SIAM J. OPTIM.
Vol. 7, No. 1, pp. 1–25, February 1997

ON THE CONVERGENCE OF PATTERN SEARCH ALGORITHMS[*]

VIRGINIA TORCZON[†]

Abstract. We introduce an abstract definition of pattern search methods for solving nonlinear unconstrained optimization problems. Our definition unifies an important collection of optimization methods that neither compute nor explicitly approximate derivatives. We exploit our characterization of pattern search methods to establish a global convergence theory that does not enforce a notion of sufficient decrease. Our analysis is possible because the iterates of a pattern search method lie on a scaled, translated integer lattice. This allows us to relax the classical requirements on the acceptance of the step, at the expense of stronger conditions on the form of the step, and still guarantee global convergence.

Key words. unconstrained optimization, convergence analysis, direct search methods, globalization strategies, alternating variable search, axial relaxation, local variation, coordinate search, evolutionary operation, pattern search, multidirectional search, downhill simplex search

AMS subject classifications. 49D30, 65K05

PII. S1052623493250780

1. Introduction.

We consider the familiar problem of minimizing a continuously differentiable function $f : \mathbf{R}^n \to \mathbf{R}$. Direct search methods for this problem are methods that neither compute nor explicitly approximate derivatives of f. Our interest is in a particular subset of direct search methods that we will call *pattern search methods*. Our purpose is to generalize these methods and to present a global convergence theory for them. To our knowledge, this is the first convergence result for some of these methods and the first general convergence theory for all of them.

Examples of pattern search methods include such classical direct search algorithms as *coordinate search* with fixed step sizes, *evolutionary operation* using factorial designs (first proposed by G. E. P. Box [2, 3, 13]), and the original *pattern search algorithm* of Hooke and Jeeves [7]. A more recent example is the *multidirectional search algorithm* of Dennis and Torczon [6, 15]. For some time, it has been apparent to us that the unifying theme that distinguishes these algorithms from other direct search methods is that each of them performs a search using a "pattern" of points that is independent of the objective function f. This informal insight is the basis for our general definition of pattern search methods—it turns out that each of the above pattern search methods is an instance of our general model.

Formally, our definition of pattern search methods requires the existence of a lattice T such that if $\{x_1, \ldots, x_N\}$ are the first N iterates generated by a pattern search method, then there exists a scale factor ϕ_N such that the steps $\{x_1 - x_0, x_2 - x_1, \ldots, x_N - x_{N-1}\}$ all lie in the scaled lattice $\phi_N T$. The lattice depends on the pattern that defines the individual method and on the initial choice of the step length control parameter, but it is independent of the objective function f. The scaling

* Received by the editors June 23, 1993; accepted for publication (in revised form) September 20, 1995. This research was sponsored by Air Force Office of Scientific Research grants 89-0363, F49620-93-1-0212, and F49620-95-1-0210; United States Air Force grant F49629-92-J-0203; and Department of Energy grant DE-FG005-86ER25017. This research was also supported in part by the Geophysical Parallel Computation Project under State of Texas contract 1059.
http://www.siam.org/journals/siopt/7-1/25078.html

† Department of Computer Science, The College of William & Mary, Williamsburg, VA 23187-8795 (va@cs.wm.edu). This work was completed while the author was in the Department of Computational and Applied Mathematics and the Center for Research on Parallel Computation, Rice University, Houston, TX 77251-1892.

depends solely on the sequence of updates that have been applied to the step length control parameter.

Despite isolated convergence results [4, 11, 16] for certain individual pattern search methods, a general theory of convergence for the class of such methods remained elusive for some time. The standard convergence theory for line search and trust region methods depends crucially on some notion of sufficient decrease, but pattern search methods do not enforce any such notion. Therefore, attempts such as [18] to apply the standard theory to pattern search methods arbitrarily introduce some notion of sufficient decrease, thereby modifying the original algorithms. Thus, the challenge was to develop a general convergence theory for pattern search methods without redefining what they are.

Our convergence analysis is guided by that found in Torczon [16] for the multidirectional search algorithm; however, the present level of abstraction makes the important elements of that analysis easier to appreciate. The present paper also includes a correction to the specification of the scaling factors found in [16].

There are three key points to our analysis. First, we show that pattern search methods are descent methods. Second, we prove that pattern search methods are gradient-related methods in the sense of [10]. Finally, we demonstrate that pattern search methods cannot terminate prematurely due to inadequate step length control mechanisms. The crucial element of this analysis is the fact that pattern search methods are able to relax the conditions on accepting a step by enforcing stronger conditions on the step itself. The lattice T, together with the way in which the step length control parameter is updated, prevent a pathological choice of steps: steps of arbitrary lengths along arbitrary search directions are not permitted.

We are able to guarantee that, if the function f is continuously differentiable, then $\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0$ without an explicit representation of the gradient or the directional derivative. In particular, we prove global convergence for pattern search methods despite the fact that they do not explicitly enforce a notion of sufficient decrease on their iterates, such as fraction of Cauchy decrease, fraction of optimal decrease, or the Armijo–Goldstein–Wolfe conditions. However, our convergence analysis does share certain characteristics with the classical convergence analysis of both line search and trust region methods. This connection is both subtle and unexpected.

Our convergence analysis for pattern search methods makes it clear why these methods are as robust as their proponents have long claimed, while clarifying some of the limitations that have long been ascribed to them. In addition, having identified the common structure of these methods, it is now possible to develop new pattern search methods with guaranteed global convergence.

In section 2 we establish the notation and general specification of pattern search methods. In section 3 we prove that if the function to be minimized is continuously differentiable, then pattern search methods guarantee that $\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0$. In addition, we identify the modifications that must be made to pattern search methods to obtain the stronger result $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$. In section 4 we show that the classical pattern search methods mentioned above, as well as the newer multidirectional search algorithm of Dennis and Torczon, conform to the general specification for pattern search methods. In section 5, we give some concluding remarks; section 6 contains technical results needed for the proofs of section 3.

Notation. We denote by $\mathbf{R}$, $\mathbf{Q}$, $\mathbf{Z}$, and $\mathbf{N}$ the sets of real, rational, integer, and natural numbers, respectively.

All norms are Euclidean vector norms or the associated operator norm. We define

$L(y) = \{x : f(x) \leq f(y)\}$, $C(y) = \{x : f(x) = f(y)\}$, and $X_* = \{x : \nabla f(x) = 0\}$.

2. Pattern search methods.

We begin by introducing the following abstraction of pattern search methods. We defer to section 4 demonstrations that the pattern search methods mentioned above fall comfortably within this abstraction.

2.1. The pattern.

To define a pattern we need two components, a *basis matrix* and a *generating matrix*.

The basis matrix can be any nonsingular matrix $B \in \mathbf{R}^{n \times n}$.

The generating matrix is a matrix $C_k \in \mathbf{Z}^{n \times p}$, where $p > 2n$. We partition the generating matrix into components

$$C_k = [M_k \quad -M_k \quad L_k] = [\Gamma_k \quad L_k]. \tag{1}$$

We require that $M_k \in \mathbf{M} \subset \mathbf{Z}^{n \times n}$, where $\mathbf{M}$ is a finite set of nonsingular matrices, and that $L_k \in \mathbf{Z}^{n \times (p-2n)}$ and contains at least one column, the column of zeros.

A *pattern* P_k is then defined by the columns of the matrix $P_k = BC_k$. Because both B and C_k have rank n, the columns of P_k span $\mathbf{R}^n$. For convenience, we use the partition of the generating matrix C_k given in (1) to partition P_k as follows:

$$P_k = BC_k = [BM_k \quad -BM_k \quad BL_k] = [B\Gamma_k \quad BL_k]. \tag{2}$$

Given $\Delta_k \in \mathbf{R}$, $\Delta_k > 0$, we define a *trial step* s_k^i to be any vector of the form

$$s_k^i = \Delta_k B c_k^i\,, \tag{3}$$

where c_k^i denotes a column of $C_k = [c_k^1 \cdots c_k^p]$. Note that Bc_k^i determines the direction of the step, while Δ_k serves as a step length parameter.

At iteration k, we define a *trial point* as any point of the form $x_k^i = x_k + s_k^i$, where x_k is the current iterate.

2.2. The exploratory moves.

Pattern search methods proceed by conducting a series of *exploratory moves* about the current iterate before declaring a new iterate and updating the associated information. These moves can be viewed as sampling the function about the current iterate x_k in a well-defined deterministic fashion in search of a new iterate $x_{k+1} = x_k + s_k$ with a lower function value. The individual pattern search methods are distinguished, in part, by the manner in which these exploratory moves are conducted. To allow the broadest possible choice of exploratory moves and yet still maintain the properties required to prove convergence for the pattern search methods, we place two requirements on the exploratory moves associated with any particular pattern search method. These requirements are given in the following Hypotheses on exploratory moves. (Please note an abuse of notation that is nonetheless convenient: $y \in A$ means that the vector y is contained in the set of columns of the matrix A.)

HYPOTHESES ON EXPLORATORY MOVES.

1. $s_k \in \Delta_k P_k \equiv \Delta_k BC_k \equiv \Delta_k\,[B\Gamma_k \; BL_k]$.
2. If $\min\{f(x_k + y),\; y \in \Delta_k B\Gamma_k\} < f(x_k)$, then $f(x_k + s_k) < f(x_k)$.

The choice of exploratory moves must ensure two things:

1. The direction of any step s_k accepted at iteration k is defined by the pattern P_k, and its length is determined by Δ_k.
2. If simple decrease on the function value at the current iterate can be found among any of the $2n$ trial steps defined by $\Delta_k B\Gamma_k$, then the exploratory moves must produce a step s_k that also gives simple decrease on the function

value at the current iterate. In particular, $f(x_k + s_k)$ need not be less than or equal to $\min\{f(x_k + y),\ y \in \Delta_k B\Gamma_k\}$.
Thus, a legitimate exploratory moves algorithm would be one that somehow guesses which of the steps defined by $\Delta_k P_k$ will produce simple decrease and then evaluates the function at only one such step. (And that step may be contained in $\Delta_k BL_k$ rather than in $\Delta_k B\Gamma_k$.) At the other extreme, a legitimate exploratory moves algorithm would be one that evaluates all p steps defined by $\Delta_k P_k$ and returns the step that produced the least function value.

These are the properties of the exploratory moves that enable us to prove

$$\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0,$$

even though we only require simple decrease on f. Thus we avoid the necessity of enforcing either fraction of Cauchy decrease, fraction of optimal decrease, or the Armijo–Goldstein–Wolfe conditions on the iterates. To obtain

$$\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0,$$

we need to place stronger hypotheses on the exploratory moves as well as place a boundedness condition on the columns of the generating matrices. These extensions will be discussed further in section 3.3.2.

2.3. The generalized pattern search method. Algorithm 1 states the generalized pattern search method for unconstrained minimization.

ALGORITHM 1. THE GENERALIZED PATTERN SEARCH METHOD.

```
Let $x_0 \in \mathbf{R}^n$ and $\Delta_0 > 0$ be given.
For $k = 0, 1, \ldots,$
    (a) Compute $f(x_k)$.
    (b) Determine a step $s_k$ using an exploratory moves algorithm.
    (c) Compute $\rho_k = f(x_k) - f(x_k + s_k)$.
    (d) If $\rho_k > 0$ then $x_{k+1} = x_k + s_k$. Otherwise $x_{k+1} = x_k$.
    (e) Update $C_k$ and $\Delta_k$.
```

To define a particular pattern search method, it is necessary to specify the basis matrix B, the generating matrix C_k, the exploratory moves to be used to produce a step s_k, and the algorithms for updating C_k and Δ_k.

2.4. The updates. Algorithm 2 specifies the requirements for updating Δ_k. The aim of the updating algorithm for Δ_k is to force $\rho_k > 0$. An iteration with $\rho_k > 0$ is *successful*; otherwise, the iteration is *unsuccessful*. Again we note that to accept a step we only require *simple*, as opposed to *sufficient*, decrease.

ALGORITHM 2. UPDATING Δ_k.

```
Given $\tau \in \mathbf{Q}$, let $\theta = \tau^{w_0}$ and $\lambda_k \in \Lambda = \{\tau^{w_1}, \ldots, \tau^{w_L}\}$, where $\tau > 1$ and
$\{w_0, w_1, \ldots, w_L\} \subset \mathbf{Z}$, $L \equiv |\Lambda| < +\infty$, $w_0 < 0$, and $w_i \geq 0$, $i = 1, \ldots, L$.
    (a) If $\rho_k \leq 0$ then $\Delta_{k+1} = \theta\Delta_k$.
    (b) If $\rho_k > 0$ then $\Delta_{k+1} = \lambda_k\Delta_k$.
```

The conditions on θ and Λ ensure that $0 < \theta < 1$ and $\lambda_i \geq 1$ for all $\lambda_i \in \Lambda$. Thus, if an iteration is successful it may be possible to increase the step length parameter Δ_k, but Δ_k is not allowed to decrease. Not surprisingly, this is crucial to the success of the analysis. Also crucial to the analysis is the relationship (overlooked in [16]) between θ and the elements of Λ.

The algorithm for updating C_k depends on the pattern search method. For theoretical purposes, it is sufficient to choose the columns of C_k so that they satisfy (1) and the conditions we have placed on the matrices $M_k \in \mathbf{M} \subset \mathbf{Z}^{n \times n}$ and $L_k \in \mathbf{Z}^{n\times(p-2n)}$.

3. The convergence theory. Having set up the machinery to define pattern search methods, we are now ready to analyze these methods. This analysis produces theorems of several types. The first, developed in section 3.1, demonstrates an algebraic fact about the nature of pattern search methods that requires no assumption on the function f. This theorem is critical to the proof of the convergence results for it shows that we only need require simple decrease in f to ensure global convergence. The second theorem, developed in section 3.2, describes the limiting behavior of the step length control parameter Δ_k if we place only a very mild condition on the function f and exploit the interaction of the simple decrease condition for the generalized pattern search method with the algorithm for updating Δ_k. Finally, the third and fourth theorems, developed in section 3.3, give the global convergence results. The first theorem guarantees $\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0$ for any generalized pattern search method that satisfies the specifications given in section 2. This is significant since the theorem applies to all the pattern search methods we discuss in section 4 without the need to impose any modifications on the methods as originally stated. The second theorem is equivalent to convergence results for line search and trust-region globalization strategies. We can guarantee $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$, but to do so requires placing stronger conditions on the specifications for generalized pattern search methods. We could certainly impose these stronger conditions on the pattern search methods presented in section 4—none of them are unreasonable to suggest or to enforce—but we would do so at the expense of attractive algorithmic features found in the original methods.

3.1. The algebraic structure of the iterates. The results found in this section are purely algebraic facts about the nature of pattern search methods; they are also independent of the function to be optimized. It is the algebraic structure of the iterates that allows us to prove global convergence for pattern search methods without imposing a notion of sufficient decrease on the iterates.

We begin by showing in what sense Δ_k is a step length parameter.

LEMMA 3.1. *There exists a constant $\zeta_* > 0$, independent of k, such that for any trial step $s_k^i \neq 0$ produced by a generalized pattern search method (Algorithm 1) we have*

$$\|s_k^i\| \geq \zeta_* \Delta_k.$$

Proof. From (3) we have $s_k^i = \Delta_k B c_k^i$. The conditions we have placed on the generating matrix C_k ensure that $c_k^i \in \mathbf{Z}^n$.

Let $\sigma_n(B)$ denote the smallest singular value of B. Then

$$\|s_k^i\| \quad = \quad \Delta_k \|B c_k^i\| \quad \geq \quad \Delta_k \sigma_n(B) \|c_k^i\| \quad \geq \quad \Delta_k \sigma_n(B).$$

The last inequality holds because at least one of the components of c_k^i is a nonzero integer, and hence $\|c_k^i\| \geq 1$. ▯

From Lemma 3.1 we can see that the role of Δ_k as a step length parameter is to regulate backtracking and thus prevent excessively short steps.

THEOREM 3.2. *Any iterate x_N produced by a generalized pattern search method*

(Algorithm 1) can be expressed in the following form:

$$x_N = x_0 + \left(\beta^{r_{LB}} \alpha^{-r_{UB}}\right) \Delta_0 B \sum_{k=0}^{N-1} z_k,$$

where

- x_0 *is the initial guess,*
- $\beta/\alpha \equiv \tau$, *with* $\alpha, \beta \in \mathbf{N}$ *and relatively prime, and* τ *is as defined in the algorithm for updating* Δ_k *(Algorithm 2),*
- r_{LB} *and* r_{UB} depend on N,
- Δ_0 *is the initial choice for the step length control parameter,*
- B *is the basis matrix, and*
- $z_k \in \mathbf{Z}^n$, $k = 0, \ldots, N-1$.

Proof. The generalized pattern search algorithm, as stated in Algorithm 1, guarantees that any iterate x_N is of the form

$$x_N = x_0 + \sum_{k=0}^{N-1} s_k. \tag{4}$$

(We adopt the convention that $s_k = 0$ if iteration k is unsuccessful.) We also know that the step s_k must come from the set of trial steps s_k^i, $i = 1, \ldots, p$. The trial steps are of the form $s_k^i = \Delta_k B c_k^i$.

Consider the step length parameter Δ_k. For any $k \geq 0$, the update for Δ_k given in Algorithm 2 guarantees that Δ_k is of the form

$$\Delta_k = \theta^{q_k^0} \lambda_1^{q_k^1} \lambda_2^{q_k^2} \cdots \lambda_L^{q_k^L} \Delta_0, \tag{5}$$

where $q_k^i \in \mathbf{Z}$ and $q_k^i \geq 0$. (Recall that $L = |\Lambda|$.) We have also placed the following restrictions on the form of θ and λ_i: for a given $\tau \in \mathbf{Q}$, $\tau > 1$, and $\{w_0, w_1, \ldots, w_L\} \subset \mathbf{Z}$, $\theta = \tau^{w_0}$, $w_0 < 0$ and $\lambda_i = \tau^{w_i}$, $w_i \geq 0$, $i = 1, \ldots, L$. We can thus rewrite (5) as:

$$\Delta_k \quad = \quad \left(\tau^{w_0}\right)^{q_k^0} \left(\tau^{w_1}\right)^{q_k^1} \left(\tau^{w_2}\right)^{q_k^2} \cdots \left(\tau^{w_L}\right)^{q_k^L} \Delta_0 \quad = \quad \tau^{r_k} \Delta_0, \tag{6}$$

where $r_k \in \mathbf{Z}$. Let

$$r_{LB} = \min_{0 \leq k < N} \{r_k\} \qquad r_{UB} = \max_{0 \leq k < N} \{r_k\}. \tag{7}$$

Then from (4) and (6) we have

$$x_N = x_0 + \sum_{k=0}^{N-1} \Delta_k B c_k = x_0 + \Delta_0 B \sum_{k=0}^{N-1} \tau^{r_k} c_k.$$

Since τ is rational, we can express τ as $\tau = \frac{\beta}{\alpha}$, where $\alpha, \beta \in \mathbf{N}$ are relatively prime. Then, using (7),

$$x_N = x_0 + \left(\beta^{r_{LB}} \alpha^{-r_{UB}}\right) \Delta_0 B \sum_{k=0}^{N-1} z_k, \tag{8}$$

where $z_k \in \mathbf{Z}^n$. $\square$

Theorem 3.2 synthesizes the requirements we have placed on the pattern, the definition of the trial steps, and the algorithm for updating Δ_k. Note that this means that for a fixed N, all the iterates lie on a translated integer lattice generated by x_0 and the columns of $\beta^{r_{LB}} \alpha^{-r_{UB}} \Delta_0 B$.

3.2. The limiting behavior of the step length control parameter. The next theorem combines the strict algebraic structure of the iterates with the simple decrease condition of the generalized pattern search algorithm, along with the algorithm for updating Δ_k, to give us a useful fact about the limiting behavior of Δ_k.

THEOREM 3.3. *Assume that $L(x_0)$ is compact. Then* $\liminf_{k\to+\infty} \Delta_k = 0$.

Proof. The proof is by contradiction. Suppose $0 < \Delta_{LB} \le \Delta_k$ for all k. From (6) we know that Δ_k can be written as $\Delta_k = \tau^{r_k}\Delta_0$, where $r_k \in \mathbf{Z}$.

The hypothesis that $\Delta_{LB} \le \Delta_k$ for all k means that the sequence $\{\tau^{r_k}\}$ is bounded away from zero. Meanwhile, we also know that the sequence $\{\Delta_k\}$ is bounded above because all the iterates x_k must lie inside the set $L(x_0) = \{x : f(x) \le f(x_0)\}$, and the latter set is compact; Lemma 3.1 then guarantees an upper bound Δ_{UB} for $\{\Delta_k\}$. This, in turn, means that the sequence $\{\tau^{r_k}\}$ is bounded above. Consequently, the sequence $\{\tau^{r_k}\}$ is a finite set. Equivalently, the sequence $\{r_k\}$ is bounded above and below.

Let

$$r_{LB} = \min_{0\le k<+\infty}\{r_k\} \qquad r_{UB} = \max_{0\le k<+\infty}\{r_k\}. \tag{9}$$

Then (8) now holds for the bounds given in (9), rather than (7), and we see that for all k, x_k lies in the translated integer lattice G generated by x_0 and the columns of $\beta^{r_{LB}}\alpha^{-r_{UB}}\Delta_0 B$.

The intersection of the compact set $L(x_0)$ with the translated integer lattice G is finite. Thus, there must exist at least one point x_* in the lattice for which $x_k = x_*$ for infinitely many k.

We appeal to the simple decrease condition in the generalized pattern search method (Algorithm 1 (d)), which guarantees that a lattice point cannot be revisited infinitely many times since we accept a new step s_k if and only if $f(x_k) > f(x_k + s_k)$. Thus there exists an N such that for all $k \ge N$, $x_k = x_*$, which implies that $\rho_k = 0$.

We now appeal to the algorithm for updating Δ_k (Algorithm 2 (a)) to see that $\Delta_k \to 0$, thus leading to a contradiction. ◻

3.3. Global convergence. Throughout the discussion in this section, we assume that f is continuously differentiable on a neighborhood of $L(x_0)$; however, this assumption can be weakened, using the same style of argument found in [16].

3.3.1. The general result. To prove Theorem 3.5 we need Proposition 3.4. We defer the proof of Proposition 3.4 to section 6 in part because we wish to discuss there several other issues that are tangential to the proof of Theorem 3.5. It is also the case that the proofs for the results in section 6 are similar to those given for the equivalent results found in [16], though now restated more succinctly in terms of the machinery developed in section 2.

PROPOSITION 3.4. *Assume that $L(x_0)$ is compact, that f is continuously differentiable on a neighborhood of $L(x_0)$, and that* $\liminf_{k\to+\infty}\|\nabla f(x_k)\| \ne 0$. *Then there exists a constant $\Delta_{LB} > 0$ such that for all k, $\Delta_k > \Delta_{LB}$.*
We emphasize that the existence of a positive lower bound Δ_{LB} for Δ_k is guaranteed only under the null hypothesis that $\liminf_{k\to+\infty}\|\nabla f(x_k)\| \ne 0$.

THEOREM 3.5. *Assume that $L(x_0)$ is compact and that f is continuously differentiable on a neighborhood of $L(x_0)$. Then for the sequence of iterates $\{x_k\}$ produced by the generalized pattern search method (Algorithm 1),*

$$\liminf_{k\to+\infty}\|\nabla f(x_k)\| = 0\,.$$

Proof. The proof is by contradiction. Suppose that $\liminf_{k\to+\infty} \|\nabla f(x_k)\| \neq 0$. Then Proposition 3.4 tells us that there exists $\Delta_{LB} > 0$ such that for all k, $\Delta_k \geq \Delta_{LB}$. But this contradicts Theorem 3.3. ∎

3.3.2. The stronger result. We can strengthen the result given in Theorem 3.5 at the expense of wider applicability. To begin with, we must add three further restrictions: one on the pattern matrix, one on the Hypotheses on exploratory moves, and one on the limiting behavior of the step length control parameter Δ_k.

First, we must ensure that the columns of the generating matrix C_k are bounded in norm, i.e., that there exists a constant $\mathcal{C} > 0$ such that for all k, $\mathcal{C} > \|c_k^i\|$ for all $i = 1, \ldots, p$. Given this bound, we can place an upper bound, in terms of Δ_k, on the norm of any trial step s_k^i.

LEMMA 3.6. *Given a constant $\mathcal{C} > 0$ such that for all k, $\mathcal{C} > \|c_k^i\|$ for all $i = 1, \ldots, p$, there exists a constant $\psi_* > 0$, independent of k, such that for any trial step s_k^i produced by a generalized pattern search method (Algorithm 1) we have*

$$\Delta_k \geq \psi_* \|s_k^i\|.$$

Proof. From (3) we have $s_k^i = \Delta_k B c_k^i$. Then $\|s_k^i\| = \Delta_k \|B c_k^i\| \leq \Delta_k \|B\| \|c_k^i\| \leq \Delta_k \mathcal{C} ||B||$. Set $\psi_* = \frac{1}{\mathcal{C}||B||}$. ∎

Note that the columns of $M_k \in \mathbf{M}$ are bounded by the assumption that $|\mathbf{M}| < +\infty$; we use this fact in the proof of Proposition 6.4. The stronger boundedness condition on the columns of $C_k = [M_k \; -M_k \; L_k]$ is needed to monitor the behavior of L_k.

Second, we must replace the original Hypotheses on exploratory moves with a stronger version, as given below. Together, Lemma 3.6 and the Strong hypotheses on exploratory moves allow us to tie decrease in f to the norm of the gradient when the step sizes get small enough. This is the import of Corollary 6.5, which is given in section 6.

STRONG HYPOTHESES ON EXPLORATORY MOVES.

1. $s_k \in \Delta_k P_k \equiv \Delta_k B C_k \equiv \Delta_k \, [B\Gamma_k \; BL_k]$.
2. If $\min\{f(x_k + y), \; y \in \Delta_k B\Gamma_k\} < f(x_k)$, then
   $f(x_k + s_k) \leq \min\{f(x_k + y), \; y \in \Delta_k B\Gamma_k\}$.

Third, we require that $\lim_{k\to+\infty} \Delta_k = 0$. We can use the algorithm for updating Δ_k (Algorithm 2) to ensure that this condition holds. For instance, we can force Δ_k to be nonincreasing by requiring $w_i = 0$, $i = 1, \ldots, L$, which when taken together with Theorem 3.3 guarantees that $\lim_{k\to+\infty} \Delta_k = 0$. All the algorithms we consider in section 4, except the multidirectional search algorithm, enforce this condition by limiting $\Lambda = \{1\} \equiv \{\tau^0\}$. However, it is not necessary to force the steps to be nonincreasing; we need only require that in the limit the step length control parameter goes to zero, which, in conjunction with Lemmas 3.1 and 3.6, has the effect of ultimately forcing the steps to zero.

THEOREM 3.7. *Assume that $L(x_0)$ is compact and that f is continuously differentiable on a neighborhood of $L(x_0)$. In addition, assume that the columns of the generating matrices are bounded in norm, that $\lim_{k\to+\infty} \Delta_k = 0$, and that the generalized pattern search method (Algorithm 1) enforces the Strong hypotheses on exploratory moves. Then for the sequence of iterates $\{x_k\}$ produced by the generalized pattern search method,*

$$\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0\,.$$

Proof. The proof is by contradiction. Suppose $\limsup_{k\to+\infty} \|\nabla f(x_k)\| \neq 0$. Let $\varepsilon > 0$ be such that there exists a subsequence $\|\nabla f(x_{m_i})\| \geq \varepsilon$. Since

$$\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0,$$

given any $0 < \eta < \varepsilon$, there exists an associated subsequence l_i such that

$$\|\nabla f(x_k)\| \quad > \eta \qquad \text{for} \qquad m_i \leq k < l_i, \qquad \|\nabla f(x_{l_i})\| \quad < \quad \eta.$$

Then, since $\Delta_k \to 0$, we can appeal to Corollary 6.5 to obtain for $m_i \leq k < l_i$, i sufficiently large,

$$f(x_k) - f(x_{k+1}) \quad \geq \quad \sigma\|\nabla f(x_k)\|\|s_k\| \quad \geq \quad \sigma\eta\|s_k\|,$$

where $\sigma > 0$. Then the telescoping sum

$$(f(x_{m_i}) - f(x_{m_i+1})) + (f(x_{m_i+1}) - f(x_{m_i+2})) + \cdots + (f(x_{l_i-1}) - f(x_{l_i})) \geq \sum_{k=m_i}^{l_i} \sigma\eta\|s_k\|$$

gives us

$$f(x_{m_i}) - f(x_{l_i}) \quad \geq \quad \textstyle\sum_{k=m_i}^{l_i} \sigma\eta\|s_k\| \quad \geq \quad c'\|x_{m_i} - x_{l_i}\|.$$

Since f is bounded below, $f(x_{m_i}) - f(x_{l_i}) \to 0$ as $i \to +\infty$, so $\|x_{m_i} - x_{l_i}\| \to 0$ as $i \to +\infty$. Then, because ∇f is uniformly continuous,

$$\|\nabla f(x_{m_i}) - \nabla f(x_{l_i})\| < \eta$$

for i sufficiently large. However,

$$\|\nabla f(x_{m_i})\| \quad \leq \quad \|\nabla f(x_{m_i}) - \nabla f(x_{l_i})\| + \|\nabla f(x_{l_i})\| \quad \leq \quad 2\eta. \tag{10}$$

Since equation (10) must hold for any η, $0 < \eta < \varepsilon$, we have a contradiction (e.g., try $\eta = \frac{\varepsilon}{4}$). $\square$

The proof of Theorem 3.7 is almost identical to that of an equivalent result for trust-region methods that was first given by Thomas [14] and which is included, in a more general form, in the survey by Moré [8].

One final note: the hypotheses of Theorem 3.7 suggest that in the absence of any explicit higher-order information about the function to be minimized, it makes sense to terminate a generalized pattern search algorithm when Δ_k is less than some reasonably small tolerance. In fact, this is a common stopping condition for algorithms of this sort and the one implemented for the multidirectional search algorithm [17].

4. The particular pattern search methods. In section 2 we stated the conditions an algorithm must satisfy to be a pattern search method. We now illustrate these conditions by considering the following specific algorithms:

- coordinate search with fixed step lengths,
- evolutionary operation using factorial designs [2, 3, 13],
- the original pattern search method of Hooke and Jeeves [7], and
- the multidirectional search algorithm of Dennis and Torczon [6, 15].

We will show that these algorithms satisfy the conditions that define pattern search methods and thus are special cases of the generalized pattern search method presented as Algorithm 1. Then we can appeal to Theorem 3.5 to claim global convergence for these methods.

There are other algorithms for which the abstraction and accompanying analysis holds—including various modifications to the algorithms presented—but we shall confine our investigation to these, the best known of the pattern search methods, to illustrate the power of our abstract approach to pattern search methods.

4.1. Coordinate search with fixed step lengths. The method of coordinate search is perhaps the simplest and most obvious of all the pattern search methods. Davidon describes it concisely in the opening of his belated preface to Argonne National Laboratory Research and Development Report 5990 [5]:

> Enrico Fermi and Nicholas Metropolis used one of the first digital computers, the Los Alamos Maniac, to determine which values of certain theoretical parameters (phase shifts) best fit experimental data (scattering cross sections). They varied one theoretical parameter at a time by steps of the same magnitude, and when no such increase or decrease in any one parameter further improved the fit to the experimental data, they halved the step size and repeated the process until the steps were deemed sufficiently small. Their simple procedure was slow but sure....

This simple search method enjoys many names, among them *alternating directions*, *alternating variable search*, *axial relaxation*, and *local variation*. We shall refer to it as *coordinate search*.

Perhaps less obvious is that coordinate search is a pattern search method. To see this, we begin by considering all possible outcomes for a single iteration of coordinate search when $n = 2$, as shown in Fig. 1. We mark the current iterate x_k. The x_k^i's denote trial points considered during the course of the iteration. The next iterate x_{k+1} is marked. Solid circles indicate successful intermediate steps taken during the course of the exploratory moves while open circles indicate points at which the function was evaluated but that did not produce further decrease in the value of the objective function. Thus, in the first scenario shown a step from x_k to x_k^1 resulted in a decrease in the objective function, so the step from x_k^1 to x_{k+1} was tried and led to a further decrease in the objective function value. The iteration was then terminated with a new point x_{k+1} that satisfies the simple decrease condition $f(x_{k+1}) < f(x_k)$. In the worst case, the last scenario shown, $2n$ trial points were evaluated (x_k^1, $x_k^{1'}$, x_k^2, and $x_k^{2'}$) without producing decrease in the function value at the current iterate x_k. In this case, $x_{k+1} = x_k$ and the step size must be reduced for the next iteration.

We now show this algorithm is an instance of a generalized pattern search method.

4.1.1. The matrices. Coordinate search is usually defined so that the basis matrix is the identity matrix; i.e., $B = I$. However, knowledge of the problem may lead to a different choice for the basis matrix. It may make sense to search using a different coordinate system. For instance, if the variables are known to differ by several orders of magnitude, this can be taken into account in the choice of the basis matrix (though, as we will see in section 6.2, this may have a significant effect on the behavior of the method).

The generating matrix for coordinate search is fixed across all iterations of the method. The generating matrix $C_k = C$ contains in its columns all possible combi-

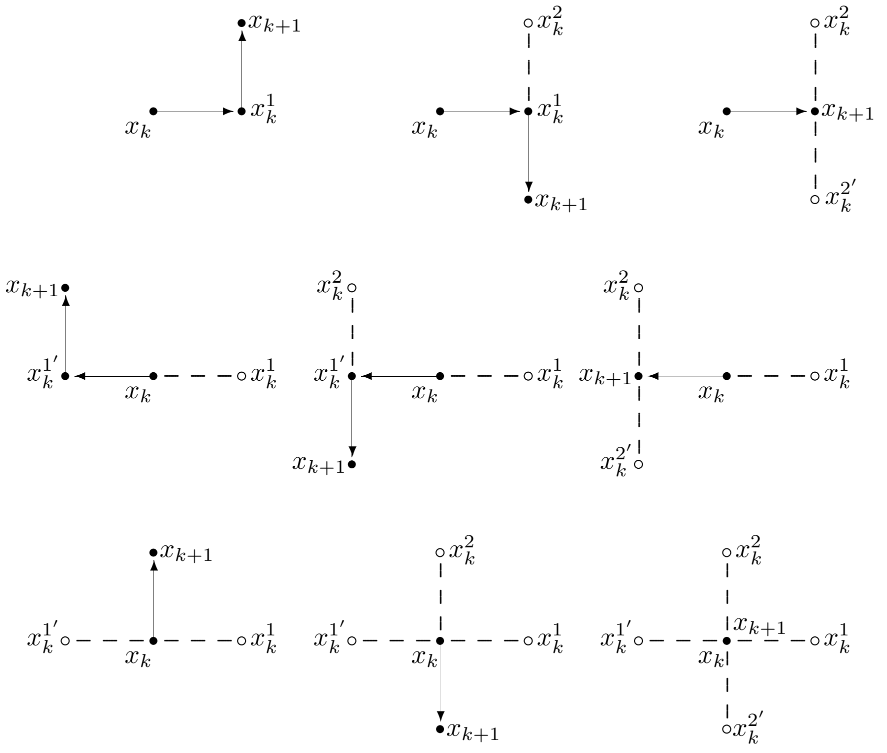

FIG. 1. *All possible subsets of the steps for coordinate search in* $\mathbf{R}^2$.

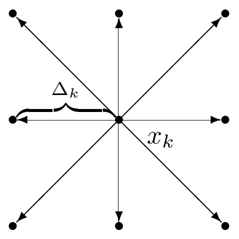

FIG. 2. *The pattern for coordinate search in* $\mathbf{R}^2$ *with a given step length control parameter* Δ_k.

nations of $\{-1, 0, 1\}$. Thus, C has $p = 3^n$ columns. In particular, the columns of C contain both I and $-I$, as well as a column of zeros. We define $M = I$; L consists of the remaining $3^n - 2n$ columns of C. Since C is fixed across all iterations of the method, there is no need for an update algorithm.

For $n = 2$ we have

$$C = \begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \end{bmatrix}.$$

Thus, when $n = 2$, all possible trial points defined by the pattern $P = BC$, for a given step length Δ_k, can be seen in Fig. 2. Note that the pattern includes all the possible trial points enumerated in Fig. 1.

4.1.2. The exploratory moves. The exploratory moves for coordinate search are given in Algorithm 3, where the e_i's denote the unit coordinate vectors.

Algorithm 3. Exploratory Moves Algorithm for Coordinate Search.

```
Given $x_k$, $\Delta_k$, $f(x_k)$, and $B$, set $s_k = 0$, $\rho_k = 0$, and min $= f(x_k)$.
For $i = 1, \ldots, n$ do
    (a) $s_k^i = s_k + \Delta_k B e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
    (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
        Otherwise,
         (i) $s_k^i = s_k - \Delta_k B e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
        (ii) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
Return.
```

The exploratory moves are executed sequentially in the sense that the selection of the next trial step is based on the success or failure of the previous trial step. Thus, while there are 3^n possible trial steps, we may compute as few as n trial steps, but we compute no more than $2n$ at any given iteration, as we saw in Fig. 1.

From the perspective of the theory, there are two conditions that need to be met by the exploratory moves algorithm. First, as Figs. 1 and 2 illustrate, all possible trial steps are contained in $\Delta_k P$.

The second condition on the exploratory moves is the more interesting; coordinate search demonstrates the laxity of this second hypothesis. For instance, in the first scenario shown in Fig. 1, decrease in the objective function was realized for the first trial step

$$s_k^1 = \Delta_k I \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

so the second trial step

$$s_k^2 = \Delta_k I \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \Delta_k I \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \Delta_k I \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

was tried and accepted. It is certainly possible that greater decrease in the value of the objective function might have been realized for the trial step

$$s_k^{'} = \Delta_k I \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

which is defined by a column in the matrix M (the step s_k^2 is defined by a column in the matrix L), but $s_k^{'}$ is not tried when simple decrease is realized by the step s_k^1. However, in the worst case, as seen in Fig. 1, the algorithm for coordinate search ensures that all $2n$ steps defined by $\Delta_k B \Gamma = \Delta_k B[M\ -M] = \Delta_k B[I\ -I]$ are tried before returning the step $s_k = 0$. In other words, the exploratory moves given in Algorithm 3 examine all $2n$ steps defined by $\Delta_k B\Gamma$ unless a step satisfying $f(x_k + s_k) < f(x_k)$ is found.

4.1.3. Updating the step length. The update for Δ_k is exactly as given in Algorithm 2. As noted by Davidon, the usual practice is to continue with steps of the same magnitude until no further decrease in the objective function is realized, at which point the step size is halved. This corresponds to setting $\theta = 1/2$ and $\Lambda = \{1\}$. Thus, $\tau = 2$, $w_0 = -1$, and $w_1 = 0$.

This suffices to verify that coordinate search with fixed step length is a pattern search method. Theorem 3.5 thus holds. The exploratory moves algorithm for coordinate search would need to be modified to satisfy the Strong hypotheses on exploratory moves for the conditions of Theorem 3.7 to be met.

4.2. Evolutionary operation using factorial designs. In 1957 G. E. P. Box [2] introduced the notion of evolutionary operation as a method for increasing industrial productivity. The ideas were developed within the context of the on-line management of industrial processes, but Box recognized that the technique had more general applicability. Subsequent authors [3, 13] argued that the basic technique was readily applicable to general unconstrained optimization and it is within this context that we examine the ideas here.

In its simplest form, evolutionary operation is based on using two-level factorial designs: evaluate the function at the vertices of a hypercube centered about the current iterate. (G. E. P. Box refers to this as one of a variety of "pattern of variants" [2].) If simple decrease in the value of the objective function is observed at one of the vertices, it becomes the new iterate. Otherwise, the lengths of the edges in the hypercube are halved and the process is repeated.

4.2.1. The matrices. As with coordinate search, the usual choice for the basis matrix is $B = I$, though, as with coordinate search, other choices may be made to reflect information known about the problem to be solved.

The generating matrix for evolutionary operation is fixed across all iterations of the method. The generating matrix $C_k = C$ contains in its columns all possible combinations of $\{-1, 1\}$; to this we append a column of zeros. Thus C has $p = 2^n + 1$ columns.

We take M to be any linearly independent subset of n columns of C; $-M$ necessarily will be contained in C. Once again, L is fixed and consists of the remaining $(2^n + 1) - 2n$ columns of C.

There is no need for an algorithm to update C since the generating matrix is fixed.

4.2.2. The exploratory moves. The exploratory moves given in Algorithm 4 are simultaneous in the sense that every possible trial step $s_k^i \in \Delta_k P = \Delta_k BC$ is computed at each iteration. It is then the case that every trial step s_k^i is contained in $\Delta_k P$. The second observation of note is that since

$$s_k = \operatorname*{arg\,min}_{s_k^i \in \Delta_k P} \{f(x_k + s_k^i)\},$$

then, if $\min\{f(x_k + y), y \in \Delta_k B\Gamma\} < f(x_k)$, we have $f(x_k + s_k) < f(x_k)$, regardless of our choice of M (and thus, by extension, our choice of Γ). Furthermore, we are guaranteed that the Strong hypotheses on exploratory moves are satisfied.

Algorithm 4. Exploratory Moves Algorithm for Evolutionary Operation.

```
Given $x_k$, $\Delta_k$, $f(x_k)$, $B$, and $C = \left[c^1 \cdots c^p\right]$, set $s_k = 0$, $\rho_k = 0$, and min $= f(x_k)$.
For $i = 1, \ldots, 2^n$ do
    (a) $s_k^i = \Delta_k B c^i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
    (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
Return.
```

4.2.3. Updating the step length. The algorithm for updating Δ_k is exactly as given in Algorithm 2, with θ usually set to $1/2$ and $\Lambda = \{1\}$.

Since we have shown that evolutionary operation satisfies all the necessary requirements, we can therefore conclude that it, too, is a pattern search method, so Theorem 3.5 holds. The algorithm, as stated above, also satisfies the conditions of Theorem 3.7.

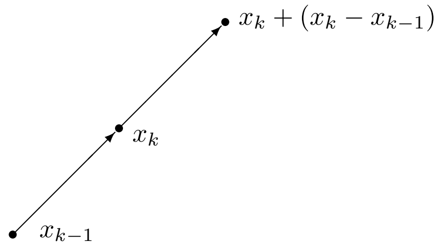

FIG. 3. *The pattern step in* $\mathbf{R}^2$*, given* $x_k \neq x_{k-1}$*,* $k > 0$*.*

4.3. Hooke and Jeeves' pattern search algorithm. In addition to introducing the general notion of a "direct search" method, Hooke and Jeeves introduced the pattern search method, a specific kind of search strategy [7]. The pattern search of Hooke and Jeeves is a variant of coordinate search that incorporates a *pattern step* in an attempt to accelerate the progress of the algorithm by exploiting information gained from the search during previous successful iterations.

The Hooke and Jeeves pattern search algorithm is opportunistic. If the previous iteration was successful (i.e., $\rho_{k-1} > 0$), then the current iteration begins by conducting coordinate search about a speculative iterate $x_k + (x_k - x_{k-1})$, rather than about the current iterate x_k. This is the pattern step. The idea is to investigate whether further progress is possible in the general direction $x_k - x_{k-1}$ (since, if $x_k \neq x_{k-1}$, then $x_k - x_{k-1}$ is clearly a promising direction).

To make this a little clearer, we consider the example shown in Fig. 3. Given x_{k-1} and x_k (we assume, for now, that $k > 0$ and that $x_k \neq x_{k-1}$), the pattern search algorithm takes the step $x_k - x_{k-1}$ from x_k. The function is evaluated at this trial step and the trial step is accepted, temporarily, even if $f(x_k + (x_k - x_{k-1})) \geq f(x_k)$. The Hooke and Jeeves pattern search algorithm then proceeds to conduct coordinate search about the temporary iterate $x_k + (x_k - x_{k-1})$. Thus, in $\mathbf{R}^2$, the exploratory moves are exactly as shown in Fig. 1, but with $x_k + (x_k - x_{k-1})$ substituted for x_k.

If coordinate search about the temporary iterate $x_k + (x_k - x_{k-1})$ is successful, then the point returned by coordinate search about the temporary iterate is accepted as the new iterate x_{k+1}. If not, i.e., $f((x_k + (x_k - x_{k-1})) + s_k) \geq f(x_k)$, then the pattern step is deemed unsuccessful, and the method reduces to coordinate search about x_k. For the two dimensional case, then, the exploratory moves would simply resort to the possibilities shown in Fig. 1.

If the previous iteration was not successful, so $x_k = x_{k-1}$ and $(x_k - x_{k-1}) = 0$, then the iteration is limited to coordinate search about x_k. In this instance, though, the updating algorithm for Δ_k will have reduced the size of the step (i.e., $\Delta_k = \theta \Delta_{k-1}$).

The algorithm does not execute the pattern step when $k = 0$.

To express the pattern search algorithm within the framework we have developed, we use all the machinery required for coordinate search. Once again, the basis matrix is usually defined to be $B = I$. We append to the generating matrix another set of 3^n columns to capture the effect of the pattern step and we change the exploratory moves algorithm, as detailed below.

4.3.1. The generating matrix. Recall that the generating matrix for coordinate search consists of all possible combinations of $\{-1, 0, 1\}$ and is never changed. For the Hooke and Jeeves pattern search method, we allow the generating matrix to change from iteration to iteration to capture the effect of the pattern step. We append another set of 3^n columns, consisting of all possible combinations of $\{-1, 0, 1\}$, to the initial generating matrix for coordinate search. Thus C_k has $p = 2 \cdot 3^n$ columns. The additional 3^n columns allow us to express the effect of the pattern step with respect to x_k, rather than with respect to the temporary iterate $x_k + (x_k - x_{k-1})$, which is how the Hooke and Jeeves pattern search method usually is described. The matrix M is unchanged; $M = I$. Now, however, $L_k \in \mathbf{Z}^{n \times (p-2n)}$ is allowed to vary, though only in the 3^n columns associated with the pattern step. For $n = 2$,

$$C_0 = \left[\begin{array}{ccccccccc} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \end{array} \right. \\ \left. \begin{array}{ccccccccc} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \end{array} \right]. \tag{11}$$

For notational convenience, we require that the last column of C_0, which we denote as c_0^p, be the column of zeros. In both the algorithm for updating C_k (Algorithm 5) and the algorithm for the exploratory moves (Algorithm 6), we use the column c_k^p to measure the accumulation of a sequence of successful pattern steps. This can be seen, in (12), for our example from Fig. 3. In this example, we have the generating matrix

$$C_k = \left[\begin{array}{ccccccccc} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \end{array} \right. \\ \left. \begin{array}{ccccccccc} 2 & 1 & 0 & 1 & 2 & 2 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 2 & 0 & 0 & 2 & 1 \end{array} \right]. \tag{12}$$

The pattern step $(x_k - x_{k-1})$ is represented by the vector $(1\ 1)^T$, seen in the last column of C_k. Note that the only difference between the columns of C_0 given in (11) and the columns of C_k given in (12) is that $(1\ 1)^T$ has been added to the last 3^2 columns of C_k.

The algorithm for updating the generating matrix updates the last 3^n columns of C_k; the first 3^n columns remain unchanged, as in coordinate search. The purpose of the updating algorithm is to incorporate the result of the search at the current iteration into the pattern for the next iteration. This is done using Algorithm 5. Note the distinguished role of c_k^p, the last column of C_k, which represents the pattern step $(x_k - x_{k-1})$.

ALGORITHM 5. UPDATING C_k.

```
For $i = 3^n + 1, \ldots, 2 \cdot 3^n$ do
    $c_{k+1}^i = c_k^i + (1/\Delta_k) s_k - c_k^p$.
Return.
```

Since $(1/\Delta_k) s_k$ is necessarily a column of C_k and $C_0 \in \mathbf{Z}^{n \times p}$, an argument by induction shows that the update algorithm for C_k ensures that the columns of C_k always consist of integers.

4.3.2. The exploratory moves. In Algorithm 6, the e_i's denote the unit coordinate vectors and c_k^p denotes the last column of C_k. We set $\rho_{-1} = 0$ so that ρ_{k-1} is defined when $k = 0$.

A useful example for working through the logic of the algorithm can be found in [1], though the presentation and notation differ somewhat from that given here.

ALGORITHM 6. EXPLORATORY MOVES ALGORITHM FOR HOOKE AND JEEVES.

```
Given $x_k$, $\Delta_k$, $f(x_k)$, $B$, and $\rho_{k-1}$, set $\rho_k = \rho_{k-1}$ and min $= f(x_k)$.
If $\rho_k > 0$ then set $s_k = \Delta_k B c_k^p$, $\rho_k = f(x_k) - f(x_k + s_k)$, and min $= f(x_k + s_k)$.
```

```
    For $i = 1, \ldots, n$ do
        (a) $s_k^i = s_k + \Delta_k B e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
        (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
            Otherwise,
                (i) $s_k^i = s_k - \Delta_k B e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
                (ii) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
If $\rho_k \leq 0$ then set $s_k = 0$, $\rho_k = 0$, and min $= f(x_k)$.
    For $i = 1, \ldots, n$ do
        (a) $s_k^i = s_k + \Delta_k B e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
        (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
            Otherwise,
                (i) $s_k^i = s_k - \Delta_k B e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
                (ii) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, and $s_k = s_k^i$.
Return.
```

All possible steps are contained in $\Delta_k P_k$ since C_k contains columns that represent the "pattern steps" tried at the beginning of the iteration. And, once again, the exploratory moves given in Algorithm 6 examine all $2n$ steps defined by $\Delta_k B \Gamma$ unless a step satisfying $f(x_k + s_k) < f(x_k)$ is found.

Since we have shown that the pattern search algorithm of Hooke and Jeeves satisfies all the necessary requirements, we can therefore conclude that it, too, is a special case of the generalized pattern search method and Theorem 3.5 holds.

4.4. Multidirectional search. The multidirectional search algorithm was introduced by Dennis and Torczon in 1989 [15] as a first step towards a general purpose optimization algorithm with promising properties for parallel computation. While subsequent work led to a class of algorithms (based on the multidirectional search algorithm) that allows for more flexible computation [6, 17], one of the unanticipated results of the original research was a global convergence theorem for the multidirectional search algorithm [16].

The multidirectional search algorithm is a simplex-based algorithm. The pattern of points can be expressed as a simplex (i.e., $n + 1$ points or vertices) based at the current iterate; as such, multidirectional search owes much in its conception to its predecessors, the simplex design algorithm of Spendley, Hext, and Himsworth [12] and the simplex algorithm of Nelder and Mead [9]. However, multidirectional search is a different algorithm—particularly from a theoretical standpoint. Convergence for the Spendley, Hext, and Himsworth algorithm can be shown only with some modification of the original algorithm, and then only under the additional assumption that the function f is convex. There are numerical examples to demonstrate that the Nelder–Mead simplex algorithm may fail to converge to a stationary point of the function because the uniform linear independence property (discussed in section 6.2), which plays a key role in the convergence analysis, cannot be guaranteed to hold [15].

The multidirectional search algorithm is described in detail in both [6] and [16]. The formulation given here is different and, in fact, introduces some redundancy that can be eliminated when actually implementing the algorithm. However, the way of expressing the algorithm that we use here allows us to make clear the similarities between this and other pattern search methods.

4.4.1. The matrices. It is most natural to express multidirectional search in terms of multiple basis matrices B_k and a fixed generating matrix C, which is at odds with our definition for generalized pattern search methods. As we shall see, however, it is possible to convert the more natural specification to one that conforms to our requirements for a pattern search method.

The multidirectional search algorithm centers around a family of basis matrices $\mathbf{B}$ that consists of all matrices representing the edges adjacent to each vertex in a nondegenerate n-dimensional simplex that the user is allowed to specify. Since the ordering of the columns is not unique and typically not preserved in the implementation of the method, we consider all possible representations of the columns of the matrices associated with the edges adjacent to the $(n+1)$ vertices of the simplex. We then add the negatives of these $(n+1)!$ basis matrices to account for the effect of the *reflection* step allowed by the multidirectional search algorithm. Thus the cardinality of the set $\mathbf{B}$ is $|\mathbf{B}| = 2(n+1)!$.

Fortunately, there is no need to construct this unwieldy number of basis matrices to initialize the method. We can update the basis matrix after each iteration k by reconstructing the new basis matrix B_{k+1}, given the outcome of the exploratory moves, from the trial points x_k^i, $i = 1, \ldots, n$, considered during the course of the exploratory moves. This procedure is given in Algorithm 7. The scalar *scale* is chosen during the course of the exploratory moves (see Algorithm 8) to ensure that $B_{k+1} \in \mathbf{B}$ by factoring out any change in the size of the simplex introduced by a change in Δ_k. This has the further effect of preserving the role of Δ_k as a step length parameter.

Algorithm 7. Updating B_k.

```
Given $B_k = [b_k^1 \cdots b_k^i \cdots b_k^n]$, $scale$, $best$, and $x_k^i$ for $i = 0, \ldots, n$,
If $\rho_k > 0$ then
    For $i = 0, \ldots, (best - 1)$ do
        $b_{k+1}^{i+1} = scale * (x_k^i - x_k^{best})$.
    For $i = (best + 1), \ldots, n$ do
        $b_{k+1}^{i} = scale * (x_k^i - x_k^{best})$.
Otherwise
    For $i = 1, \ldots, n$ do
        $b_{k+1}^i = b_k^i$.
Return.
```

Given this use of a family of basis matrices to help define the multidirectional search algorithm, the generating matrix is then the fixed matrix $C = [I \; -I \; -\mu I \;\; 0]$. Thus, C contains $p = 3n + 1$ columns, with $M = I$. To ensure that $C \in \mathbf{Z}^{n \times p}$, we require $\mu \in \mathbf{Z}$. Furthermore, to ensure that the role of Δ_k as a step length parameter is not lost with the introduction of the *expansion* step represented by $-\mu I$, we require $\mu \in \Lambda$. The algorithm is defined so that $\Lambda = \{\tau^{w_1}, \tau^{w_2}\}$, with $\mu = \tau^{w_2}$. This requires the further restriction that $\tau \in \mathbf{N}$. Again, this is not an onerous restriction. Multidirectional search usually is specified so that $\tau = 2$, $w_2 = 1$, and thus $\mu = 2$.

Now, to bring this notation into conformity with our definition for a generalized pattern search method, observe that we can represent all possible basis matrices $B_\nu \in \mathbf{B}$ in terms of a single reference matrix $B \in \mathbf{B}$ so that

$$B_\nu = B\hat{B}_\nu, \qquad \nu = 1, \ldots, |\mathbf{B}|.$$

A convenient feature of using the edges of a simplex to form the set of basis matrices is that the matrices $\hat{B}_\nu$ consist only of elements from the set $\{-1, 0, 1\}$. The matrices $\hat{B}_\nu$ are necessarily nonsingular because of the nondegeneracy of the simplex. We use $\hat{\mathbf{B}}$ to represent the set of matrices $\hat{B}_\nu$ and observe that since $\mathbf{B}$ is a finite set, the set $\hat{\mathbf{B}}$ is also finite.

We then observe that

$$\begin{aligned} P_k = \quad & B_k C \quad = \quad B_k \quad [I \quad -I \quad -\mu I \quad 0] \\ \equiv \quad & B \quad [\hat{B}_k \quad -\hat{B}_k \quad -\mu \hat{B}_k \quad 0] \quad = \quad BC_k. \end{aligned}$$

Thus we can define the pattern in terms of the single reference matrix B and the redefined generating matrix

$$C_k \quad \equiv \quad [\hat{B}_k \quad -\hat{B}_k \quad -\mu \hat{B}_k \quad 0],$$

with $M_k \equiv \hat{B}_k$ and $\mathbf{M} \equiv \hat{\mathbf{B}}$. We also have $L_k \equiv [-\mu \hat{B}_k \;\; 0]$ and since $\mu \in \mathbf{Z}$, $L_k \in \mathbf{Z}^{n \times (n+1)}$, as required.

4.4.2. The exploratory moves. The exploratory moves for the multidirectional search method are given in Algorithm 8; the e_i's denote the unit coordinate vectors. We use the notion of $B_k \in \mathbf{B}$ for consistency with the update algorithm given in Algorithm 6, but we could just as easily substitute $B\hat{B}_k$ for B_k in the algorithm given below.

ALGORITHM 8. EXPLORATORY MOVES ALGORITHM FOR MULTIDIRECTIONAL SEARCH.

```
Given $x_k$, $\Delta_k$, $f(x_k)$, $B_k$, and $\mu = \tau^{w_2} \in \mathbf{N}$, set $s_k = 0$, $\rho_k = 0$, min $= f(x_k)$,
$\lambda_k = 1$, $scale = 1/\Delta_k$, $best = 0$, and $x_k^0 = x_k$.
For $i = 1, \ldots, n$ do
  (a) $s_k^i = \Delta_k B_k e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
  (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, $s_k = s_k^i$, and $best = i$.
If $\rho_k \leq 0$ then
  For $i = 1, \ldots, n$ do
    (a) $s_k^i = -\Delta_k B_k e_i$ and $x_k^i = x_k + s_k^i$. Compute $f(x_k^i)$.
    (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, $s_k = s_k^i$, and $best = i$.
  If $\rho_k > 0$ then set $scale = 1/\mu\Delta_k$.
    For $i = 1, \ldots, n$ do
      (a) $s_k^i = -\mu \Delta_k B_k e_i$ and $x_k^i = x_k + s_k$. Compute $f(x_k^i)$.
      (b) If $f(x_k^i) <$ min then $\rho_k = f(x_k) - f(x_k^i)$, min $= f(x_k^i)$, $s_k = s_k^i$, $best = i$,
          and $\lambda_k = \mu$.
Return.
```

Clearly, $s_k \in \Delta_k P_k$. Since the exploratory moves algorithm considers all steps of the form $\Delta_k B \Gamma_k$, unless simple decrease is found after examining only the steps defined by $\Delta_k B M_k$, this guarantees we satisfy the condition that if $\min\{f(x_k + y), y \in \Delta_k B \Gamma_k\} < f(x_k)$, then $f(x_k + s_k) < f(x_k)$.

4.4.3. Updating the step length. The algorithm for updating Δ_k is that given in Algorithm 2. In this case, while θ usually is set to $1/2$ so that $\tau = 2$, $w_0 = -1$, and $w_1 = 0$, we also include an expansion factor $\mu = \tau^{w_2}$, where w_2 usually equals one. Thus $\Lambda = \{1, \mu\}$, where μ is usually 2. The choice of $\lambda_k \in \Lambda$ is made during the execution of the exploratory moves.

Since we have shown that the multidirectional search algorithm satisfies all the necessary requirements, we conclude that it is also a pattern search method and thus Theorem 3.5 applies. Note that since we allow $\mu > 1$, which is a useful algorithmic feature, we cannot guarantee that $\lim_{k \to +\infty} \Delta_k = 0$ and so Theorem 3.7 does not automatically apply.

5. Conclusions.

We have presented a framework in which one can analyze pattern search methods. This framework abstracts and quantifies the similarities of the classical pattern search methods and enables us to prove $\liminf_{k\to+\infty} \|\nabla f(x_k)\| = 0$ for this class of algorithms. We also specify the conditions under which the limit $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$ can be shown to hold.

These convergence results are perhaps surprising, given the simplicity of pattern search methods, but derive from the algebraic rigidity imposed on the iterates produced by pattern search methods. This is gratifying, since while this rigidity originally was introduced as a heuristic for directing the exploratory moves, it turns out to be the key to proving convergence as well. This analysis also highlights just how weak the conditions on the acceptance of the step can be and yet still allow a global convergence analysis, an observation that may prove useful in the analysis of other classes of optimization methods.

6. Technical results.

We deferred the proof of Proposition 3.4 for several reasons. First, many of the results in this section are generalizations of similar results to be found in [16]. The abstraction in section 2 leads to more succinct proofs. Second, the proof of Proposition 3.4 is closely related to that of several other results presented in this section and requires us to introduce several additional notions.

We return to our definition of the pattern as $P_k = BC_k$ to show that the pattern contains at least one direction of descent whenever $\nabla f(x_k) \neq 0$.

Recall that we require the columns of C_k to contain both M_k and $-M_k$. Thus, P_k can be partitioned as follows:

$$P_k \quad = \quad BC_k \quad = \quad B[M_k \quad -M_k \quad L_k] \quad = \quad B[\Gamma_k \quad L_k].$$

We now elaborate on these requirements. Since M_k is an $n \times n$ nonsingular matrix and B is nonsingular, we are guaranteed that BM_k forms a basis for $\mathbf{R}^n$. Further, we are guaranteed that at any iteration k, if $\nabla f(x_k) \neq 0$, $x_k - Bc_k^i$ is a direction of descent for at least one column c_k^i contained in the block Γ_k.

6.1. Descent methods.

Of course, the existence of a trial step in a descent direction is not sufficient to guarantee that decrease in the value of the objective function will be realized. To guarantee that a pattern search method is a descent method, we need to guarantee that in a finite number of iterations the method produces a positive step size Δ_k that achieves decrease on the objective function at the current iterate. We now show that this is the case.

LEMMA 6.1. *Suppose that f is continuously differentiable on a neighborhood of $L(x_0)$. If $\nabla f(x_k) \neq 0$, then there exists $q \in \mathbf{Z}$, $q \geq 0$ such that $\rho_{k+q} > 0$ (i.e., the $(k+q)$th iteration is* successful*).*

Proof. A key hypothesis placed on the exploratory moves is that if descent can be found for one of the trial steps defined by $\Delta_k B \Gamma_k$, then the exploratory moves returns a step that produces descent.

Because BC_k has rank n, if $\nabla f(x_k) \neq 0$, then there exists at least one trial direction $d_k^i = x_k - Bc_k^i$, where $c_k^i \in \Gamma_k$, such that $\nabla f(x_k)^T d_k^i \neq 0$. But, since $-c_k^i \in \Gamma_k$, $\nabla f(x_k)^T d_k^i < 0$ without loss of generality. Thus, there exists an $h_k > 0$ such that for $0 < h \leq h_k$, $f(x_k + hd_k^i) < f(x_k)$.

If at iteration k, $\Delta_k > h_k$, then the iteration may be unsuccessful; that is, $\rho_k = f(x_k) - f(x_k + s_k) \leq 0$. When the iteration is unsuccessful, the generalized pattern search method sets $x_{k+1} = x_k$ and the updating algorithm sets $\Delta_{k+1} = \theta\Delta_k$. Since θ is strictly less than one, there exists $q \in \mathbf{Z}$, $q \geq 0$, such that $\theta^q \Delta_k \leq h_k$. Thus we are guaranteed descent, i.e., a successful iteration, in at most q iterations. $\square$

6.2. Uniform linear independence. The pattern P_k guarantees the existence of at least one direction of descent whenever $\nabla f(x_k) \neq 0$. We now want to guarantee the existence of a bound on the angle between the direction of descent contained in $B\Gamma_k$ and the negative gradient at x_k (whenever $\nabla f(x_k) \neq 0$). We will show, in fact, that this bound is uniform across all iterations of the pattern search algorithm. To do so, we use the notion of *uniform linear independence* [10].

LEMMA 6.2. *For a pattern search algorithm, there exists a constant $\xi > 0$ such that for all $k \geq 0$ and $x \neq 0$,*

$$\max \left\{ \frac{|x^T(x_k^i - x_k)|}{\|x\| \|x_k^i - x_k\|}, i = 1, \ldots, p \right\} \geq \xi. \tag{13}$$

Proof. To demonstrate the existence of ξ, we first consider the simplest possible case, $B = I$ and $C = [M \ -M \ 0] = [I \ -I \ 0]$, and use this to derive a bound for any choice of B and C_k that satisfies the conditions we have imposed.

LEMMA 6.3. *Suppose $\|y\| = 1$. Define $\theta(y) \in [0, \pi/2]$ by*

$$\cos\theta(y) = \max_{1 \leq j \leq n} \left\{ |y^T e_j| \right\},$$

where the e_j's are the unit coordinate vectors.

If $B = I$ and $C = [I \ -I \ 0]$, then

$$\min_{y \in \mathbf{R}^n} \cos\theta(y) = \frac{1}{\sqrt{n}}.$$

Proof. We have $|y^T e_j| = |y_j|$, where $y = (y_1, \ldots, y_n)^T$. Since $\sum_{j=1}^n |y_j|^2 = 1$, we are guaranteed that $|y_j| \geq 1/\sqrt{n}$ for some j, so $|y^T e_j| \geq 1/\sqrt{n}$ for some j. Thus $\cos\theta(y) \geq 1/\sqrt{n}$.

Now note that $\cos\theta(y)$ attains this lower bound for any $y = \alpha_1 e_1 + \alpha_2 e_2 + \cdots + \alpha_n e_n$, where $\alpha_j = \pm 1/\sqrt{n}$. ◻

Thus, if the pattern search is restricted to the coordinate directions defined by $P = [I \ -I \ 0]$, $\xi = 1/\sqrt{n}$ gives the lower bound on the absolute value of the cosine of the angle between the gradient and a guaranteed direction of descent. We now use the bound for this particular case to derive a bound for the general case.

Assume a general basis matrix B and a general matrix $M_k \in \mathbf{M}$, where $|\mathbf{M}| < +\infty$. We adopt the notation $BM_k = [y_k^1 \cdots y_k^n]$. Then for any $x \neq 0$ we have the following:

$$|\cos\theta| = \frac{\left|x^T y_k^j\right|}{\|x\| \|y_k^j\|} = \frac{|x^T BM_k e_j|}{\|x\| \|BM_k e_j\|} = \frac{\left|\left((BM_k)^T x\right)^T e_j\right|}{\|x\| \|BM_k e_j\|}.$$

If we set $w = (BM_k)^T x$ so that $x = (BM_k)^{-T} w$, we have

$$\begin{aligned} |\cos\theta| &= \frac{|w^T e_j|}{\|(BM_k)^{-T} w\| \|BM_k e_j\|} \geq \frac{|w^T e_j|}{\|(BM_k)^{-T}\| \|w\| \|BM_k\| \|e_j\|} \\ &= \frac{1}{\|(BM_k)^{-T}\| \|BM_k\|} \left(\frac{|w^T e_j|}{\|w\| \|e_j\|} \right) = \frac{1}{\|(BM_k)^{-1}\| \|BM_k\|} \left(\frac{|w^T e_j|}{\|w\| \|e_j\|} \right) \\ &\geq \frac{1}{\kappa(BM_k)} \frac{1}{\sqrt{n}}, \end{aligned}$$

where $\kappa(BM_k)$ is the condition number of the matrix BM_k. Thus, we have

$$|\cos\theta| \geq \frac{1}{\kappa(BM_k)\sqrt{n}} > 0\,.$$

To ensure a bound ξ that is independent of the choice of any particular matrix $M \in \mathbf{M}$, we simply observe that the set $\mathbf{M}$ is required to be finite. Thus, ξ is taken to be

$$\xi = \min_{M\in\mathbf{M}} \left\{ \frac{1}{\kappa(BM)\sqrt{n}} \right\}. \qquad \square \tag{14}$$

The bound given in (14) points to two features that explain much about the behavior of pattern search methods. Since we never explicitly calculate—or approximate—the gradient, we are dependent on the fact that in the worst case at least one of our search directions is not orthogonal to the gradient; ξ gives us a bound on how far away we can be. Thus, as either the condition number of the product BM_k increases, or the dimension of the problem increases, our bound on the angle between the search direction and the gradient deteriorates. This suggests two things. First, we should be very careful in our choice of B and $\mathbf{M}$ for any particular pattern search method. Second, we should not be surprised that these methods become less effective as the dimension of the problem increases.

Nevertheless, even though pattern search methods neither require nor explicitly approximate the gradient of the function, the uniform linear independence condition demonstrates that the pattern search methods are, in fact, *gradient-related methods*, as defined by Ortega and Rheinboldt [10], which is one reason why we can establish global convergence.

6.3. The descent condition. Having introduced the notion of uniform linear independence with the bound ξ, we are now ready to show that pattern search methods reduce Δ_k only when necessary to find descent. To do this we will show that once the steps $s_k^i \equiv (x_k^i - x_k)$ are small enough, then a successful step must be returned by the exploratory moves algorithm. Lemma 3.1 allows us to restate this condition in terms of Δ_k. We use the result to prove Proposition 3.4.

PROPOSITION 6.4. *Suppose that $L(x_0)$ is compact and f is continuously differentiable on a neighborhood of $L(x_0)$. Given $\epsilon > 0$, let*

$$\Omega_\epsilon = \{x \in L(x_0) : \mathit{dist}(x, X_*) \geq \epsilon\}.$$

Suppose also that $x_0 \in \Omega_\epsilon$. Then there exists $\delta > 0$, independent of k, such that if $x_k \in \Omega_\epsilon$ and $\Delta_k < \delta$, then the kth iteration of a generalized pattern search method (see Algorithm 1) will be successful (i.e., $\rho_k = f(x_k) - f(x_k + s_k) > 0$) and thus $\Delta_{k+1} \geq \Delta_k$.

Proof. We restrict our attention to the steps defined by the columns of $\Delta_k B\Gamma_k$. This is sufficient since the Hypotheses on exploratory moves ensure that a step s_k satisfying the simple decrease condition $\rho_k > 0$ must be returned if a trial step defined by a column of $\Delta_k B\Gamma_k$ satisfies the simple decrease condition.

If s_k^i, $i = 1, \ldots, 2n$, is a step defined by $\Delta_k B\Gamma_k$ (we assume that P_k is partitioned as in (2) so that the first $2n$ columns of P_k contain the columns of $B\Gamma_k \equiv [BM_k \; -BM_k]$), then for some $\zeta^* > 0$, independent of k,

$$\|s_k^i\| \;=\; \|\Delta_k B c_k^i\| \;\leq\; \|B\|\|c_k^i\|\Delta_k \;\leq\; \zeta^*\Delta_k, \qquad i = 1, \ldots, 2n, \tag{15}$$

since $M_k \in \mathbf{M} \subset \mathbf{Z}^{n\times n}$ and $\mathbf{M}$ is a finite set of matrices. Together, (15) and Lemma 3.1 yield

$$\zeta_* \Delta_k \quad \leq \quad \|s_k^i\| \leq \quad \zeta^* \Delta_k, \qquad i = 1, \ldots, 2n.$$

Since $x_0 \in \Omega_\epsilon$, Lemma 6.1 allows us to define $N = \min\{k : x_k \neq x_0\}$. Define $d = \operatorname{dist}(L(x_N), C(x_0))$. Because $L(x_N)$ and $C(x_0)$ are compact and disjoint, we know that $d > 0$. If $\Delta_k < d/2\zeta^*$, then $\|s_k^i\| \leq \zeta^* \Delta_k < d/2$ for all $i = 1, \ldots, 2n$. Thus x_k^i lies in the interior of $L(x_0)$ for all $i = 1, \ldots, 2n$. More precisely, for all $i = 1, \ldots, 2n$, x_k^i lies in the ball $B(x_k, d/2) \subset L(x_0)$.

Let $\alpha = \min_{x\in\Omega_\epsilon} \|\nabla f(x)\|$. By design, $\alpha > 0$. Since ∇f is continuous on a neighborhood of $L(x_0)$, ∇f is uniformly continuous on a neighborhood of $L(x_0)$. Thus, there exists a constant $r > 0$, depending only on α and the ξ from (13), such that

$$\|\nabla f(x) - \nabla f(x_k)\| \quad \leq \quad \tfrac{\xi\alpha}{2} \qquad \text{whenever} \qquad \|x - x_k\| \quad \leq \quad r \qquad (\text{and } x \in L(x_0)).$$

We define

$$\delta = \frac{1}{\zeta^*} \min\left\{\frac{d}{2},\ r\right\}. \tag{16}$$

We are now assured that if

$$\Delta_k < \delta \tag{17}$$

then

$$x_k^i \in B\left(x_k, \frac{d}{2}\right) \subset L(x_0),\ i = 1, \ldots, 2n, \tag{18}$$

and

$$\|\nabla f(x_k^i) - \nabla f(x_k)\| \leq \tfrac{\xi\alpha}{2},\ i = 1, \ldots, 2n. \tag{19}$$

We are ready to argue that if at any iteration $k \geq N$, $x_k \in \Omega_\epsilon$ and (17) is satisfied, then an acceptable step will be found.

Choose a trial point x_k^i, $i = 1, \ldots, 2n$, that satisfies both $\nabla f(x_k)^T(x_k^i - x_k) < 0$ and

$$\frac{|\nabla f(x_k)^T(x_k^i - x_k)|}{\|\nabla f(x_k)\| \|x_k^i - x_k\|} \geq \xi.$$

The definitions of Ω_ϵ and the pattern P_k, together with Lemma 6.2, guarantee the existence of at least one such x_k^i.

Since (17) holds by assumption, (18) also holds. We can apply the mean value theorem to obtain $f(x_k^i) - f(x_k) = \nabla f(\omega)^T(x_k^i - x_k)$ for some $\omega \in (x_k, x_k^i)$, where

$$f(x_k^i) - f(x_k) = \nabla f(x_k)^T(x_k^i - x_k) + (\nabla f(\omega) - \nabla f(x_k))^T(x_k^i - x_k). \tag{20}$$

Consider the first term on the right-hand side of (20). Our choice of x_k^i gives us

$$\left|\nabla f(x_k)^T(x_k^i - x_k)\right| \geq \xi \|\nabla f(x_k)\| \|x_k^i - x_k\|.$$

Furthermore, since $\nabla f(x_k)^T(x_k^i - x_k) < 0$, we have

$$\nabla f(x_k)^T(x_k^i - x_k) \leq -\xi \|\nabla f(x_k)\| \|x_k^i - x_k\|. \tag{21}$$

Now consider the second term on the right-hand side of (20). The Cauchy–Schwarz inequality gives us

$$\left|(\nabla f(\omega) - \nabla f(x_k))^T(x_k^i - x_k)\right| \leq \|\nabla f(\omega) - \nabla f(x_k)\| \|x_k^i - x_k\|. \tag{22}$$

Combine (21) and (22) to rewrite (20) as

$$\begin{aligned} f(x_k^i) - f(x_k) &\leq -\xi \|\nabla f(x_k)\| \|x_k^i - x_k\| + \|\nabla f(\omega) - \nabla f(x_k)\| \|x_k^i - x_k\| \\ &= (-\xi \|\nabla f(x_k)\| + \|\nabla f(\omega) - \nabla f(x_k)\|) \|x_k^i - x_k\|. \end{aligned}$$

Since $\omega \in (x_k, x_k^i)$ and (17) holds by assumption, (19) also holds. We then have

$$f(x_k^i) - f(x_k) \leq (-\xi \|\nabla f(x_k)\| + \frac{\xi}{2} \|\nabla f(x_k)\|) \|x_k^i - x_k\| < 0. \tag{23}$$

Thus, when $\Delta_k < \delta$, $f(x_k^i) \equiv f(x_k + s_k^i) < f(x_k)$ for at least one s_k^i defined by $\Delta_k B c_k^i$, $i = 1, \ldots, 2n$. The Hypotheses on exploratory moves guarantee that if $\min\{f(x_k + y), y \in \Delta_k B \Gamma_k\} < f(x_k)$, then $f(x_k + s_k) < f(x_k)$. Thus, $\rho_k = f(x_k) - f(x_k + s_k) > 0$ and the algorithm for updating Δ_k (Algorithm 2) ensures that $\Delta_{k+1} \geq \Delta_k$. ∎

Proposition 6.4 guarantees that if Δ_k is small enough, a generalized pattern search method realizes simple decrease because there exists at least one step among the $2n$ steps defined by $\Delta_k B \Gamma_k$ that gives decrease as a function of the norm of the gradient at the current iterate, as shown in (23); the Hypotheses on exploratory moves then ensure that the exploratory moves algorithm must return a step that satisfies at least simple decrease. However, there are no guarantees that the step returned by an exploratory moves algorithm satisfies more than the simple decrease condition.

To tie the amount of actual decrease to the norm of the gradient, we must place much stronger conditions on the generalized pattern search method, as discussed in section 3.3.2. Once we have done so, Corollary 6.5 follows more or less immediately from Proposition 6.4.

COROLLARY 6.5. *Suppose that $L(x_0)$ is compact and f is continuously differentiable on a neighborhood of $L(x_0)$. Suppose that the columns of the generating matrix are bounded in norm and that the generalized pattern search method (Algorithm 1) enforces the Strong hypotheses on exploratory moves. Given $\epsilon > 0$, let*

$$\Omega_\epsilon = \{x \in L(x_0) : dist(x, X_*) \geq \epsilon\}.$$

Suppose also that $x_0 \in \Omega_\epsilon$. Then there exist $\delta > 0$ and $\sigma > 0$, independent of k, such that for all but finitely many k, if $x_k \in \Omega_\epsilon$ and $\Delta_k < \delta$, then

$$f(x_{k+1}) \quad \leq \quad f(x_k) - \sigma \|\nabla f(x_k)\| \|s_k\| \quad < \quad f(x_k).$$

Proof. From Proposition 6.4, (23) says that for $k \geq N = \min\{k : x_k \neq x_0\}$ (Lemma 6.1 guarantees the existence of N), there exists at least one trial step $s_k^i \in \Delta_k B \Gamma_k$ such that once $\Delta_k < \delta$, where δ is as defined in (16), we have

$$f(x_k^i) \quad \leq \quad f(x_k) - \tfrac{\xi}{2} \|\nabla f(x_k)\| \|s_k^i\| \quad < \quad f(x_k).$$

The Strong hypotheses on exploratory moves give us

$$f(x_{k+1}) \quad \leq \quad f(x_k) - \tfrac{\xi}{2}\|\nabla f(x_k)\|\|s_k^i\| \quad < \quad f(x_k).$$

Lemma 3.1 ensures that

$$f(x_{k+1}) \quad \leq \quad f(x_k) - \tfrac{\xi}{2}\zeta_* \Delta_k \|\nabla f(x_k)\| \quad < \quad f(x_k).$$

Lemma 3.6, which holds only when the columns of the generating matrix are bounded in norm, gives us

$$f(x_{k+1}) \quad \leq \quad f(x_k) - \tfrac{\xi}{2}\zeta_* \psi_* \|\nabla f(x_k)\|\|s_k\| \quad < \quad f(x_k).$$

We define $\sigma = \frac{\xi}{2}\zeta_*\psi_*$ to complete the proof. □

We now prove Proposition 3.4.

Proof. By assumption, $\liminf_{k\to+\infty} \|\nabla f(x_k)\| \neq 0$. Then we can find N_1 and $\epsilon > 0$ such that for all $k \geq N_1$, $x_k \in \Omega_\epsilon = \{x \in L(x_0) : \text{dist}(x, X_*) \geq \epsilon\}$. Lemma 6.1 guarantees the existence of $N_2 = \min\{k : x_k \neq x_0\}$. Let $N = \max(N_1, N_2)$.

From Proposition 6.4 we are assured of $\delta > 0$ such that if $\Delta_k \leq \delta$, then the iteration will be successful. Given Δ_0, there exists a constant $q \in \mathbf{Z}$, $q \geq 0$, such that $\theta^q \Delta_0 \leq \delta$, where $\theta \in (0,1)$ and is as defined in the algorithm for updating Δ_k (Algorithm 2). Thus, for $k \geq N$, $\theta^{q+1}\Delta_0 < \Delta_k$.

Set $\Delta_{LB} = \theta \min(\theta^q \Delta_0,\ \Delta_1, \ldots,\ \Delta_{N-1})$. Then for all k, $\Delta_{LB} < \Delta_k$. □

Acknowledgments. This paper benefited from conversations with J. E. Dennis, Stephen Nash, Michael Trosset, Luís Vicente, and especially Michael Lewis. In particular, discussions with Michael Lewis were critical in the distillation of the abstraction for pattern search methods found in section 2 and in the development of the analytic arguments for the algebraic structure of the iterates found in section 3.1.

The review and comments made by an anonymous referee, Danny Ralph (the second referee, who agreed to reveal his identity for this acknowledgment), and Jorge Moré, the editor, are gratefully acknowledged. In particular, the observation by Danny Ralph that the pattern contains only vectors in a fixed lattice led to a more general result and a much more elegant presentation.

REFERENCES

[1] M. Avriel, *Nonlinear Programming: Analysis and Methods*, Prentice–Hall, Englewood Cliffs, NJ, 1976.

[2] G. E. P. Box, *Evolutionary operation: A method for increasing industrial productivity*, Appl. Statist., 6 (1957), pp. 81–101.

[3] M. J. Box, D. Davies, and W. H. Swann, *Non-Linear Optimization Techniques*, ICI Monograph No. 5, Oliver & Boyd, Edinburgh, 1969.

[4] J. Céa, *Optimisation: Théorie et algorithmes*, Dunod, Paris, 1971.

[5] W. C. Davidon, *Variable metric method for minimization*, SIAM J. Optim., 1 (1991), pp. 1–17. Originally published without the preface as Argonne National Laboratory Research and Development Report 5990, May, 1959.

[6] J. E. Dennis, Jr. and V. Torczon, *Direct search methods on parallel machines*, SIAM J. Optim., 1 (1991), pp. 448–474.

[7] R. Hooke and T. A. Jeeves, *"Direct search" solution of numerical and statistical problems*, J. Assoc. Comput. Mach., 8 (1961), pp. 212–229.

[8] J. J. Moré, *Recent developments in algorithms and software for trust region methods*, in Math. Programming, The State of the Art, A. Bachem, M. Grötschel, and G. Korte, eds., Springer-Verlag, Berlin, New York, 1983, pp. 256–287.

[9] J. A. Nelder and R. Mead, *A simplex method for function minimization*, Comput. J., 7 (1965), pp. 308–313.

[10] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[11] E. Polak, *Computational Methods in Optimization: A Unified Approach*, Academic Press, New York, 1971.

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

[13] W. H. Swann, *Direct search methods*, in Numerical Methods for Unconstrained Optimization, W. Murray, ed., Academic Press, New York, 1972, pp. 13–28.

[14] S. W. Thomas, *Sequential Estimation Techniques for Quasi-Newton Algorithms*, Ph.D. thesis, Cornell University, Ithaca, NY, 1975.

[15] V. Torczon, *Multi-Directional Search: A Direct Search Algorithm for Parallel Machines*, Ph.D. thesis, Department of Mathematical Sciences, Rice University, Houston, TX, 1989.

[16] V. Torczon, *On the convergence of the multidirectional search algorithm*, SIAM J. Optim., 1 (1991), pp. 123–145.

[17] V. Torczon, *PDS: Direct Search Methods for Unconstrained Optimization on Either Sequential or Parallel Machines*, Tech. Report 92–9, Department of Mathematical Sciences, Rice University, Houston, TX, 1992.

[18] Y. Wen-ci, *Positive basis and a class of direct search techniques*, Scientia Sinica, Special Issue of Mathematics, 1 (1979), pp. 53–67.