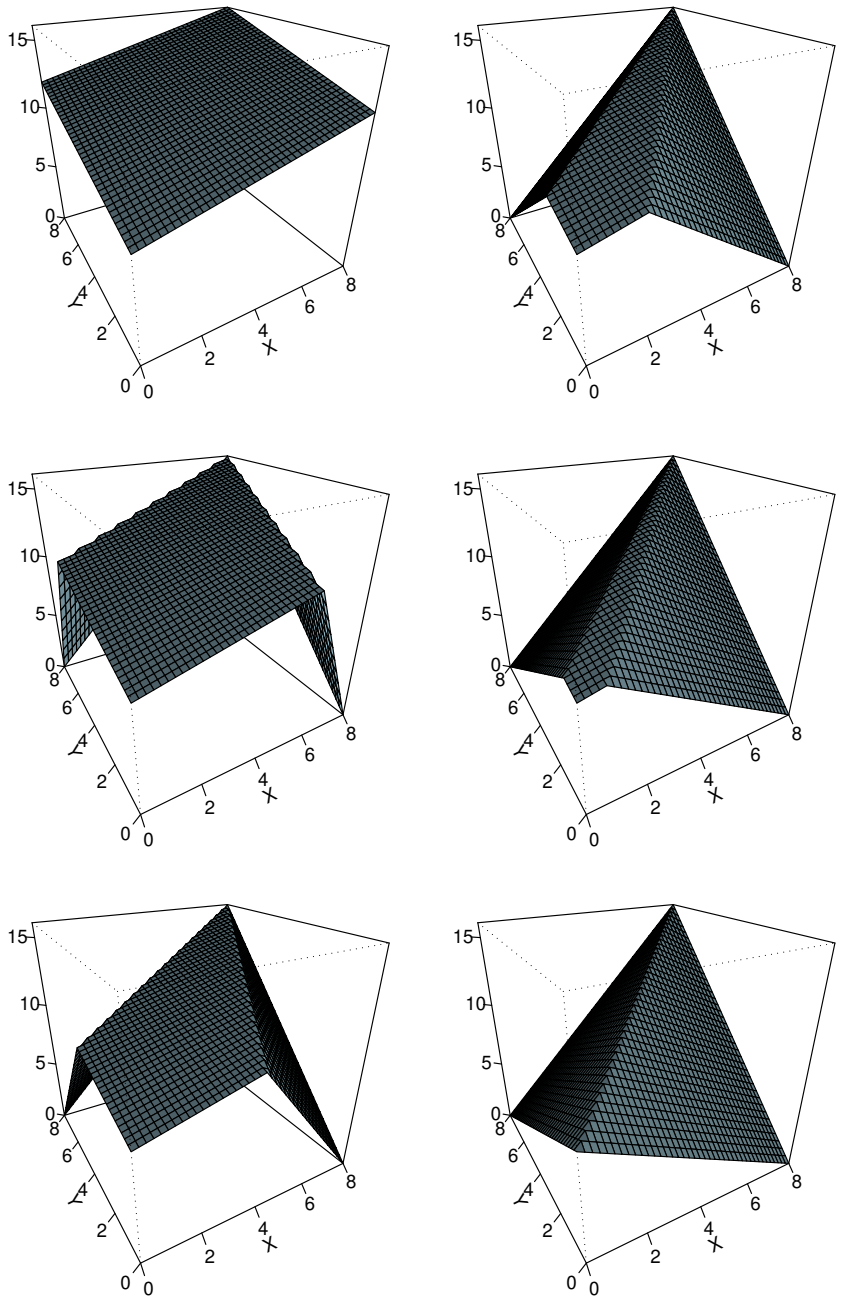

Figure 9. br_g^{α}. Left, top to bottom :$\alpha = 0, 0.25, 0.50$. Right, top to bottom :$\alpha = 0.75, 0.90, 1$

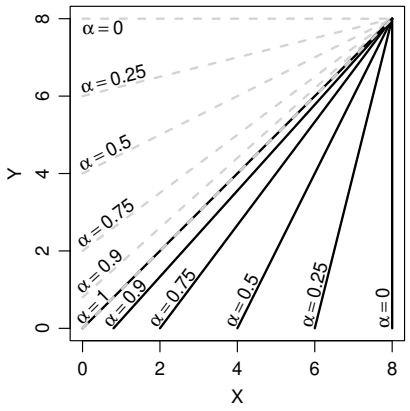

Figure 10. Best-response curves for br_8^{α}. Black continuous lines denote $bestResponseX$ and gray dashed lines denote $bestResponseY$.

For $n = 8$ and the six values of α mentioned above we ran the same population size and elitism experiments as before[5]. As can be seen in figures 11 and 12, our expectations were confirmed.

This argues for the fact that the best-response curves are important properties of co-evolutionary domains and their interaction is a key factor in determining which algorithm settings are better for performance. We do not expect to have exhausted all possible interactions between best-response curves for cooperative payoffs. For example, we expect that performance will not vary with population size and elitism if along a region of overlap the function value is constant. We may also see cases where there are several disjoint points (or regions) of intersection for best-response curves. We expect these to be related to the presence of local optima and performance is likely to also be affected by the shapes and sizes of their basins of attraction.

An example of this latter case can be observed on a family of functions introduced in (Panait et al., 2004):

$$MTQ(x,y) = max \left\{ \begin{array}{l} H_1(1 - 16\frac{(x-X_1)^2+(y-Y_1)^2}{S_1}) \\ H_2(1 - 16\frac{(x-X_2)^2+(y-Y_2)^2}{S_2}) \end{array} \right. .$$

The function is basically the maximum of two quadratic hills, centered at (X_1, Y_1) and (X_2, Y_2) with respective heights H_1 and H_2. S_1 and S_2 affect the area that the two hills cover: higher values for one of them result in a wider coverage of the specific hill. Intuitively, this should make it more probable that the coevolutionary search algorithm will converge to the peak of this hill, even though it may be suboptimal.

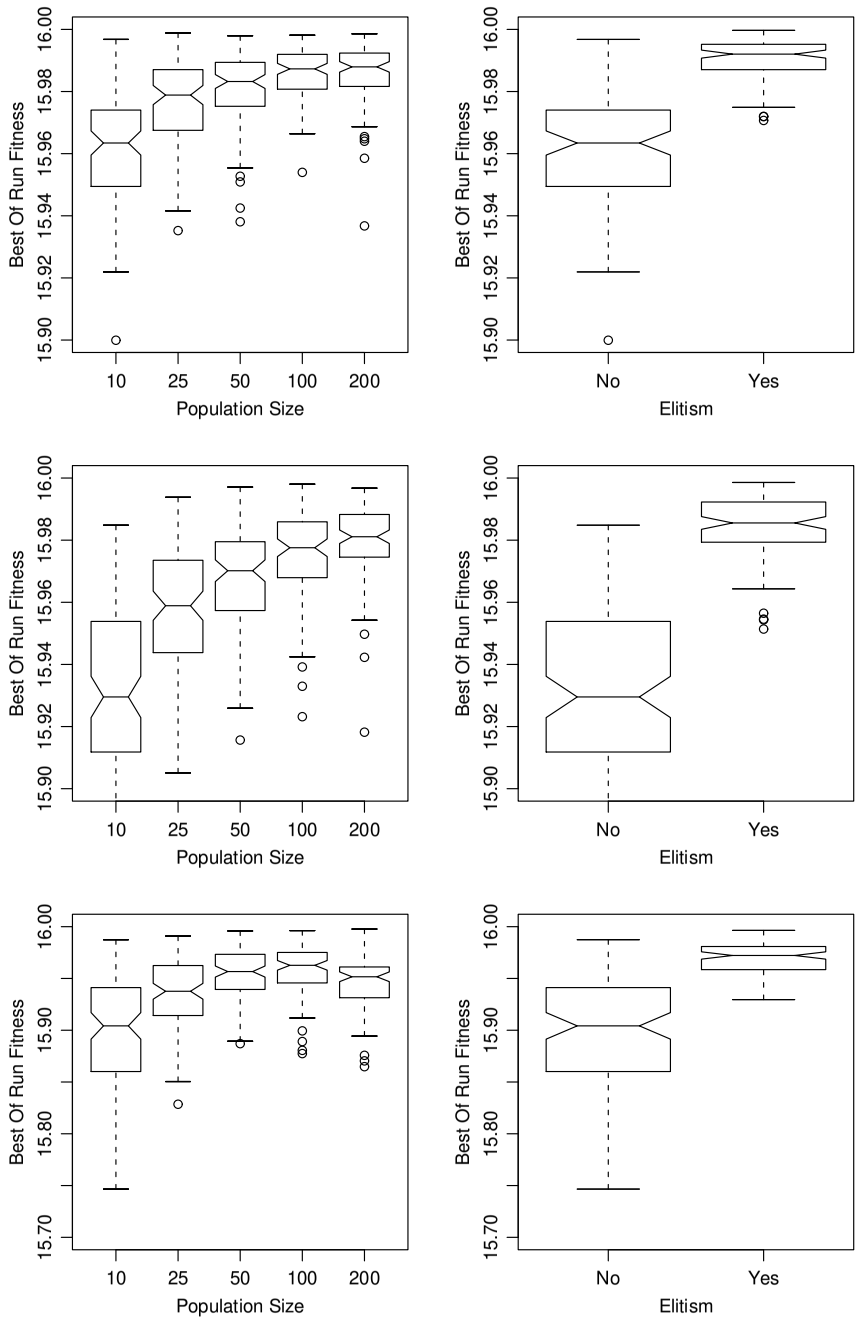

Figure 11. Best-of-run performance for br_8^{α}. Left: population size effects. Right: elitism effects. Top row: $\alpha = 0$; middle row: $\alpha = 0.25$; bottom row: $\alpha = 0.5$

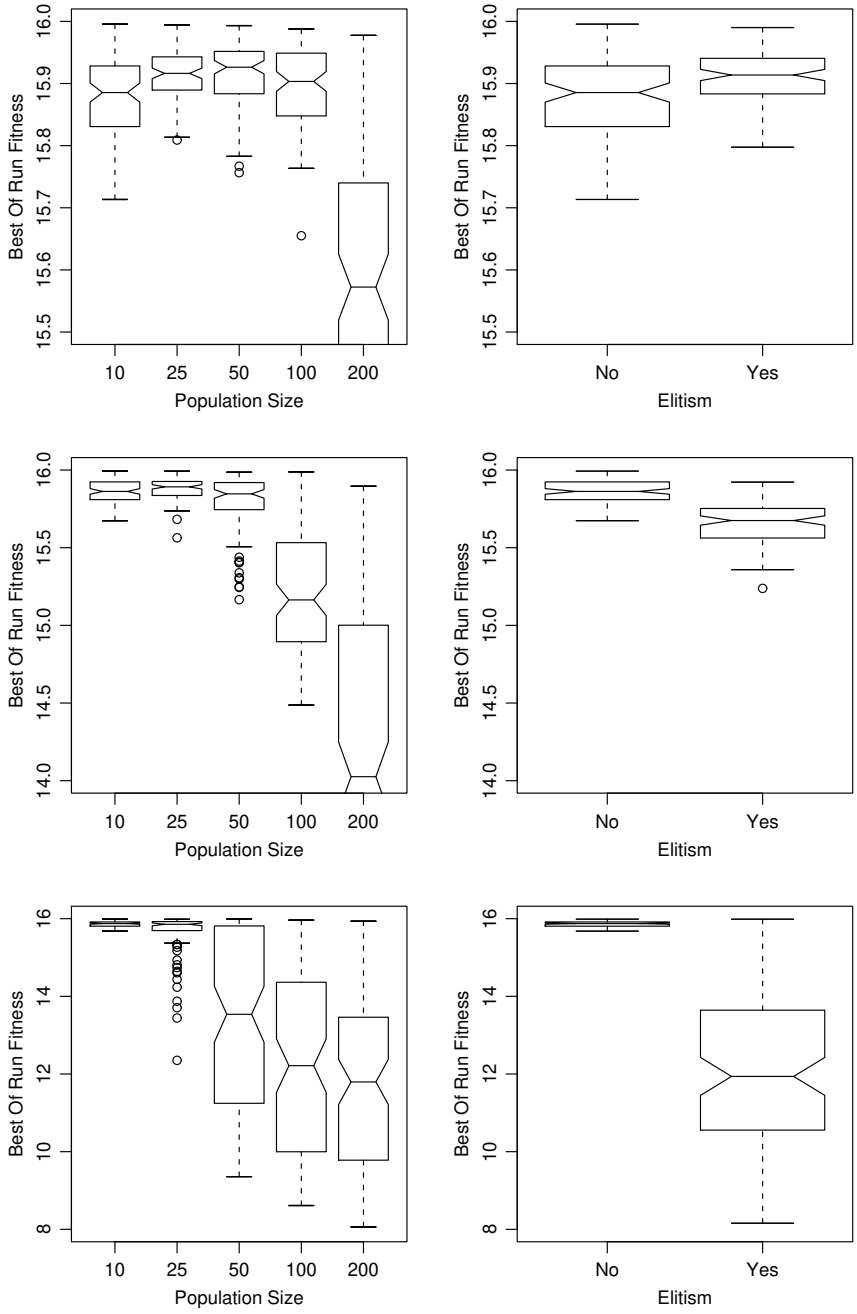

Figure 12. Best-of-run performance for br_8^{α}. Left: population size effects. Right: elitism effects. Top row: $\alpha = 0.75$; middle row: $\alpha = 0.90$; bottom row: $\alpha = 1$

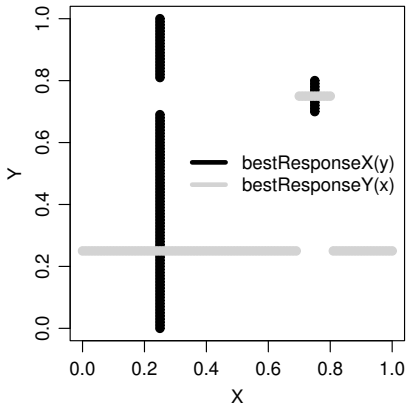

Figure 13. Best-response curves for MTQ with settings $X_1 = 1/4$, $Y_1 = 1/4$, $S_1 = 1.6$, $H_1 = 50$, $X_2 = 3/4$, $Y_2 = 3/4$, $S_2 = 1/32$, $H_2 = 150$.

The best-response curves help us understand better why that is. For parameter settings $X_1 = 1/4$, $Y_1 = 1/4$, $S_1 = 1.6$, $H_1 = 50$, $X_2 = 3/4$, $Y_2 = 3/4$, $S_2 = 1/32$, $H_2 = 150$ they look like in Figure 13. It is obvious from this image that a CoEA with a single-best collaboration strategy will in just a couple of steps be drawn to one of the two intersection points (one of which is a local optima and the other a global optima). Which point is picked highly depends on the random y individual used for evaluation in the first X generation (see end note 1), or a random x individual, if we were to start with a Y generation. In both cases, there are obviously fewer chances of reaching the top right point than reaching the bottom left one.

We can easily construct a modification of the function for which hill coverage doesn't tell us everything and the best-response curves uncover subtle effects we may not understand otherwise. What we do is that we add separate controls for the x-spread and the y-spread of the two hills:

$$asymMTQ(x,y) = max \begin{cases} H_1(1 - 16\frac{(x-X_1)^2}{S_1^x} - 16\frac{(y-Y_1)^2}{S_1^y}) \\ H_2(1 - 16\frac{(x-X_2)^2}{S_2^x} - 16\frac{(y-Y_2)^2}{S_2^y}) \end{cases}.$$

For parameter settings $X_1 = 1/4$, $Y_1 = 1/4$, $S_1^x = 1.6$, $S_1^y = 1/32$, $H_1 = 50$, $X_2 = 3/4$, $Y_2 = 3/4$, $S_2^x = 1/32$, $S_2^x = 1.6$, $H_2 = 150$ the best-response curves now look like those in Figure 14. They show that the peak more likely to be reached now depends on whether we start with an X generation or a Y generation and not on which hill covers more space!

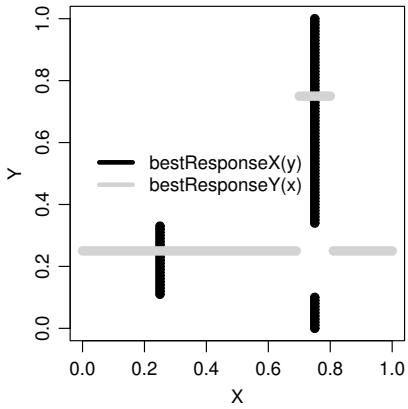

Figure 14. Best-response curves for $asymMTQ$ with settings $X_1 = 1/4$, $Y_1 = 1/4$, $S_1^x = 1.6$, $S_1^y = 1/32$, $H_1 = 50$, $X_2 = 3/4$, $Y_2 = 3/4$, $S_2^x = 1/32$, $S_2^x = 1.6$, $H_2 = 150$.

This is a further illustration of the usefulness best-response curves for understanding and improving co-evolutionary performance.

6. Conclusions

We believe that the key to efficiently using co-evolution as a problem solving tool is understanding its dynamical behavior. In this paper we present a new methodology of analyzing the dynamics of CoEAs based on trajectories of best-of-generation individuals. This methodology helped us identify some "hidden" problem properties, namely the best-response curves, that have a strong influence on co-evolutionary performance. Understanding the relationship between the best-response curves of a domain helps a CoEC designer tune various parameters of the algorithm in order to improve performance on that domain.

For most practical applications, formulas for the best-response curves will not be available. However, as it was shown in the paper, analyzing the trajectories of best-of-generation individuals will help infer their shapes and/or the relationships between them. The information gained in this way can further be used to make performance-improving changes to the algorithm.

While the work presented here looked only at population size and elitism, in (Popovici and De Jong, 2005a) we have used the same technique to show that best-response curves are also helpful in understanding the effects the collaboration methods have on performance. Additionally, although this paper has focused on cooperative setups, our previous work (Popovici and De Jong, 2004; Popovici and De Jong, 2005b) showed that the methods described here can be used to gain insights into competitive co-evolution as well.

In summary, we believe that this technique provides the CoEC practitioner with a valuable tool for understanding those aspects of co-evolutionary dynamics directly related to CoEC performance.

Acknowledgements

We would like to thank:

– Gabriel Balan, for suggesting to us the "best-response" terminology as appropriate for our work;

– An anonymous reviewer of our paper (Popovici and De Jong, 2005a), for suggesting parameterizing the angle between the best-response curves;

– Jayshree Sarma, for proof reading the manuscript and providing useful comments that we incorporated for improving paper.

Notes

[1] Co-evolutionary computation (CoEC) is the subfield of computer science that is concerned with the study of co-evolutionary algorithms (CoEAs) and their application to problem solving. We believe it is important to make this terminological distinction: use CoEAs to refer to the algorithms and CoEC to refer to the field, which encompasses both the algorithms and the problems they're applied to.

[2] The design decisions for our experiments were based on two heuristics: keep things simple and use reasonable settings. The simplest method of evaluating an individual in one population is to couple it with the current best member of the other population and the value of the function at that point is assigned as fitness. This has been termed *single best collaboration strategy* by (Wiegand, 2004) and is the equivalent of what was termed in competitive co-evolution LEO (last elite opponent) evaluation (Sims, 1994). It is also a case of symmetric payoff as presented by (Wiegand, 2004).

The two populations take turns in evolving, which means that during each generation only one population of the two is active (this has been termed *sequential update timing* by (Wiegand, 2004)). The pseudo-code of the algorithm for one X generation is given below for clarity (the one for Y can easily be inferred):

- evaluate the X population using the current y_{best};
- select parents according to determined fitness values;
- breed;
- evaluate the new population using the same y_{best};
- determine x_{best} according to these new fitnesses.

At the begining of each run, both populations are initialized uniformly random across the domain. Co-evolution starts by evaluating the members of the initial X generation in conjunction with a random y individual. For the first Y generation (second generation of the run) the x_{best} used is the actual best x individual from the first (X) generation.

In each population we use as starting point a non-overlapping generational EA with a real-valued representation, binary tournament selection, and a gaussian mutation operator. To keep things simple, we use a fixed sigma, but we set its value depending on the size of each function's domain: 0.08 for *rosenbrock* and 2.6

for $offAxisQuadratic$, i.e. about 1/50 the size of the variables' range. To obtain effects similar to a $1/L$ bit-flip mutation for the binary representation, we set the probability of altering a gene to 90%.

We experimented with a total of 2000 evaluations for the whole system (which means 1000 evaluations per population) and the following *per-population* setups: population size 10 x 100 generations, population size 25 x 40 generations, population size 50 x 20 generations, population size 100 x 10 generations and population size 200 x 5 generations. This means a total number of generations for the system of 200, 80, 40, 20 and 10. We stopped at population size 200, as increasing it more would leave us with too few generations for any evolution to take place. Elitism was used in the context of a population size of 10.

[3] The boxplot format permits a concise, comparative visualization of the median (center line), the 95% confidence interval for the median (notch around the median line), the inter-quartile range (box), the outliers (circles) and the spread of the remaining data (dotted lines and whiskers).

[4] Which parts of the best-response curves are visited is in part influenced by which population we start with. Section 5 contains additional comments on this issue.

[5] The value for the standard deviation of the gaussian mutation was adjusted to the size of these domains, namely set to 0.16

References

Alligood, K., T. Sauer, and J. Yorke: 1996, *Chaos: An Introduction to Dynamical Systems*. Springer.

Angeline, P. J. and J. B. Pollack: 1994, 'Coevolving high-level representations'. In: C. G. Langton (ed.): *Artificial Life III*, Vol. XVII. Santa Fe, New Mexico, pp. 55–71, Addison-Wesley.

Bucci, A. and J. Pollack: 2002, 'Order-theoretic Analysis of Coevolution Problems: Coevolutionary Statics'. In: W. B. Langdon et al. (eds.): *Genetic and Evolutionary Computation Conference Workshop: Understanding Coevolution*. Morgan Kaufmann.

Cliff, D. and G. F. Miller: 1994, 'Co-Evolution of Pursuit and Evasion I: Biological And Game-Theoretic Foundations. Technical Report'. Technical report, School of Cognitive and Computing Sciences, University of Sussex.

Cliff, D. and G. F. Miller: 1995, 'Tracking the Red Queen: Measurements of Adaptive Progress in Co-Evolutionary Simulations'. In: *European Conference on Artificial Life*. pp. 200–218.

Ficici, S., O. Melnik, and J. Pollack: 2000, 'A Game-Theoretic Investigation of Selection Methods Used in Evolutionary Algorithms'. In: e. a. A. Zalzala (ed.): *Proc. of CEC2000*. IEEE Press.

Ficici, S. G. and J. B. Pollack: 1998, 'Challenges in Coevolutionary Learning: Arms-Race Dynamics, Open-Endedness, and Mediocre Stable States'. In: C. Adami, R. K. Belew, H. Kitano, and C. Taylor (eds.): *Artificial Life VI: Proc. of the Sixth Int. Conf. on Artificial Life*. Cambridge, MA, pp. 238–247, The MIT Press.

Hillis, W. D.: 1990, 'Co-evolving parasites improve simulated evolution as an optimization procedure'. In: *CNLS '89: Proceedings of the ninth annual international conference of the Center for Nonlinear Studies on Self-organizing, Collective, and Cooperative Phenomena in Natural and Artificial Computing Networks on Emergent computation*. pp. 228–234, North-Holland Publishing Co.

Hofbauer, J. and K. Sigmund: 1998, *Evolutionary Games and Population Dynamics*. Cambridge University Press.

Luke, S. and R. P. Wiegand: 2003, 'When Coevolutionary Algorithms Exhibit Evolutionary Dynamics'. In: *Workshop Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003)*. Springer.

Pagie, L. and M. Mitchell: 2002, 'A comparison of evolutionary and coevolutionary search'. *International Journal of Computational Intelligence and Applications* **2**(1), 53–69.

Panait, L., R. P. Wiegand, and S. Luke: 2004, 'A Sensitivity Analisys of a Cooperative Coevolutionary Algorithm Biased for Optimization'. In: *Proceedings of the Genetic and Evolutionary Computation Conference – GECCO-2004*.

Popovici, E. and K. De Jong: 2004, 'Understanding Competitive Co-evolutionary Dynamics via Fitness Landscapes'. In: S. Luke (ed.): *AAAI Fall Symposium on Artificial Multiagent Learning*. AAAI Press.

Popovici, E. and K. De Jong: 2005a, 'A Dynamical Systems Analysis of Collaboration Methods in Cooperative Co-evolution'. In: *AAAI Fall Symposium Series Co-evolution Workshop*. to appear.

Popovici, E. and K. De Jong: 2005b, 'Relationships between Internal and External Metrics in Co-evolution'. In: *Congress on Evolutionary Computation*. to appear.

Popovici, E. and K. De Jong: 2005c, 'Understanding Cooperative Co-evolutionary Dynamics via Simple Fitness Landscapes'. In: H.-G. Beyer et al. (eds.): *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2005*. New York, ACM Press.

Potter, M. and K. De Jong: 1994, 'A Cooperative Coevolutionary Approach to Function Optimization'. In: *Proceedings of the Third Conference on Parallel Problem Solving from Nature (PPSN III)*. Jerusalem, Israel, pp. 249–257, Springer.

Rosin, C. D. and R. K. Belew: 1995, 'Methods for Competitive Co-Evolution: Finding Opponents Worth Beating'. In: *Proceedings of the 6th International Conference on Genetic Algorithms*. pp. 373–381, Morgan Kaufmann Publishers Inc.

Sims, K.: 1994, 'Evolving 3D Morphology and Behaviour by Competition'. In: R. Brooks and P. Maes (eds.): *Artificial Life IV Proceedings*. MIT, Cambridge, MA, USA, pp. 28–39, MIT Press.

Wiegand, R. P.: 2004, 'An Analysis of Cooperative Coevolutionary Algorithms'. Ph.D. thesis, George Mason University, Fairfax, VA.