A Filter-Based Evolutionary Algorithm for Constrained Optimization

Lauren Clevenger Lmcleve@aol.com
University of New Mexico

Lauren Ferguson alaferg@yahoo.com
Texas Technical Institute

William E. Hart wehart@sandia.gov
Sandia National Laboratories, Discrete Algorithms and Math Dept., P. O. Box 5800, MS 1110, Albuquerque, NM 87185-1110

Abstract

We introduce a filter-based evolutionary algorithm (FEA) for constrained optimization. The filter used by an FEA explicitly imposes the concept of dominance on a partially ordered solution set. We show that the algorithm is provably robust for both linear and nonlinear problems and constraints. FEAs use a finite pattern of mutation offsets, and our analysis is closely related to recent convergence results for pattern search methods. We discuss how properties of this pattern impact the ability of an FEA to converge to a constrained local optimum.

Keywords

Constrained optimization, evolutionary pattern search, convergence theory, multiobjective evolutionary algorithm

1 Introduction

Although evolutionary algorithms (EAs) have been successfully applied to many unconstrained optimization applications, the investigation of constrained EAs has received far less attention (Coello, 2002). Despite this, handling constraints in EAs is necessary for their application to many problem domains. Thus the development of provably robust EAs is crucial to ensure that these methods can be effectively applied to a wide range of problems, including linear, non-linear, equality and inequality constraints.

Of the many different constraint-handling techniques used with EAs, the most common are penalty functions. Although penalty functions can have good convergence properties for specific problems, they have drawbacks that limit their use in practice. Some penalty functions require an initial feasible solution that must be provided by the user or by another algorithm. Penalty approaches may also require extra parameters that can be hard to choose, especially when they are problem-dependent. Part of the difficulty in implementing penalty functions is the automation of their definitions, because the boundary between the feasible and infeasible regions is usually unknown (Coello, 2002).

Alternatives to penalty functions tend to be developed for very specific problems and problems in which estimating good penalty functions and generating even a single

feasible solution are difficult. Some of the techniques surveyed by Coello (2002) include approaches that use problem-specific representations and operators, algorithms that repair infeasible points to make them feasible, and approaches that separate objectives and constraints (e.g. multi-objective optimization techniques). Unfortunately, these methods sometimes have difficulty preserving diversity and avoiding stagnation. Additionally, some of these approaches require the generation of an initial feasible point (or population), which may itself be NP-hard (Smith and Coit, 1997).

Considering all of these challenges for handling constraints in EAs, an approach that minimizes these difficulties and still maintains good convergence results is very desirable. We propose filter-based evolutionary algorithms (FEAs), which use a constraint-handling technique that is similar to multi-objective EAs. These are Pareto-based EA's, but we will call them FEA's to allow for our changes. The optimization problem that we consider is

$$\begin{aligned} \min_{x \in \mathbf{R}^n} \quad & f(x) \\ \text{s.t.} \quad & C(x) \leq 0 \\ & l \leq x \leq u \end{aligned}$$

where $f : \mathbf{R}^n \to \mathbf{R} \cup \{\infty\}$ and $C : \mathbf{R}^n \to (\mathbf{R} \cup \{\infty\})^m$ are the constraint functions with $C = (C_1, \ldots, C_m)^T$; $u, l \in \mathbf{Q}^n$ define upper and lower bounds on each dimension.

Our design of FEAs is inspired by a recent analysis of pattern search methods for this class of problems (Audet and Dennis, 2004). Like EAs, pattern search methods are direct search methods that do not attempt to estimate a derivative in their search process. The main qualitative difference between pattern search methods and common real-coded EAs is that pattern search methods restrict the search in each iteration to a finite pattern of trial points, while most real-coded EAs employ continuous random variables to generate mutation steps. However, this restriction provides mathematical leverage for demonstrating convergence of pattern search methods, which has been effectively translated to EAs (Hart, 2005).

The following section describes FEAs and motivates their use for constrained optimization problems. We define a simple FEA and discuss how this EA is related to other EAs for constrained optimization. Section 3 describes general conditions for which convergence is guaranteed, focusing on *multipoint pattern search sequences*. Section 4 demonstrates that FEAs generate a multipoint pattern search sequence with probability one, so all of the convergence results described in Section 3 are applicable to FEAs. Finally, we consider three examples that illustrate how the choice of search pattern can impact the convergence of FEAs in Section 5.

2 Algorithmic Formulation

A filter-based optimizer uses a nonnegative continuous function to aggregate the constraint violations and then treats the resulting biobjective problem (Audet and Dennis, 2004). In other words, a filter-based optimizer tries to minimize both the objective function and the aggregate constraint violation function simultaneously. Since a feasible solution is desired, priority is usually given to the aggregate function until a feasible solution is found. We give two definitions that are very similar to those stated in Audet and Dennis (2004), which will be used throughout this paper.

Definition 1. *Given objective functions $f_i(), i = 0, \ldots, k$, if $f_i(x_1) \leq f_i(x_2)$ for every $i \in 1, \ldots, k$ and there is at least one j such that $f_j(x_1) < f_j(x_2)$, then x_1 is said to* dominate *x_2. This is denoted by $x_1 \prec x_2$. Also, $x_1 \preceq x_2$ denotes that either $x_1 \prec x_2$ or $x_1 = x_2$.*

Definition 2. *Given a set of points S, a point $s \in S$ is said to be a* nondominated point *if there does not exist $x \in S$ such that $x \prec s$.*

A *filter* F is a (finite) set of points in $\mathbf{R}^n$ such that no pair (x, x') in the filter are in the relation $x \prec x'$. That is, no point in F dominates or is dominated by any other point in F. Filter-based optimizers employ a filter that is used to eliminate trial points from consideration if they are dominated by points in the filter either by having a worse function value or worse aggregate constraint violation.

2.1 A Filter-Based EA

Figure 1 presents the basic steps of Algorithm A, a filter-based evolutionary algorithm that implicitly applies the notion of a filter-based optimizer to an EA. This EA evolves a set of points $W_t = Y_t \bigcup X_t$, where Y_t are infeasible and X_t are feasible. We say $x \prec x'$ if and only if $(f(x), h(x)) \prec (f(x'), h(x'))$, where $f(x)$ is the objective function and $h(x) = \sum_{i=1}^{m} \max[0, C_i(x)]^2$. Note that $h(x) = \infty$ if any of the constraint function values at x are infinite.

Given Δ_0, $\tau > 1$ ($\tau \in \mathbf{Q}$) and mutation directions D
Randomly initialize X_0 and Y_0; $W_0 = X_0 \bigcup Y_0$
Select $D_0 \subseteq D$
For $t = 0, \ldots, \infty$
  For $j = 1, \ldots, P$
    Randomly select $d \in D_t$ and $w \in W_t$
    $\hat{w}_j = \Delta_t d + w$
    Evaluate $\hat{w}_j$
  End For
  Update X_{t+1}, Y_{t+1}; $W_{t+1} = X_{t+1} \bigcup Y_{t+1}$
  Update x^*_{t+1} and y^*_{t+1}
  If $(f(x^*_{t+1}) < f(x^*_t))$ or
    $(h(y^*_{t+1}) < h(y^*_t))$ or
    $((h(y^*_{t+1}) = h(y^*_t))$ and $(f(y^*_{t+1}) < f(y^*_t)))$ Then
    $\Delta_{t+1} = \Delta_t \tau^\nu$, where $0 \leq \nu \leq \nu_{max}$
    Select $D_{t+1} \subseteq D$
  Else If x^*_{t+1} or y^*_{t+1} is locally optimal Then
    $\Delta_{t+1} = \Delta_t \tau^\nu$, where $\nu_{\min} \leq \nu < 0$
    Select $D_{t+1} \subseteq D$
  Else
    $\Delta_{t+1} = \Delta_t$ and $D_{t+1} = D_t$
  Terminate if $\Delta_{t+1} < \Delta_{min}$
End For

Figure 1: Pseudo-code for Algorithm A. For simplicity, we have not included the checks to see if either x^*_{t+1} or y^*_{t+1} are not defined because a feasible or infeasible point has not been encountered by iteration $t+1$. These checks would be used in all of the conditional statements after x^*_{t+1} and y^*_{t+1} are updated.

This FEA implicitly uses a filter, which is the subset of W_t containing the best non-dominated infeasible solutions Y_t and the best feasible solution x^*_t. Let x^*_t be the point in X_t with the best function value, and y^*_t be the point in Y_t with the minimal constraint violation (as defined by h). If two points have the same minimal constraint violation, y^*_t

is the point with the minimal function value. The following general assumption defines the necessary criteria for the updates to X_{t+1} and Y_{t+1}:

Assumption 1. *Given a set of new trial points $\hat{W}_t$, let $\hat{X}_t$ and $\hat{Y}_t$ be the subsets of feasible and infeasible points in $\hat{W}_t$ respectively. X_{t+1} and Y_{t+1} satisfy the following criteria:*

- *$X_{t+1} \subseteq X_t \bigcup \hat{X}_t$ such that it contains the best feasible point in this set,*
- *$Y_{t+1} \subseteq Y_t \bigcup \hat{Y}_t$ such that it contains the nondominated point in this set with the minimal aggregate constraint violation for which the objective function f is minimal, and*
- *$|X_{t+1}| \leq \mu_F$ and $|Y_{t+1}| \leq \mu_I$.*

Note that Assumption 1 ensures that $|W_{t+1}| < \mu_F + \mu_I$, for some predefined parameters μ_F and μ_I. Additionally, this assumption simply requires that if the best feasible or least-infeasible points are improved, those improvements are retained in the next iteration. Consequently, this FEA could be implemented with a method similar to a $(\mu + \lambda)$-evolutionary strategy (ES): P new trial points are generated in each iteration, and at most $\mu_F + \mu_I$ of the best points are kept for the next iteration.

Intuitively, a point is locally optimal if it cannot be improved. In the context of an FEA, a point cannot be improved if the mutation steps about it are dominated by the points in the filter contained in W_t. The following definition formally defines this notion of local optimality:

Definition 3. *Let $\mathcal{N}(x, \Delta_t, D_t) = \{x + \Delta_t d \mid d \in D_t\}$. Given $x \in \mathbf{R}^n$, we say that x is* locally optimal *if all of the points $\mathcal{N}(x, \Delta_t, D_t)$ have been generated and there does not exist $d \in D_t$ for which $x + \Delta_t d \prec x$.*

Note that Assumption 1 ensures that this notion of local optimality is well-defined. Suppose that all points in $\mathcal{N}(x, \Delta_t, D_t)$ have been generated. If either x_t^* or y_t^* is in this set, then x is locally optimal only if $(f(x), h(x))$ equals $(f(x_t^*), h(x_t^*))$ or $(f(y_t^*), h(y_t^*))$, so generating this mutation step does not give a simple improvement in either x_t^* or y_t^*. Suppose that neither x_t^* nor y_t^* is in $\mathcal{N}(x, \Delta_t, D_t)$. Then all of the points in $\mathcal{N}(x, \Delta_t, D_t)$ are dominated by either x_t^* or y_t^*. Further, in subsequent iterations these values will only improve so for all $t' > t$, the points in $\mathcal{N}(x, \Delta_t, D_t)$ are dominated by either $x_{t'}^*$ or $y_{t'}^*$.

The following provides further details about the definition of Algorithm A:

- X_1 and Y_1 could be simply initialized by randomly generating P points within the bound constraints, and then applying the standard update rule. However, in practice this initialization could exploit domain knowledge of the structure of the constraints.
- $D \in \mathbf{Q}^{n \times q}$ is a finite set of mutation offsets that can be applied. All subsets $D_t \subseteq D$ must be selected to ensure that D_t is a positive spanning set (i.e. non-negative linear combinations of points in D_t generate $\mathbf{R}^n$).
- The determination of whether x_{t+1}^* or y_{t+1}^* is locally optimal is not made with respect to the current population W_t. Instead, this requires the explicit cataloging of the history of mutation steps about these points.
- Algorithm A updates the step length Δ_t by (a) possibly increasing it if some new point dominates either x_t^* or y_t^*, or (b) decreasing it if x_{t+1}^* or y_{t+1}^* are locally optimal (and thus no progress can be made about these points using D_t).

- Algorithm A terminates if the step length shrinks below some predetermined threshold, which is the termination rule commonly used with pattern search methods.

2.2 Numerical Example

To illustrate the search behavior of Algorithm A, we consider the linear problem defined by Lewis and Torczon (2000)

$$\begin{aligned} \min \quad & -a - 2b \\ \text{s.t.} \quad & 0 \le a \le 1 \\ & b \le 0 \end{aligned}$$

where the optimal solution is $x^* = (a^*, b^*) = (1, 0)$. We will begin with two initial solutions $x_0 = (0, 0)$ (feasible) and $y_0 = (0, 1)$ (infeasible), an initial mesh size parameter $\Delta_0 = 1$, and an initial positive spanning set D_0 which consists of four directions: $\pm(1, 1)$ and $\pm(1, -1)$. In this example, $\mu_I = \mu_F = 1$, so the FEA only considers two points: the best feasible and best infeasible solutions. To simplify our presentation, we suppose that $P = 8$, so all mutation steps are generated in each iteration.

Figures 2 and 3 show the first four iterations of Algorithm A. The points x_t^* and y_t^* are denoted by black points. During the first iteration, the algorithm evaluates $f(x)$ and $h(x)$ at the points generated from x_0 and y_0, and two new feasible points are produced. One of these, the trial point $(1, 0)$, dominates the other new feasible solution and the feasible incumbent. Thus $x_1^* = (1, 0)$. Similarly, of the six new infeasible points generated, the trial point $(1, 1)$ is the only nondominated solution and so $y_1^* = (1, 1)$. Thus the new best infeasible point was generated from the feasible incumbent and the new best feasible point was generated from the infeasible incumbent, which is only possible because our algorithm allows searching around both feasible and infeasible solutions simultaneously.

In the second iteration, the algorithm evaluates both functions at the points generated from the new incumbent solutions. No better points are found and so the mesh size parameter Δ_t is decreased for the next iteration, i.e. $\Delta_3 = 0.5$. In the third iteration, a new best infeasible solution is found, but the best feasible point is unchanged. In fact, the best feasible point is now the optimal point, so subsequent iterations simply refine the search about the best feasible point.

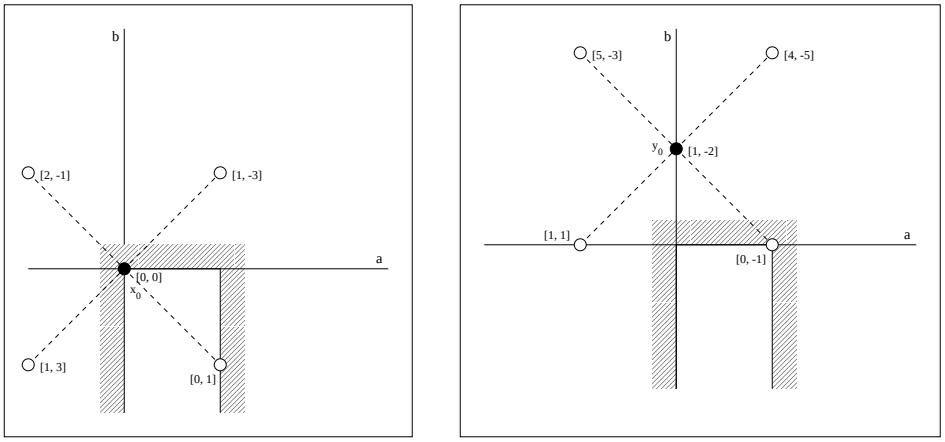
2.3 Related Constrained EAs

The method of constraint handling proposed here shares some commonalities with a few of the techniques surveyed by Coello (2002). Since Algorithm A separates constraints from objectives, it is most similar to approaches that also use this separation. Consider the Similarity of Feasible Points technique proposed by Deb (2001). Deb gives three rules for comparing points:

1. A feasible point is always preferred over an infeasible one.

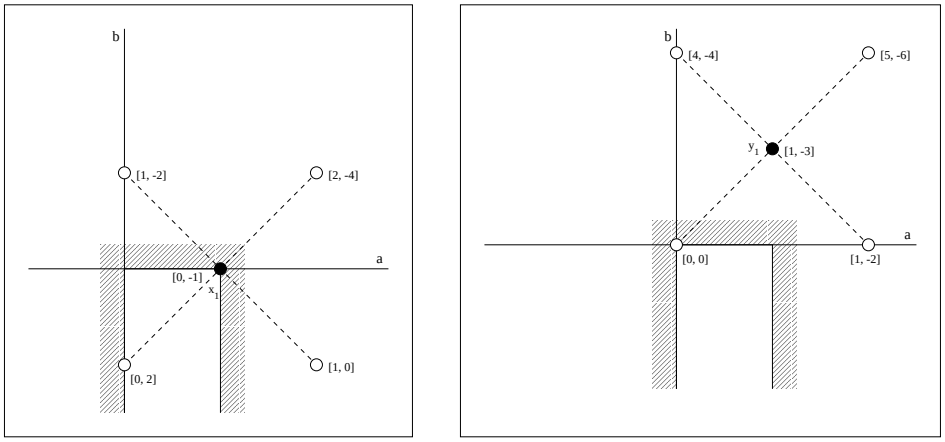
2. Between two feasible points, the one having a better objective function value is preferred.

3. Between two infeasible points, the one having a smaller constraint violation is preferred.



Iteration 1: Feasible **Iteration 1: Infeasible**

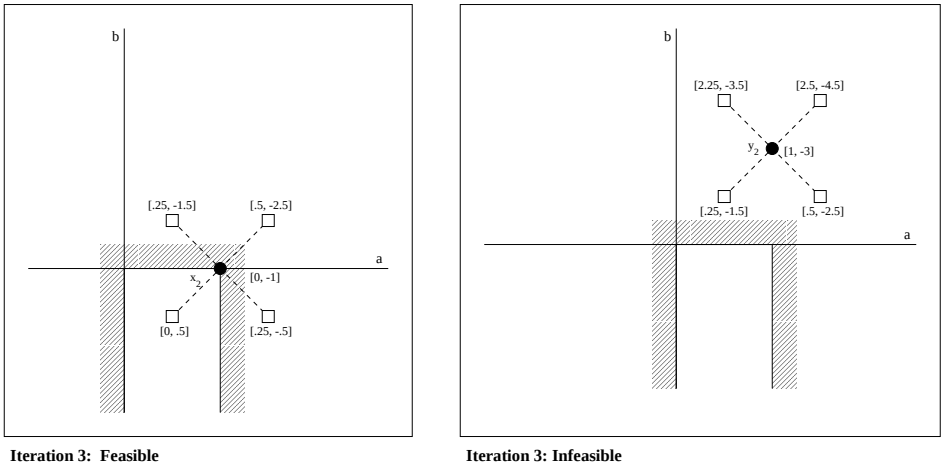
(a)



Iteration 2: Feasible **Iteration 2: Infeasible**

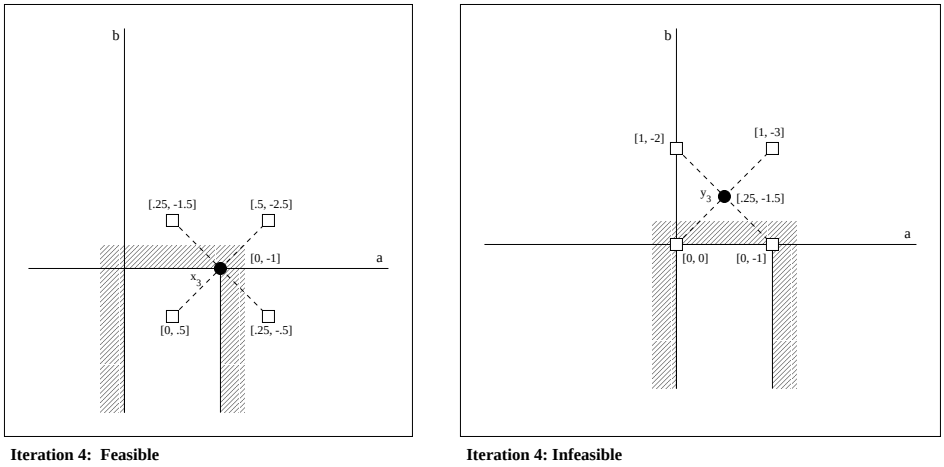
(b)

Figure 2: Illustration of two iterations of Algorithm A. The initial step length is Δ_0. The infeasible region is denoted by the gray hashmarks along the feasible region. The values $[h(x), f(x)]$ are provided for each point in the search.



Iteration 3: Feasible **Iteration 3: Infeasible**

(a)



Iteration 4: Feasible **Iteration 4: Infeasible**

(b)

Figure 3: Illustration of two iterations of Algorithm A that continue the search illustrated in Figure 2.

Deb's method also includes a selection procedure that only performs pairwise comparisons so that no penalty factor is required (Coello, 2002). Similarly, Algorithm A performs pairwise comparison for selection and follows rules 2 and 3 of Deb's method. It does not necessarily follow the first rule because we want to keep infeasible points (which are often needed to perform an effective search on constrained problems).

Algorithm A is also similar to some of the multiobjective optimization techniques surveyed by Coello (2000). The most closely related technique is the one proposed by Camponogara and Talukdar (1997). Their procedure restates a single optimization problem to consider two objectives: the optimization of the original objective function and the optimization of

$$\Phi(x) = \sum_{i=1}^{n} \max[0, C_i(x)].$$

Thus Φ is the analogue of h using the L_1 norm instead of the squared L_2 norm. Camponogara and Talukdar use pareto sets (implicitly using a filter) to impose dominance-based selection, which is used to estimate new search directions. The technique we propose implicitly uses a filter to impose dominance-based selection, but it is not used to generate new search directions. Instead, the filter is used to determine when step lengths are expanded and contracted (by imposing conditions for local optimality).

3 Multipoint Pattern Search Sequences

The main goal of our analysis is to show that there exist subsequences of $\{x_t^*, y_t^*\}$ that have interesting convergence properties. The sequence $\{x_t^*, y_t^*\}$ represents the sequence of the best feasible, x_t^*, and best least-infeasible, y_t^*, points generated by Algorithm A. In this section, we describe *multipoint pattern search sequences* and discuss their convergence properties. In Section 4 we demonstrate that Algorithm A generates a multipoint pattern search sequence with high probability, thereby demonstrating that this FEA is provably convergent.

We consider the convergence properties of a pair of points $v_t = \{x_t, y_t\}$ that have a corresponding step length sequence $\{\Delta_t\}$ and pattern sequence $\{D_t\}$. We consider $x_t, y_t \in \mathbf{R}^n \bigcup \{\varpi\}$, where the value ϖ represents an "undefined" point. The point x_t is either feasible or it is ϖ, and similarly the point y_t is either infeasible or it is ϖ. Let $f(x) < f(\varpi)$ and $h(x) < h(\varpi)$ for all $x \in \mathbf{R}^n$. We require that v_t only contains points that are either the same as the points in v_{t-1} or improvements of these points via steps taken from D_{t-1}. Thus, either $x_t = x_{t-1}$ or $x_t \prec x_{t-1}$, and similar conditions apply to y_t. We refer to v_t as the t-th *iterate* in the sequence.

Definition 4. *Consider a sequence $\{v_t\}$, where $v_t = \{x_t, y_t\}$ with $x_t, y_t \in \mathbf{R}^n \bigcup \{\varpi\}$. Let $\{\Delta_t\}$ be a corresponding step length sequence and $\{D_t\}$ a corresponding pattern sequence, where $D_t \subseteq D \in \mathbf{Q}^{n \times q}$ and D_t is a positive spanning set. We say that $\{v_t\}$ is a* multipoint pattern search sequence *when the following conditions hold:*

1. *$\{v_t\}$ is an infinitely long sequence for which all points $\{x_0, y_0, x_1, y_1, \ldots \mid x_t, y_t \in \mathbf{R}^n\}$ lie in a compact set,*
2. *$\Delta_t = \Delta_0 \tau^{r_t}$ for some $\tau = \tau_n/\tau_d$, $\tau_n, \tau_d \in \mathbf{Z}^+$ and for $r_t \in \mathbf{Z}$,*
3. *Let $\forall t \geq 0$, $\rho_t = \min_{k=0,\ldots,t} r_k$. There exist $r_{\max} > 0$ and $u_i \in \mathbf{R}^n$, $i = 1, \ldots, g$ (where g is independent of t), such that $x_t, y_t \in \{\varpi\} \bigcup \left(\bigcup_{j=1}^{g} \{u_i + \Delta_0 \tau_d^{-r_{\max}} \tau_n^{\rho_t} Dz \mid z \in \mathbf{Z}^q\} \right)$.*

4. *If* $v_{t+1} = v_t$, *then*
   - *Either* x_t *or* y_t *is locally optimal with respect to* Δ_t *and* D_t, *and*
   - $\Delta_{t+1} = \Delta_t \tau^{\nu}$, *where* $-\infty < \nu_{\min} \leq \nu < 0$
5. *If* $v_{t+1} \neq v_t$, *then*
   - *Either* $x_{t+1} \prec x_t$ *or* $y_{t+1} \prec y_t$ *(or both), and*
   - $\Delta_{t+1} = \Delta_t \tau^{\nu}$, *where* $0 \leq \nu \leq \nu_{max} < \infty$

If $v_{t+1} = v_t$, then a multipoint pattern search requires that all neighboring points have been generated for either x_{t+1} or y_{t+1}, and that all of these points are not improving. Consequently, the step length is reduced. Similarly, if $v_{t+1} \neq v_t$, then either x_{t+1} or y_{t+1} is an improving point, and the step length may be expanded. Note that the condition that the points $\{x_0, y_0, \ldots \mid x_t, y_t \in \mathbf{R}^n\}$ lie in a compact set is an assumption that is commonly made in the analysis of optimization algorithms.

The following theorem demonstrates a key feature of a multipoint pattern search sequence: a subsequence of the step lengths converges to zero.

Theorem 1. *If* $\{v_t\}$ *is a multipoint pattern search sequence, then* $\liminf_{t\to\infty} \Delta_t = 0$.

Proof. We assume toward a contradiction that there exists $\Delta_{\min} > 0$ such that $\Delta_t \geq \Delta_{\min}$ for all t. Now $\Delta_t = \Delta_0 \tau^{r_t}$, so there is a greatest lower bound of $\{r_t\}$, $\rho_{\min} > -\infty$. We know from the definition of a multipoint pattern search sequence that for all t,

$$x_t, y_t \in \{\varpi\} \bigcup \left(\bigcup_{j=1}^{g} \{u_i + \Delta_0 \tau_d^{-r_{\max}} \tau_n^{\rho_{\min}} Dz \mid z \in \mathbf{Z}^q\} \right).$$

Thus the points x_t and y_t lie on a union of meshes defined by D. From the definition of a multipoint pattern search sequence, we know that the points in $\{v_t\}$ lie within a compact set. It follows that there are finitely many values of x_t and y_t, so there are finitely many possible iterates v_t. Since we have $x_{t+1} \preceq x_t$ and $y_{t+1} \preceq y_t$, it follows that there are finitely many iterations for which $v_t \neq \hat{v}$. Consequently, there must be infinitely many iterations for which v_t equals v_{t+1}. The step length Δ_t is contracted in these iterations, so it follows that Δ_t is reduced infinitely often. This contradicts our assumption. □

Let $z_k = \{x_0, y_0, x_1, y_1, \ldots \mid x_t, y_t \in \mathbf{R}^n\}$ be the sequence of real-valued points taken from $\{v_t\}$, and let $\{\Delta_k\}$ denote the associated lengths for each of the points in $\{z_k\}$. The following definition describes a *refining subsequence* of $\{z_k\}$, about which we will demonstrate convergence results.

Definition 5. *A subsequence of a multipoint pattern search sequence* $\{z_k\}_{k\in K}$ *(for some subset of indices K) is said to be a* refining subsequence *if* $\{z_k\}_{k\in K}$ *is convergent and* $\lim_{k\in K} \Delta_k = 0$.

We say that a point z' generated from z_k is *filtered* if $\exists k' \in K$, $k' \leq k$ such that $z_{k'} \prec z'$ (so z' does not improve on either the best feasible or least infeasible point). A search direction $d \in \mathbf{R}^n$ is said to be *associated* with the refining subsequence $\{z_k\}_{k\in K}$ if for every term of the subsequence, the point $z_k + \Delta_k d$ has been evaluated and filtered. A positive spanning set consisting of directions associated with a given refining subsequence is said to be an *associated* positive spanning set.

The following lemma demonstrates the existence of refining subsequences.

Lemma 1. *There exists a refining subsequence of a multipoint pattern search sequence $\{z_k\}$ with associated positive spanning set $\hat{D}$.*

Proof. Note that Δ_t only decreases if either x_t or y_t is locally optimal. Consider the subsequence K that defines such locally optimal points in $\{z_k\}$. Let $\{D_k\}_{k\in K}$ be the associated sequence of positive spanning sets for which these points are locally optimal.

Since Δ_k is only reduced for iterates in K, we know from Theorem 1 that there exists a subsequence $K' \subseteq K$ for which $\lim_{k\in K'} \Delta_k = 0$. Further, the sequence $\{z_k\}_{k\in K'}$ lies in a compact set, so it has a convergent subsequence $K'' \subseteq K'$ (by the Bolzano-Weierstrass Theorem (Ross, 1980)).

Finally, the set D is finite, so there are finitely many positive spanning sets that can be composed from directions in D. Thus there must be some positive spanning set $\hat{D}$ that occurs infinitely often in $\{D_k\}_{k\in K''}$. Let $K''' \subseteq K''$ be the subsequence for which $\hat{D}$ is applied. □

The following two theorems and their immediate corollaries give derivative results for f and h. These result consider the properties of a directional derivative at a limit point $\hat{z}$ of a refining subsequence. Clarke (1990) defines the generalized directional derivative of h in the direction s to be

$$h^0(\hat{z}; s) = \limsup_{y\to\hat{z},\ t\downarrow 0} \frac{h(y+ts) - h(y)}{t}.$$

Theorem 2 shows that the generalized directional derivative of the constraint violation function, h, is nonnegative when h is Lipschitz near the limit point.

Theorem 2. *Let $\hat{z}$ be the limit point of a refining subsequence $\{z_k\}_{k\in K}$ with associated direction s. Assume h is Lipschitz near $\hat{z}$. Then the generalized directional derivative of h at $\hat{z}$ in the direction s is nonnegative, i.e., $h^0(\hat{z}; s) \geq 0$.*

Proof. If the limit point $\hat{z}$ is feasible, then $h(\hat{z}) = 0$ and nonnegativity of h implies that $h^0(\hat{z}; s) \geq 0$ for any associated direction s.

Consider the case where $\hat{z}$ is infeasible, and let $K' \subseteq K$ be iterates for which $h(\hat{z} + \Delta_k s) \neq 0$. Since h is Lipschitz near $\hat{z}$, Clarke (1990) shows that we have:

$$h^0(\hat{z}; s) = \limsup_{y\to\hat{z},\ t\downarrow 0} \frac{h(y+ts) - h(y)}{t} \ \geq\ \limsup_{k\in K'} \frac{h(z_k + \Delta_k s) - h(z_k)}{\Delta_k}.$$

Since s is an associated direction of a refining sequence, we know that the point $z_k+\Delta_k s$ has been generated and filtered. It follows that $h(z_k + \Delta_k s) \geq h(z_k)$, so the quotient $\frac{h(z_k+\Delta_k s)-h(z_k)}{\Delta_k}$ is nonnegative. Thus we have $h^0(\hat{z}; s) \geq 0$. □

The following corollary generalizes Theorem 2 to the case where the constraint violation function h is strictly differentiable at the limit point $\hat{z}$. Note that this result would also apply if h was continuously differentiable, since that is a stronger condition than strict differentiability (Clarke, 1990).

Corollary 1. *Let $\hat{z}$ be the limit point of a refining subsequence $\{z_k\}_{k\in K}$ with an associated positive spanning set $\hat{D} \subseteq D$. If h is strictly differentiable at $\hat{z}$ then $\nabla h(\hat{z}) = 0$.*

Proof. If h is strictly differentiable at $\hat{z}$, then $h^0(\hat{z}; w) = \nabla h(\hat{z})^T w$ for any direction $w \neq 0$ (Clarke, 1990). By Theorem 2, for all $d_i \in \hat{D},\ 0 \leq h^0(\hat{z}; d_i) = \nabla h(\hat{z})^T d_i$. Since $\hat{D}$ is a

positive spanning set, we can write w as a nonnegative linear combination of d_i and λ_i: $w = \sum \lambda_i d_i,\ \lambda_i \geq 0$. Thus we have

$$\nabla h(\hat{z})^T w = \nabla h(\hat{z})^T \left(\sum \lambda_i d_i \right) = \sum \lambda_i (\nabla h(\hat{z})^T d_i) \geq 0.$$

Similarly, we can write $-w$ as $-w = \sum \lambda_i' d_i,\ \lambda_i' \geq 0$. So we have

$$\nabla h(\hat{z})^T (-w) = \nabla h(\hat{z})^T \left(\sum \lambda_i' d_i \right) = \sum \lambda_i' (\nabla h(\hat{z})^T d_i) \geq 0.$$

Consequently, $\nabla h(\hat{z}) = 0$. □

Theorem 3 provides an analogous analysis of the directional derivative for the objective function, f. However, this result imposes a condition on h that is only likely to be satisfied if $\hat{z}$ is a feasible point.

Theorem 3. *Let $\hat{z}$ be the limit point of a refining subsequence $\{z_k\}_{k \in K}$ with associated direction s. Assume f is Lipschitz near $\hat{z}$. If $h(z_k) = h(z_k + \Delta_k s)$ for each k in K, then $f^0(\hat{z}; s) \geq 0$.*

Proof. By way of contradiction, assume that $f^0(\hat{z}; s) < 0$. Since f is Lipschitz near $\hat{z}$, we again have from Clarke (1990):

$$f^0(\hat{z}; s) = \limsup_{y \to \hat{z}, t \downarrow 0} \frac{f(y + ts) - f(y)}{t} \ \geq \ \limsup_{k \in K} \frac{f(z_k + \Delta_k s) - f(z_k)}{\Delta_k}.$$

Thus the right hand side of this inequality is bounded above by $f^0(\hat{z}; s)$, which is negative. It follows that $f(z_k + \Delta_k s) < f(z_k)$ for sufficiently large k in K. But since $h(z_k) = h(z_k + \Delta_k s)$, it follows that $z_k + \Delta_k s \prec z_k$. But this forms a contradiction; $z_k + \Delta_k s$ is filtered because s is an associated direction of the refining sequence $\{z_k\}_{k \in K}$, which implies that $z_k + \Delta_k s \not\prec z_k$. Thus we must have $f^0(\hat{z}; s) \geq 0$. □

The following corollary to Theorem 3 demonstrates that multipoint pattern search sequences have the desirable optimality properties for strictly feasible limit points.

Corollary 2. *Let $\hat{z}$ be the limit point of a refining subsequence $\{z_k\}_{k \in K}$ and s an associated direction. If $\hat{z}$ is strictly feasible, then $f^0(\hat{z}; s) \geq 0$.*

Proof. If $\hat{z}$ is strictly feasible, then there exists $\epsilon > 0$ such that $h(x) = 0$ for every x satisfying $\|x - \hat{z}\| \leq \epsilon$. If $k \in K$ is large enough, then both $h(z_k) = 0$ and $h(z_k + \Delta_k s) = 0$. In other words, there is a radius of length ϵ' around $\hat{z}$ for which this is true. Since $h(z_k) = h(z_k + \Delta_k s)$, Theorem 3 shows that $f^0(\hat{z}; s) \geq 0$. □

Unfortunately, Theorem 3 does not ensure that when $\hat{z}$ is on the boundary of the feasible region, $-\nabla f(\hat{z})$ is in the normal cone to the feasible region at $\hat{z}$.[1] However, the following corollary provides a cone that contains both the normal cone to the feasible region at $\hat{z}$ as well as $-\nabla f(\hat{z})$.

Corollary 3. *Let C_s be the cone generated by all the associated directions of all refining subsequences that converge to the limit point $\hat{z}$ and that satisfy the conditions of Theorem 3. If f is strictly differentiable at $\hat{z}$, then $-\nabla f(\hat{z})$ belongs to the polar C_s^0 of C_s.*

Proof. Theorem 3 guarantees that $f^0(\hat{z}; s) \geq 0$ for any vector s that generates C_s. Since f is strictly differentiable at $\hat{z}, \nabla f(\hat{z})^T s \geq 0$ for all s. From the definition of C_s, any vector y from $\hat{z}$ pointing into C_s is a nonnegative linear combination of these associated directions, i.e. $\nabla f(\hat{z})^T y \geq 0$. It follows that $-\nabla f(\hat{z})^T y \leq 0$, so $-\nabla f(\hat{z})$ belongs to the polar C_s^0 of C_s where $C_s^0 = \{x \in \Re^n : x^T y \leq 0,\ \forall y \in C_s\}$. □

[1] The normal cone of the feasible region at $\hat{z}$ is the polar of the tangent cone to the feasible region at $\hat{z}$.

4 Analysis of Filtered EAs

In this section we provide a convergence theory for Algorithm A. The main goal of the analysis in this section is to demonstrate that with probability one, some subsequence of the points $\{x_t^*, y_t^*\}$ generated by the Algorithm A provably converges. Let X_t and Y_t be the stochastic processes, defined on some probability space $(\Omega, \mathcal{F}, P)$, that describe the behavior of how Algorithm A generates the sequence $\{x_t^*, y_t^*\}$ for some problem and for some set of algorithmic parameters (e.g. μ, λ, etc.). We make the standard assumption that the processes X_t and Y_t generate points that lie in a compact set.

For all $\omega \in \Omega$, consider the sequence $v_t(\omega) = (x_t^*(\omega), y_t^*(\omega))$, where $x_t^*(\omega)$ equals ϖ if no feasible point has been generated by Algorithm A (and similarly for y_t^*). Further, consider the index set $T(\omega)$ such that $t \in T(\omega)$ if

- either $x_{t+1}^*(\omega) \prec x_t^*(\omega)$ or $y_{t+1}^*(\omega) \prec y_t^*(\omega)$, or
- either $x_t^*(\omega)$ or $y_t^*(\omega)$ is locally optimal.

The following lemmas will be used to demonstrate that a sequence $\{v_t(\omega)\}_{t \in T(\omega)}$ is a multipoint pattern search sequence. The proof of Lemma 2 follows directly from the definition of Algorithm A.

Lemma 2. *Let $\omega \in \Omega$. Let $\{\Delta_t(\omega)\}$ be the sequence of step lengths generated by Algorithm A. Then $\Delta_t = \Delta_0 \tau^{r_t}$ for some $r_t(\omega) \in \mathbf{Z}$.*

The following lemma demonstrates that points generated by Algorithm A lie on a set of lattices.

Lemma 3. *Let $\omega \in \Omega$. We define $\forall t \geq 0$, $\rho_t(\omega) = \min_{k=0,\ldots,t} r_k(\omega)$. There exist $r_{\max} > 0$ such that $x_t^*(\omega), y_t^*(\omega) \in \{\varpi\} \bigcup \left(\bigcup_{j=1}^{g} \{w_j(\omega) + \Delta_0 \tau_d^{-r_{\max}} \tau_n^{\rho_t(\omega)} Dz \mid z \in \mathbf{Z}^q\} \right)$, where $\{w_0(\omega), \ldots, w_g(\omega)\} = W_0(\omega)$.*

Proof. Recall that $\{x_t^*(\omega)\}$ and $\{y_t^*(\omega)\}$ lie within a compact set. Thus there exists a diameter R such that for any two points within the set, the distance between the two points is less than or equal to R. Now if the step length $\Delta_t(\omega)$ is greater than R, all of the points generated about $x_t^*(\omega)$ and $y_t^*(\omega)$ lie outside this compact set, so these points must necessarily be locally optimal. Consequently, $\Delta_t(\omega)$ is bounded above by a constant independent of t, so there exists $r_{\max}$ such that $r_t(\omega) \leq r_{\max}$ for all t.

If $x_t^*(\omega) \neq \varpi$, then we can write $x_t^*(\omega)$ as

$$x_t^*(\omega) = w(\omega) + \sum_{i=0}^{t-1} \Delta_i(\omega) D z_i(\omega),$$

where $z_i(\omega) \in \mathbf{Z}^q$ and $w(\omega) \in W_0(\omega)$. Now $\tau = \tau_n / \tau_d$, for $\tau_n, \tau_d \in \mathbf{Z}^+$. It follows that

$$x_t^*(\omega) = w(\omega) + \sum_{i=0}^{t-1} \Delta_0 \tau^{r_i(\omega)} D z_i(\omega) = w(\omega) + \Delta_0 \tau_d^{-r_{\max}} \tau_n^{\rho_t(\omega)} \sum_{i=0}^{t-1} D z_i'(\omega),$$

for some $z_i'(\omega) \in \mathbf{Z}^q$. Note that $\sum_{i=1}^{t} D z_i'(\omega) = D z''(\omega)$ for some $z''(\omega) \in \mathbf{Z}^q$. Further, this argument applies equally well to $y_t^*(\omega)$. Thus we have

$$x_t^*(\omega), y_t^*(\omega) \in \{\varpi\} \bigcup \left(\bigcup_{j=1}^{g} \{w_i(\omega) + \Delta_0 \tau_d^{-r_{\max}} \tau_n^{\rho_t(\omega)} Dz \mid z \in \mathbf{Z}^q\} \right),$$

where $\{w_0(\omega), \ldots, w_g(\omega)\} = W_0(\omega)$. □

The following theorem is the main result of this section. This result demonstrates that with probability one, Algorithm A generates a multipoint pattern search sequence. Consequently, the results of Section 3 apply to Algorithm A with probability one.

Theorem 4. *Let* $A = \{\omega \in \Omega \mid \{v_t\} \text{ is a multipoint pattern search sequence}\}$. *Then* $P(A) = 1$.

Proof. Our goal is to show that the five conditions in Definition 4 are satisfied for sequences $v_t(\omega)$. Consider $\omega \in \Omega$. Lemmas 2 and 3 demonstrate that conditions 2 and 3 are satisfied. Conditions 4 and 5 are naturally enforced in Algorithm A through the update rule for $x_t^*(\omega)$ and $y_t^*(\omega)$, and from the definition of $T(\omega)$. Thus we need to show that condition 1 is satisfied with probability one. Again, we make the standard assumption that the sequences $\{v_t(\omega)\}$ lie in a compact set, for all ω.

If $\{v_t(\omega)\}_{t \in T(\omega)}$ is finite, then there exists $K_0(\omega)$ such that for all $t > K_0(\omega)$, $v_t(\omega) = v_{t+1}(\omega)$ and $\Delta_t(\omega) = \Delta_{t+1}(\omega)$. We also know that for these iterations, $v_t(\omega)$ is not locally optimal, so for all $t > K_0(\omega)$, $D_t(\omega) = D_{t+1}(\omega)$. Further, there is at least one direction $d(\omega) \in D_t(\omega)$ that is never sampled for all $t > K_0(\omega)$. It follows that $d(\omega)$ is sampled finitely many times.

Now, let $B \subset \Omega$ be events such that for all $\omega \in B$, $\{v_t(\omega)\}$ is finite. Consider $\omega \in B$, and let $d(\omega)$ be the corresponding direction in $\lim_{t\to\infty} D_t(\omega)$ that is sampled finitely many times. Note that $B = \bigcup_{k=1}^{\infty} B_k$, where B_k is the set of events where $v_t(\omega) = v_{t+1}(\omega)$ and $\Delta_t(\omega) = \Delta_{t+1}(\omega)$ for all $t \geq k$. From the definition of Algorithm A, we know that the probability of sampling each $d \in D$ is at least $1/|D|$. Thus since $d(\omega)$ is only sampled finitely many times, it follows that $P(B_k) = 0$. Further, since $B_k \subseteq B_{k+1}$, we have $P(B) = \lim_{k\to\infty} P(B_k) = 0$. Note that $A = B^c$, so $\{v_t(\omega)\}_{t\in T(\omega)}$ is infinitely long with probability one. Note that we have assumed that X_t and Y_t generate points in a compact set, so condition 1 is satisfied. □

5 Convergence Examples

Theorem 4 demonstrates that Algorithm A generates a multipoint pattern search sequence with probability one. Thus there are interesting limit points generated by this method. However, we have noted that our convergence analysis does not guarantee that if a limit point $\hat{z}$ is on the boundary of the feasible region then $-\nabla f(\hat{z})$ is in the normal cone to the feasible region at $\hat{z}$. Although Algorithm A generates convergent limit points, it may generate limit points on the boundary of the feasible region that are not constrained stationary points (i.e. KKT points (Gill et al., 1981)).

The following examples illustrate the implications of Theorem 3 and Corollary 3 on test problems. Specifically, these examples illustrate how the choice of search directions impacts the ability of Algorithm A to converge to constrained stationary points. In each of these examples, we consider the case where Algorithm A is used with a single search pattern throughout the search, which is consistent with the manner in which most pattern search methods are employed. We discuss this point further in the next section.

5.1 Example I

Consider the problem

$$\begin{array}{ll} \min & -ab \\ \text{s.t.} & a^2 + b^2 \leq 16 \\ & 0 \leq a, b \leq 4 \end{array}$$

for which the optimal solution is $x^* = (a^*, b^*) = (2\sqrt{2}, 2\sqrt{2})$. Consider the behavior of Algorithm A using the search pattern $D_t = D = \{\pm(1,1), \pm(1,-1)\}$. As in our earlier example, we simplify our presentation by assuming that $\mu_F = \mu_I = 1$ and $P = 8$, so all mutation steps are generated in each iteration. We consider feasible starting points, so in initial iterations P is effectively equal to 4.

Figure 4a illustrates the convergence behavior of Algorithm A when started from a set of points along the x- and y-axes. The lines in this figure connect the initial and final points, and it is clear that in every case the FEA converges to the optimal solution. This can be explained using Corollary 3. Note that $-\nabla f(x) = (b, a)$. Now suppose that Algorithm A generates a limit point $\hat{z} = (a, b)$ on the constraint boundary for which $b > a$. It follows that the directions $(-1, -1)$ and $(1, -1)$ are the associated directions that satisfy Theorem 3 at this limit point (because the constraint violation function is constant in these directions), and they define the cone C_s. The polar cone C_s^o is defined by the directions $(-1, 1)$ and $(1, 1)$. However, at $\hat{z}$ we have $-\nabla f(\hat{z}) = (b, a)$ which is not in this cone if $b > a$ (note that $(-1, 1)(b, a)^T = a - b < 0$). Consequently, the only limit point that Algorithm A could generate that satisfies the conditions of Theorem 3 and Corollary 3 is the point (a^*, b^*).

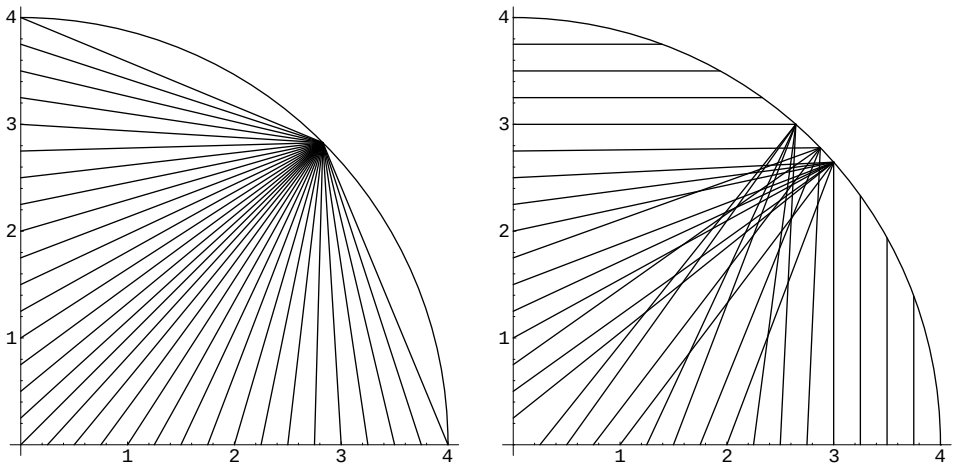


Figure 4: Illustration of the convergence of Algorithm A for Example I when (a) $D_t = D = \{\pm(1,1), \pm(1,-1)\}$ and (b) $D_t = D = \{\pm(1,0), \pm(0,1)\}$. The lines in these figures connect the initial and final points. The lack of symmetry in (b) is due to the fact that ties are broken arbitrarily.

Now consider the behavior of Algorithm A using the search pattern $D_t = D = \{\pm(1,0), \pm(0,1)\}$. Figure 4b illustrates the convergence behavior of Algorithm A with this pattern when started from a set of points along the x- and y-axes. It is clear that this FEA does not consistently converge to the optimal solution. Again, this can be explained using Corollary 3. Suppose that Algorithm A generates a limit point $\hat{z} = (a, b)$ on the constraint boundary for which $b > a$. It follows that the directions $(-1, 0)$ and $(0, -1)$ are the associated directions that satisfy Theorem 3 at this limit point, and they define the cone C_s. The polar cone C_s^o is defined by the directions $(1, 0)$ and $(0, 1)$, which includes the direction $-\nabla f(\hat{z}) = (b, a)$ when $b > a$. Consequently, this limit

point is allowed within our convergence theory for Algorithm A.

The contrast between these two search patterns highlights the degree to which the choice of search pattern can impact how closely Algorithm A converges to constrained stationary points. If the search pattern is selected well, you may be able to ensure that a constrained stationary point is generated, but if the search pattern is selected poorly then any point on the nonlinear constraint boundary may be a limit point. Furthermore, it is clear that if the search pattern $D_t = D = \{\pm(1,1), \pm(1,-1)\}$ were perturbed slightly then this FEA could converge to points other than the constrained stationary point. Consequently, this method may be sensitive to numerical instabilities such as round-off errors.

5.2 Example II

Consider the problem

$$\begin{array}{ll} \min & f(a,b) \\ \text{s.t.} & \frac{1}{5}a + \frac{4}{5} \leq b \\ & 5a - 4 \geq b \\ & 0 \leq a, b \end{array}$$

where $f : \mathbf{R}^2 \rightarrow \mathbf{R}$ is an arbitrary function. The solution to this problem lies within the feasible region, but we consider the convergence of Algorithm A starting from an initial point (λ, λ) for some $\lambda > 1$. The following analysis shows that Algorithm A converges to the point $(1,1)$ on the constraint boundary, regardless of whether this is a constrained stationary point. In fact, all iterates remain infeasible on this problem, so f could even be minimized at a strictly feasible point. Again, we assume that $\mu_F = \mu_I = 1$, and that all mutation steps are generated in each iteration (so we are taking the best of all neighboring points).

Figure 5 illustrates the initial point and the three search directions in the search pattern used in this example. From a point (a,b) the solution set steps during the search are 120 degrees apart from one another, given by

$$\begin{aligned} \left(a + \Delta\cos\left(\frac{\pi}{4}\right), b + \Delta\sin\left(\frac{\pi}{4}\right)\right) &= \left(a + \Delta\sqrt{2}/2, b + \Delta\sqrt{2}/2\right) \\ \left(a + \Delta\cos\left(\frac{-5\pi}{12}\right), b + \Delta\sin\left(\frac{-5\pi}{12}\right)\right) &= \left(a + \Delta\sqrt{2}\,\omega_2, b - \Delta\sqrt{2}\,\omega_1\right) \\ \left(a + \Delta\cos\left(\frac{11\pi}{12}\right), b + \Delta\sin\left(\frac{11\pi}{12}\right)\right) &= \left(a - \Delta\sqrt{2}\,\omega_1, b + \Delta\sqrt{2}\,\omega_2\right) \end{aligned}$$

where $\omega_1 = (\sqrt{3}+1)/4$ and $\omega_2 = (\sqrt{3}-1)/4$. We label these points $\overline{a}$, $\overline{b}$ and $\overline{c}$ respectively, and we label the initial point $\overline{x}$.

We denote constraint (1) to be $b \geq \frac{1}{5}a + \frac{4}{5}$ and constraint (2) to be $b \leq 5a - 4$. Let $D_1^{\overline{x}}$ be the shortest squared distance from $\overline{x}$ to constraint (1), and let $D_2^{\overline{x}}$ be the shortest squared distance from $\overline{x}$ to constraint (2). We define similar values for $\overline{a}$, $\overline{b}$, and $\overline{c}$. To compute these values, we need to be able to compute the shortest squared distance from a point to the constraints that point is violating. The following lemma defines the point on a line that is closest to a given point.

Lemma 4. *The shortest squared distance between a point (r,s) and a line $y = mx + b$ is at $x = \frac{r+(s-b)m}{m^2+1}$.*

The following corollary follows directly from Lemma 4.

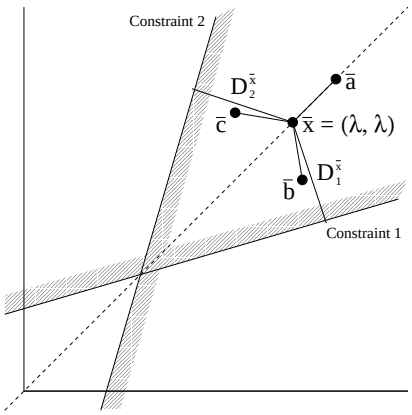


Figure 5: An illustration of the initial point and associated search directions in Example II.

Corollary 4. *The squared distance from* (r, s) *to constraint (1) is* $\frac{(4+r-5s)^2}{26}$*, and the distance to constraint (2) is* $\frac{(4-5r+s)^2}{26}$.

The following theorem demonstrates that Algorithm A converges to the feasible point $(1, 1)$, though all iterates remain infeasible.

Theorem 1. *If* $z_0 = (\lambda, \lambda)$ *for some* $\lambda > 1$*, then no feasible point is generated and* $\lim_{t\to\infty} z_t = (1, 1)$.

Figure 6 illustrates the four different *algorithmic states* that can occur when Algorithm A searches from a point (λ, λ). The four lemmas in Appendix A demonstrate how the search progresses as follows:

(a) The point z_t is at some point (λ', λ') and the points $\overline{b}$ and $\overline{c}$ are infeasible for both constraints. Lemma 5 shows that either (1) the step length is reduced or (2) either $z_{t+1} = \overline{b}$ or $z_{t+1} = \overline{c}$.

(b) The point z_t is at some point (λ', λ') and the points $\overline{b}$ and $\overline{c}$ are each feasible for a single constraint. Lemma 6 shows that the step length is reduced.

(c) The previous algorithmic state was state (a), and the point $\overline{b}$ is infeasible for both constraints. Lemma 7 shows that $z_{t+1} = (\lambda', \lambda')$, where $\lambda' > 1$.

(d) The previous algorithmic state was state (a), and the point $\overline{b}$ is feasible for a single constraint. Lemma 8 shows that $z_{t+1} = (\lambda', \lambda')$, where $\lambda' > 1$.

We now prove Theorem 1 using this decomposition of the search of Algorithm A.

Proof of Theorem 1. Beginning at some point $z_0 = (\lambda, \lambda)$, $\lambda > 1$, it is clear from the results of Lemmas 5, 6, 7 and 8 that Algorithm A generates a sequence of points such that if $x_t \neq y_t$ then $x_{t+1} = y_{t+1}$. At most half of the iterations do not lie on the line $x = y$, and a simple inspection of these iterations (cases (c) and (d) in Figure 6) confirms that for all iterations t such that $x_t = y_t$, $x_t > 1$.

Thus there is an infinite subsequence K s.t. $x_k = y_k, \forall k \in K$. Let $\{(\lambda_k, \lambda_k)\}_{k\in K}$ denote this subsequence. We know that $\lambda_k > 1$ for all $k \in K$, so the sequence $\{\lambda_k\}_{k\in K}$ is monotonic and bounded below. Thus there exists a limit point of $\{(\lambda_k, \lambda_k)\}_{k\in K}$.

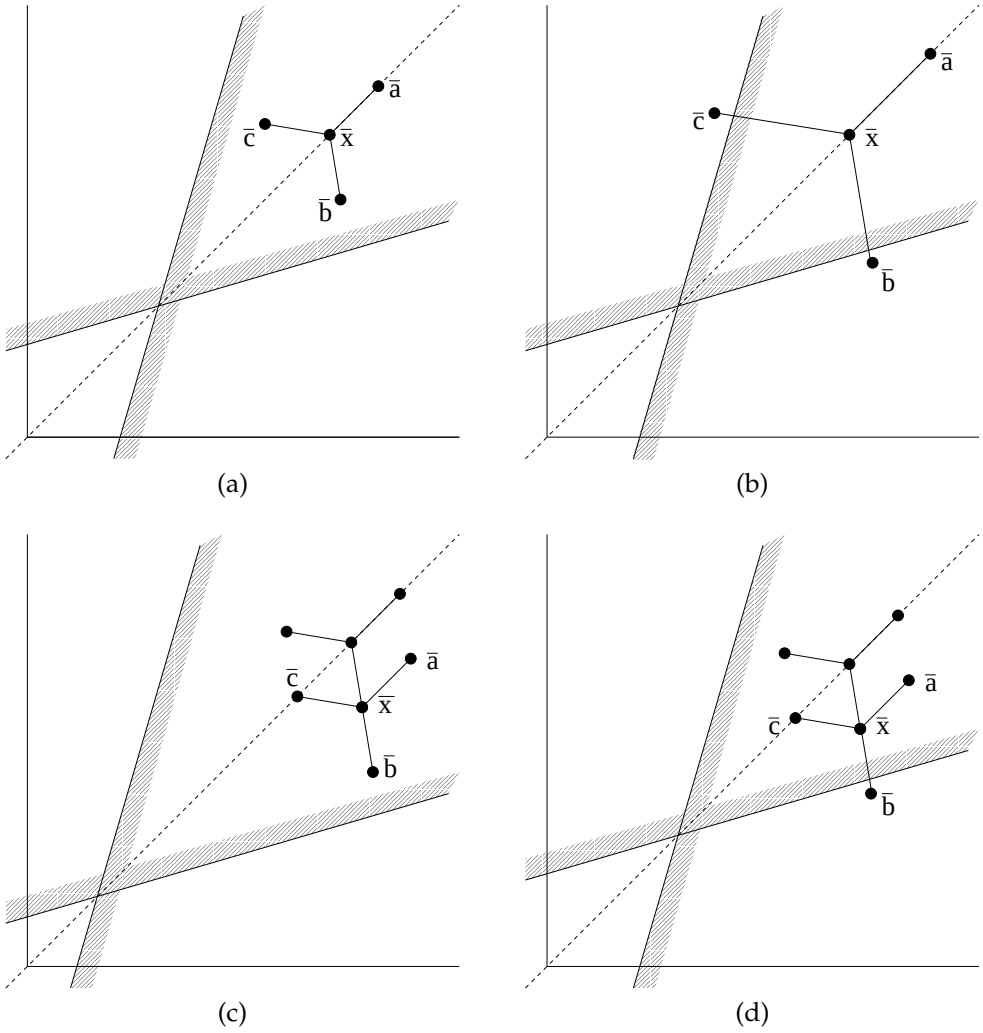


Figure 6: The four different algorithmic states that can occur in Example II.

Suppose that $\exists\ \lambda^* > 1$ s.t. (λ^*, λ^*) is this limit point. The sequence $(\lambda_k, \lambda_k)_{k \in K}$ is a refining subsequence because $\lim_{k \in K} \Delta_k = 0$, with associated positive spanning set defined by the fixed pattern used in this example. Now h is continuously differentiable, so h is strictly differentiable at $z^* = (\lambda^*, \lambda^*)$. Thus we know from Corollary 1 that $\nabla h(z^*) = 0$. But this is a contradiction, because ∇h is only zero at the point $(1, 1)$. Thus we conclude that $z^* = (1, 1)$. □

This analysis demonstrates a less obvious limitation of our convergence theory than is illustrated in Example I. Specifically, this example shows that although the convergence theory may ensure convergence to a feasible point, the local properties of that point may be poorly characterized. Although the specific choice of search pattern was crucial to our analysis, we conjecture that this is a more fundamental weakness of multi-objective EA search strategies for constrained optimization. By treating the objective function as one of two or more objectives, the search may proceed to find feasible solutions without regard to whether these solutions are interesting. Multi-objective EA strategies must also search in the neighborhood of the limit point to ensure that it is a constrained stationary point.

5.3 Example III

As we have seen in our previous examples, the choice of the search pattern can fundamentally impact the convergence properties of an FEA. However, our third illustrates how a judicious choice of search direction can be leveraged by FEAs to ensure fast convergence to a constrained stationary point. Consider the problem

$$\begin{array}{ll} \min & \sum_{i=1}^{n} x_i^2 \\ \text{s.t.} & 0 \leq x_i \leq 10 \end{array} .$$

This is simply a bound-constrained quadratic, which should be easily solved by any constrained EA.

We consider the relative performance of a simple FEA with a self-adaptive ES (Schwefel, 1995). Specifically, we again consider a simple FEA with $\mu_F = \mu_I = 1$, and $P = 2n$. The step length Δ_t is contracted by a factor of 2 when current point is locally optimal.[2] For comparison purposes, we consider a $(1 + 2n)$-ES which rejects infeasible points, using standard parameters for self-adaptation. Thus the FEA and ES should have similar search dynamics and population sizes. Note that in this particular example, the FEA's search dynamics are deterministic. The ES search dynamics are stochastic, so we consider the performance of the ES over 30 random trials. These optimizers were applied to this problem for several different dimensions, n, and they were terminated after reaching a solution with value $n * 0.001$. Finally, the step scale used by both optimizers was set to 1.0, and the optimizers were started from an initial point near 9^n.

Figure 7 compares the performance of the FEA and ES. The vertical scale of this figure is the number of feasible function evaluations required before termination, normalized by the dimension. Consequently, an ideal line would be flat. Figure 7 illustrates the distribution of the ES results by plotting lines for the 25th, 50th and 75th quartiles of the ES results. It is clear that the ES becomes increasingly more expensive as dimension increases, while the FEA's effort roughly scales with the dimension of the problem.

[2] A contraction factor of 2 is very commonly used with pattern search methods, and it appears to work well in practice.

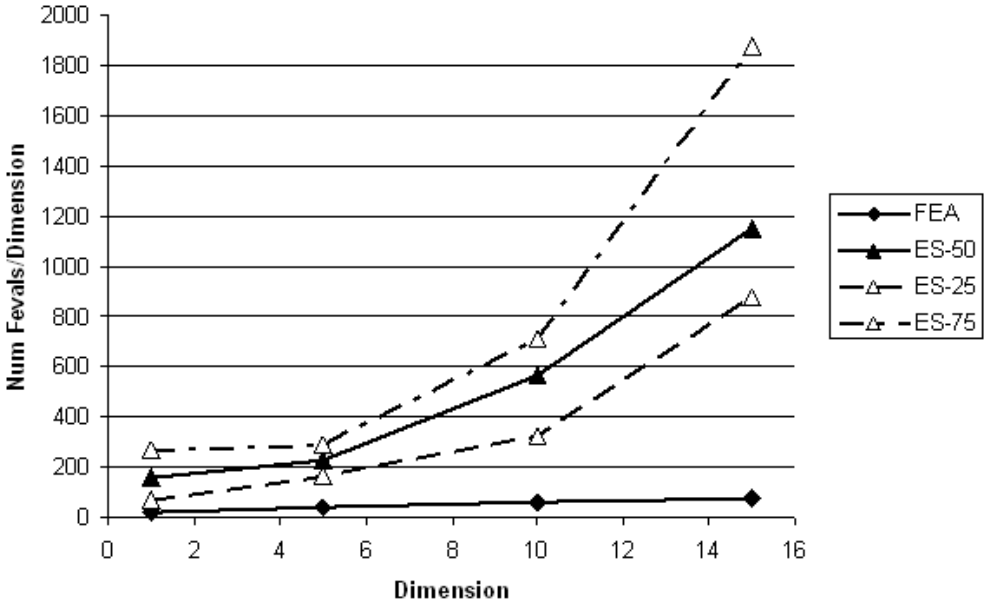


Figure 7: Numerical comparison of self-adapting Evolutionary Strategies and basic pattern search methods.

Although the ES uses a different methodology for adapting the search step scale, it is well-known that ES can effectively minimize quadratic problems. Consequently, the disparity between the ES and FEA performance in this example can be attributed to how they generate new trial points. The search directions used by the FEA are well-aligned with the local constraint boundary near the optimal solution, while the ES's search directions are uniformly scattered in all directions. This suggests that an advantage of FEAs is that they enable a user to exploit domain knowledge about the constraint boundary to improve the (local) search performance.

6 Discussion

Our analysis of FEAs demonstrates that they generate interesting limit points with probability one on a general class of constrained optimization problems. To our knowledge, this is the only convergence theory for EAs on constrained problems that does not require the use of derivative information. Consequently, these results suggest that similar multi-objective constrained EA methods will be effective in practice.

An FEA's use of a positive spanning set, D, is similar to the type of samples that are required for estimating derivative information using standard finite differencing techniques. The connection between EAs to pattern search methods is partially motivated by the fact that so few researchers integrate gradient information into their search process. Thus, it is intentional that FEAs are quite similar to previously formulated constrained EAs; this convergence analysis provides insight into other multi-objective EAs for constrained optimization. More generally, there is a general sense that derivative-free search methods like EAs and pattern search are more effective when derivatives are either difficult to compute or they are noisy (Kolda et al., 2003). Thus, while our conver-

gence of analysis of FEAs relies on standard smoothness assumptions for the objective and constraint functions, we can expect FEAs to work effectively on a broader class of applications.

Theorem 3 is not quite as strong as would be desired, since it does not guarantee convergence to a first-order constrained stationary point. In particular, this result depends on the set of search directions D that are predefined, since this ultimately limits the cone that contains $-\nabla f(\hat{x})$. The limitations of Theorem 3 beg the question of whether or not better analytical approaches can be applied. Our analysis of FEAs is directly related to recent analyses of filter-based pattern search methods (Audet and Dennis, 2004). Although other convergence theories for pattern search methods on constrained problems do not rely on derivative information (Kolda et al., 2003), it is not obvious how these could be adapted for the population-based search used in EAs. For example, Lewis and Torczon (2002) have analyzed the convergence of a pattern search method that solves a series of bound-constrained sub-problems that are used to estimate Lagrange multipliers. Using a similar strategy for EAs would require the application of an EA to a sequence of bound-constrained problems, which would probably limit the EA's ability to search globally. By contrast, the FEA we have considered is closely related to existing multi-objective EAs, which effectively perform global search.

The examples in Section 5 illustrate how the performance of Algorithm A is sensitive to the choice of the search directions, D. We expect that in practice Algorithm A will perform a robust search for constrained local minima if a sufficiently rich set of search directions are employed. For example, the examples in Section 5 employ a single pattern throughout their search, but it is easy to imagine how search patterns could be dynamically adapted using directions from a large, finite set D. More generally, we argue that greater attention needs to be paid to how EAs search near constraint boundaries. Although this is highlighted in our analysis of FEAs, our last experimental example illustrates that this issue is also relevant for well-established EAs like ESs.

Finally, we note that these results have considered a very simple FEA design. However, we expect that these results can be generalized to account for recombination operators, other selection strategies (so long as elitism is enforced), non-rational search directions and per-individual step scale parameters. These types of generalizations have been developed for unconstrained and bound-constrained evolutionary pattern search methods (Hart, 2001, 2003), but a simpler presentation was chosen here to illustrate the core search dynamics of FEAs. For example, the FEA and ES in our third example would have worked more effectively if trial points were mapped onto the constraint boundary, rather than simply being rejected. This is a common mechanism for bound-constraints, and it has been widely used with ESs. Although we expect that this would also be useful for FEAs, this mechanism would require an extension of the FEA convergence theory.

References

Audet, C. and Dennis, Jr., J. E. (2004). A pattern search filter method for nonlinear programming without derivatives. *SIAM J Opt*, 14(4):980–1010.

Camponogara, E. and Talukdar, S. N. (1997). A genetic algorithm for constrained and multiobjective optimization. In Alander, J. T., editor, *3rd Nordic Workshop on Genetic Algorithms and Their Applications*, pages 49–62.

Clarke, F. H. (1990). *Optimization and Nonsmooth Analysis*, volume 5. SIAM Classics in Applied Mathematics, Philadelphia, PA.

Coello, C. A. C. (2000). Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering Systems*, 17:319–346.

Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287.

Deb, K. (2001). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):311–338.

Gill, P. E., Murray, W., and Wright, M. H. (1981). *Practical Optimization*. Academic Press.

Hart, W. E. (2001). Evolutionary pattern search algorithms for unconstrained and linearly constrained optimization. *IEEE Trans Evolutionary Computation*, 5(4):388–397.

Hart, W. E. (2003). Locally-adaptive and memetic evolutionary pattern search algorithms. *Evolutionary Computation*, 11(1):29–52.

Hart, W. E. (2005). Rethinking the design of real-coded evolutionary algorithms: Making discrete choices in continuous search domains. *Soft Computing Journal*, 9:225–235.

Kolda, T. G., Lewis, R. M., and Torczon, V. (2003). Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45(3):385–482.

Lewis, M. and Torczon, V. (2000). Pattern search methods for linearly constrained minimization. *SIAM J Optimization*, 9(4):917–941.

Lewis, R. M. and Torczon, V. J. (2002). A globally convergent augmented Lagrangian pattern search algorithm for optimization with general constraints and simple bounds. *SIAM J Opt*, 12(4):1075–1089.

Ross, K. A. (1980). *Elementary Analysis: The Theory of Calculus*. Springer-Verlag.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking*. John Wiley & Sons, New York.

Smith, A. E. and Coit, D. W. (1997). Constraint handling techniques - penalty functions. In Bäck, T., Fogel, D. B., and Michalewicz, Z., editors, *Handbook of Evolutionary Computation*, chapter C 5.2. Oxford University Press and Institute of Physics Publishing.

A Analysis of Example II

Lemma 5. *If $\overline{x} = (\lambda, \lambda)$ for some $\lambda > 1$ and $\overline{b} = (\lambda + \Delta\sqrt{2}\omega_2, \lambda - \Delta\sqrt{2}\omega_1))$ violates both constraints, then $h(\overline{x}) < h(\overline{a})$ and for sufficiently small Δ, $h(\overline{x}) > h(\overline{c}) = h(\overline{b})$.*

Proof. If $h(\overline{x}) < h(\overline{a})$ then we have $D_1^{\overline{x}} + D_2^{\overline{x}} < D_1^{\overline{a}} + D_2^{\overline{a}}$. Now $D_1^{\overline{x}} = D_2^{\overline{x}} = \frac{8(1-\lambda)^2}{13}$ and $D_1^{\overline{a}} = D_2^{\overline{a}} = \frac{8(1-\lambda-\Delta/\sqrt{2})^2}{13}$, so we need $\frac{16}{13}(1-\lambda)^2 < \frac{16}{13}(1-\lambda-\Delta\sqrt{2})^2$. But $0 > 1-\lambda > 1-\lambda-\Delta/\sqrt{2}$ since $\Delta > 0$. Thus this inequality is always true.

If $h(\overline{x}) > h(\overline{b}) = h(\overline{c})$ then we have $D_1^{\overline{x}} + D_2^{\overline{x}} > D_1^{\overline{b}} + D_2^{\overline{b}}$. Now

$$D_1^{\overline{b}} = \frac{8}{13}\left(1-\lambda+\Delta\sqrt{2}(\omega_2+5\omega_1)/4\right)^2, \text{ and } D_2^{\overline{b}} = \frac{8}{13}\left(1-\lambda-\Delta\sqrt{2}(\omega_1+5\omega_2)/4\right)^2.$$

Thus we need to show that

$$\begin{aligned}\frac{16}{13}(\lambda-1)^2 &> \frac{8}{13}\left(\left(1-\lambda+\Delta\sqrt{2}(\omega_2+5\omega_1)/4\right)^2+\left(1-\lambda-\Delta\sqrt{2}(\omega_1+5\omega_2)/4\right)^2\right)\\ &= \frac{8}{13}\left(2(1-\lambda)^2+2\Delta\sqrt{2}(1-\lambda)(\omega_1-\omega_2)+O(\Delta^2)\right).\end{aligned}$$

Now for sufficiently small values of Δ, the Δ^2 term will be dominated by the Δ term. Further, $\omega_1-\omega_2>0$ so the Δ term is negative. Thus for sufficiently small values of Δ, this inequality is true. □

Lemma 6. *If $\overline{x}=(\lambda,\lambda)$ for some $\lambda>1$ where $D_2^{\overline{b}}=0$, then $h(\overline{b})>h(\overline{x})$.*

Proof. If $h(\overline{b})>h(\overline{x})$ and $D_1^{\overline{b}}=0$, then we must have $D_2^{\overline{b}}>D_1^{\overline{x}}+D_2^{\overline{x}}$. Now $D_1^{\overline{x}}+D_2^{\overline{x}}=\frac{16(1-\lambda)^2}{13}$ and

$$D_2^{\overline{b}}=\frac{8}{13}\left(1-\lambda-\Delta\sqrt{2}(\omega_1+5\omega_2)/4\right)^2.$$

Now since $D_1^{\overline{b}}=0$, we know that $\Delta\geq 2\sqrt{2}(\lambda-1)/(\omega_2+5\omega_1)$. Thus, it suffices to show that

$$\begin{aligned}\frac{8}{13}\left(1-\lambda-\left(\frac{2\sqrt{2}(\lambda-1)}{\omega_2+5\omega_1}\right)\left(\frac{\sqrt{2}(\omega_1+5\omega_2)}{4}\right)\right)^2 &= \frac{8}{13}\left(1-\lambda-(\lambda-1)\frac{\omega_1+5\omega_2}{\omega_2+5\omega_1}\right)^2\\ &= \frac{8}{13}\left(1+\frac{\omega_1+5\omega_2}{\omega_2+5\omega_1}\right)^2(\lambda-1)^2\\ &> \frac{16(1-\lambda)^2}{13}.\end{aligned}$$

But this last inequality is true because $\left(1+\frac{\omega_1+5\omega_2}{\omega_2+5\omega_1}\right)^2>2$. □

Lemma 7. *If $\overline{x}=(\lambda+\Delta\sqrt{2}\omega_2,\lambda-\Delta\sqrt{2}\omega_1))$ for some $\lambda>1$ and $\overline{b}=(\lambda+2\Delta\sqrt{2}\omega_2,\lambda-2\Delta\sqrt{2}\omega_1))$ violates both constraints, then $h(\overline{x})<h(\overline{a})$, $h(\overline{c})<h(\overline{x})$ and $h(\overline{c})<h(\overline{b})$.*

Proof. We have

$$\begin{array}{ll}D_1^{\overline{x}}=\frac{8}{13}\left(-A+BC\right)^2 & D_2^{\overline{x}}=\frac{8}{13}\left(-A-BD\right)^2\\ D_1^{\overline{a}}=\frac{8}{13}\left(-A+B\left(C-\varepsilon\right)\right)^2 & D_2^{\overline{a}}=\frac{8}{13}\left(-A-B\left(D+\varepsilon\right)\right)^2\\ D_1^{\overline{b}}=\frac{8}{13}\left(-A+2BC\right)^2 & D_2^{\overline{b}}=\frac{8}{13}\left(-A-2BD\right)^2\\ D_1^{\overline{c}}=\frac{8}{13}\left(-A+2B\right)^2 & D_2^{\overline{c}}=\frac{8}{13}\left(-A+2B\right)^2\end{array}$$

where $A=\lambda-1$, $B=\frac{\Delta\sqrt{2}}{4}$, $C=\omega_2+5\omega_1$, $D=\omega_1+5\omega_2$, and $\varepsilon=2$.

If $h(\overline{x})<h(\overline{a})$ then we must have $D_1^{\overline{x}}+D_2^{\overline{x}}<D_1^{\overline{a}}+D_2^{\overline{a}}$. Thus we need to show that

$$\begin{aligned}\left(-A+BC\right)^2+\left(-A-BD\right)^2 &< \left(-A+B\left(C-\varepsilon\right)\right)^2+\left(-A-B\left(D+\varepsilon\right)\right)^2\\ &= \left(-A+BC\right)^2+2\left(-2A+B\left(C-D\right)\right)\left(-B\varepsilon\right)+\\ &\quad 2B^2\varepsilon^2+\left(-A-BD\right)^2.\end{aligned}$$

By combining like terms we get

$$0 < 2B^2\varepsilon^2\left(\frac{2A+D-C}{\varepsilon}+1\right).$$

Since $\varepsilon = 2$, $\frac{2A+D-C}{\varepsilon} + 1$ is positive for all values of A and this inequality is true.

If $h(\overline{c}) < h(\overline{x})$ then we must have $D_1^{\overline{c}} + D_2^{\overline{c}} < D_1^{\overline{x}} + D_2^{\overline{x}}$. Thus we need to show

$$2A^2 - 8AB + 8B^2 \quad < \quad 2A^2 + 2ABD - 2ABC + B^2\left(C^2 + D^2\right).$$

Now, by expanding and combining like terms, we get

$$0 \quad < \quad 2AB\left(D - C + 4\right) + B^2\left(C^2 + D^2 - 8\right).$$

Since $(D - C + 4)$ and $C^2 + D^2 - 8$ are both positive this inequality is true.

If $\overline{b}$ violates both constraints and $h(\overline{c}) < h(\overline{b})$ then we must have $D_1^{\overline{c}} + D_2^{\overline{c}} < D_1^{\overline{b}} + D_2^{\overline{b}}$. Thus we need to show

$$2A^2 - 8AB + 8B^2 \quad < \quad 2A^2 + 4AB\left(D - C\right) + 4B^2\left(C^2 + D^2\right).$$

Now by expanding and combining like terms, we get

$$0 \quad < \quad 4AB\left(D - C + 2\right) + 4B^2\left(C^2 + D^2 - 2\right).$$

Since $D - C = -2$, $4AB(D - C + 2) = 0$ and since $4B^2(C^2 + D^2 - 2)$ is positive, this inequality is true. □

Lemma 8. *If $\overline{x} = (\lambda + \Delta\sqrt{2}\omega_2, \lambda - \Delta\sqrt{2}\omega_1)$ for some $\lambda > 1$ where $D_1^{\overline{b}} = 0$, then $h(\overline{x}) < h(\overline{b})$.*

Proof. We have

$$D_1^{\overline{x}} = \frac{8}{13}\left(-A + BC\right)^2, D_2^{\overline{x}} = \frac{8}{13}\left(-A - BD\right)^2 \text{ and } D_2^{\overline{b}} = \frac{8}{13}\left(-A - 2BD\right)^2,$$

where $A = \lambda - 1$, $B = \frac{\Delta\sqrt{2}}{4}$, $C = \omega_2 + 5\omega_1$ and $D = \omega_1 + 5\omega_2$.

If $h(\overline{x}) < h(\overline{b})$ then we must have $D_1^{\overline{x}} + D_2^{\overline{x}} < D_2^{\overline{b}}$. Thus we need to show that

$$2A^2 + 2AB\left(D - C\right) + B^2\left(D^2 + C^2\right) < A^2 + 4ABD + 4B^2D^2.$$

Now, by expanding and combining like terms, we get

$$A^2 - 2AB\left(D + C\right) + B^2\left(C^2 - 3D^2\right) \quad < \quad 0.$$

Since constraint (1) is not violated by $\overline{b}$, we can compute a lower bound on Δ, which gives us a lower bound on B. Additionally, the fact that $\overline{x}$ is feasible gives us an upper bound on B. Specifically, we have $B = \varepsilon\frac{A}{2C}$ where $1 \leq \varepsilon < 2$. Thus we need to show that

$$A^2 - \frac{A^2\varepsilon}{C}\left(D + C\right) + \frac{A^2\varepsilon^2}{4}\left(1 - \frac{3D^2}{C^2}\right) \quad < \quad 0.$$

By factoring out an A^2 we get

$$A^2\left(\frac{\varepsilon^2}{4}\left(1 - \frac{3D^2}{C^2}\right) - \frac{\varepsilon}{2C}\left(D + C\right) + 1\right) < 0.$$

Now $1 > \frac{3D^2}{C^2}$, so the second factor is a convex parabola in ε, with a minimum value at $\varepsilon = \frac{6(9+2\sqrt{3})}{-31+24\sqrt{3}}$. This parabola is negative at $\varepsilon = 1$, and it becomes more negative as ε moves to $\varepsilon = 2$. So this inequality is true. □