Abstract Models for the Synthesis Of Optimization Algorithms

by

G. Meyer and E. Polak

Memorandum No. ERL-268

October 1969

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Introduction.

The convergence of optimization algorithms has been studied extensively in recent years (see [4], [5], [6], [7], [8], [9], [10]). The approach generally adopted in these studies consisted of defining a class of algorithms and then giving convergence theorems which applied to every algorithm in this class. This approach has resulted in the development of general procedures which have considerably simplified the task of establishing whether an algorithm is convergent. However, the emphasis so far has been on analysis. Very few of the existing results provide guidelines for the synthesis of algorithms.

In this paper we present a systematic approach to the problem of synthesis of optimization algorithms. The development of an algorithm usually evolves through three phases. The first is a heuristic, or invention phase in which intuition plays an extremely important part. In the second phase one transforms one's intuitive ideas into a "conceptual" algorithm, i.e. an algorithm which may consist of operations that are inadmissible in a practical method. (For example, a conceptual algorithm may require us to find the limit of an infinite sequence at each iteration). The last phase consists of converting the "conceptual" algorithm into an "implementable" algorithm. Our approach to the problem of synthesis consists of two parts. First we develop abstract models for algorithms. These models guide the inventive process towards "conceptual" algorithms which will later be easily made implementable. Once the abstract models are established, we present a set of methods for converting "conceptual" algorithms,

falling into the class defined by the abstract models, into "implementable" iterative procedures.

One of the most frequently occurring difficulties in the implementation of a conceptual algorithm is the requirement that an implicit relation be solved at each iteration, e.g. minimize a function along a line, maximize a linear function in a convex set, etc. Generally, we only have methods for constructing a sequence whose limit point satisfies such an implicit relation. Now it is well known that no limit point of an infinite sequence can be determined on a digital computer in a finite time. Consequently, the task most frequently encountered in the design of a transition from a "conceptual" algorithm to an "implementable" one is that of finding methods for avoiding the need to construct limit points.

In this paper, we propose two methods for obviating the need for constructing infinite sequences in subprocedures. The first is a truncation procedure and is presented in Section II, the second is an ε - approximation procedure which is presented in Section IV.

The scope to which a paper must be held does not permit us to illustrate the applicability of the ideas presented copiously. It is our hope, however, that the two examples given in Section III will convince the reader of the great usefulness of the approach described.

I. Abstract Models for a Class of Iterative Procedures.

Throughout this paper, we shall assume that we are given a closed and bounded subset T of R^n in which we wish to find points with a specific property π. We shall call points in T with the property π desirable.

The simplest algorithms for finding desirable points in T are composed of a map $\xi(\cdot)$ from T into R, of a map $A(\cdot)$ from T into all the subsets of T and have the following form.

Algorithm 1.

Step 0: Compute a point z_0 in T and set i = 0.

Step 1: Compute a point z_{i+1} in $A(z_i)$.

Step 2: If $\xi(z_{i+1}) < \xi(z_i)$ set i = i+1 and go to Step 1, otherwise stop.

Definition 1. We shall say that an iterative procedure of the form of Algorithm 1 is convergent if any sequence of points $\{z_i\}$ it generates satisfies one of the following conditions:

(i) If the sequence $\{z_i\}$ is finite i.e. $\{z_i\} = \{z_0, z_1, \ldots z_k\}$ then z_{k-1} is desirable;

(ii) If the sequence $\{z_i\}$ is infinite then any of its cluster points is desirable.

It is quite easy to show that Algorithm 1 is convergent under the following assumption (see [10]).

Hypothesis 1. [Zangwill]

(i) z in T is desirable if there exists at least one point a in A(z) such that $\xi(a) - \xi(z) \geq 0$;

(ii) $\xi(\cdot)$ is continuous on T;

(iii) $A(\cdot)$ is closed.[+]

Hypothesis 1 is not the only assumption which ensures that Algorithm 1 is convergent. We now state another assumption which ensures that Algorithm 1 is convergent (see [9]).

Hypothesis 2. [Polak]

(i) z in T is desirable if there exists at least one point a in $A(z)$ such that $\xi(a) - \xi(z) \geq 0$;

(ii) $\xi(\cdot)$ is either continuous on T or else $\xi(z)$ is bounded from below on T;

(iii) If z in T satisfies $\xi(a) - \xi(z) < 0$ for all a in $A(z)$, then there exist $\varepsilon > 0$ and $\delta > 0$ such that $\xi(a') - \xi(z') \leq -\delta < 0$ for all z' in T such that $|| z' - z || \leq \varepsilon$, for all a' in $A(z')$.

Remark. It can be shown that Hypothesis 2 is weaker than Hypothesis 1 (see [1]).

In this paper we differentiate between explicit and implicit algorithms. This differentiation is largely heuristic but is extremely important in the construction of computationally efficient algorithms.

By an explicit algorithm we shall mean an algorithm of the form of

---

[+]Note: Let $A(\cdot)$ be a map from T into all the subsets of T. If for any sequence $\{y_i\}$ converging to y^* and for any sequence $\{\alpha_i\}$ converging to α^*, with α_i in $A(y_i)$, α^* belongs to $A(y^*)$, then we say that the map $A(\cdot)$ is closed.

Algorithm 1 in which the map $A(\cdot)$ is a point to point map, i.e. $A(\cdot)$ maps T into T. We use the word explicit to indicate that the computation of $A(z)$ for z in T can be carried out in a reasonably straightforward manner.

Explicit algorithms do not lead to computational difficulties and therefore we shall say no more about them. Implicit algorithms on the other hand cannot be readily implemented on a digital computer, as we shall shortly show, and must therefore be regarded as "conceptual" rather than as "practical" algorithms. The following sections of this paper will be devoted to developing methods for modifying convergent implicit algorithms in such a way as to produce convergent algorithms in explicit form.

We shall consider two abstract models of implicit algorithms. The first one, which is defined below, uses a map $A(\cdot)$ such that to compute a point z in $A(z)$, we must solve an implicit equation.

Definition 2. Let $U(\cdot)$ be a map from T into all the subsets of T and let $\alpha(\cdot)$ and $\gamma(\cdot)$ be maps from T into R^1. Then for every z in T, we define the set $A(z)$ as consisting of all y in $U(z)$ such that $\gamma(y) = \alpha(z)$.

The following assumption ensures that the set $A(z)$ is not empty.

Hypothesis 3. Given any point z in T, there exists a point y in $U(z)$ such that $\alpha(z) = \gamma(y)$.

In this case, Algorithm 1 takes on the following expanded form.

Algorithm 2.

Step 0: Compute a point z_0 in T and set i = 0.

Step 1: Compute a point z_{i+1} in $U(z_i)$ satisfying $\gamma(z_{i+1}) = \alpha(z_i)$.

Step 2: If $\xi(z_{i+1}) < \xi(z_i)$, set i = i+1 and go to Step 1, otherwise stop.

Hypothesis 4.

(i) A point z in T is desirable if there exists at least one point y in U(z) such that $\alpha(z) = \gamma(y)$ and $\xi(y) - \xi(z) \geq 0$;

(ii) The maps $\alpha(\cdot)$, $\gamma(\cdot)$ and $\xi(\cdot)$ are continuous on T;

(iii) The map $U(\cdot)$ is closed.

The proof of the following proposition is easy and has been omitted.

Proposition 1. If the maps $\alpha(\cdot)$, $\gamma(\cdot)$, $\xi(\cdot)$, and $U(\cdot)$ satisfy Hypotheses 3 and 4, then the map $A(\cdot)$ given by Definition 2 is closed and Algorithm 2 is convergent.

We now present the second specific form of the map $A(\cdot)$ that we wish to consider. This form is characterized by the fact that to find points in A(z) we must compute intermediate points.

Definition 3. Let U be a closed and bounded subset of R^n, let $\beta(\cdot,\cdot)$ be a map from T x U into R, let $b(\cdot,\cdot)$ be a map from T x U into T and let $\alpha(\cdot)$ be a map from T into R. Then for every z in T, we define the set A(z) as follows:

$$A(z) = \{y | y = b(z, w) \ , \ w \in U \text{ such that } \beta(z,w) = \alpha(z)\}.$$

The following assumption ensures that for every z in T the set A(z) is non empty.

Hypothesis 5. Given any z in T, there exists a point w in U such that $\beta(z,w) = \alpha(z)$.

With the map A(·) defined as above, Algorithm 1 expands as follows.

Algorithm 3.

Step 0: Compute a point z_0 in T and set i = 0.

Step 1: Compute a point w_i in U satisfying $\alpha(z_i) = \beta(z_i, w_i)$.

Step 2: Set $z_{i+1} = b(z_i, w_i)$.

Step 3: If $\xi(z_{i+1}) < \xi(z_i)$ let i = i + 1 and go to Step 1, otherwise stop.

Hypothesis 6.

(i) A point z in T is desirable if there exists at least one point w in U satisfying $\alpha(z) = \beta(z,w)$ such that $\xi(b(z,w)) - \xi(z) \geq 0$;

(ii) The maps $\alpha(\cdot)$ and $\xi(\cdot)$ are continuous on T;

(iii) The maps $\beta(\cdot,\cdot)$ and $b(\cdot,\cdot)$ are jointly continuous on T x U.

Proposition 2. If the maps $\alpha(\cdot)$, $\beta(\cdot,\cdot)$, $b(\cdot,\cdot)$ and $\xi(\cdot)$ satisfy Hypotheses 5 and 6, then the map A(·) given by Definition 3 is closed and Algorithm 3 is convergent.

The transformation of an implicit algorithm into an explicit one usually depends upon the specific conceptual method one has in mind for finding points in the set $A(z)$. We shall now introduce two infinite subprocedures for calculating points in $A(z)$ and in the next section we shall show how these subprocedures can be truncated to produce convergent explicit algorithms.

We begin by considering a subprocedure for computing points in the set $A(z)$ when the map $A(\cdot)$ is defined as in Definition 2 (we suppose, of course that we are unaware of a more straightforward method for calculating points in $A(z)$). Let us denote by N the set of positive integers and suppose that we have a mapping $m(\cdot,\cdot,\cdot)$ from $T \times T \times N$ into T which satisfies the following assumption.

Hypothesis 7.

(i) $m(z,y,j)$ belongs to $U(z)$ for all z in T, y in $U(z)$ and j in N;

(ii) The sequence $\{\gamma(m(z,y,j))\}_{j=0}^{\infty}$ converges to $\alpha(z)$ for all z in T and y in $U(z)$.

Clearly, for any z in T and y in $U(z)$, every cluster point of the sequence $\{m(z,y,j)\}_{j=0}^{\infty}$ is in $A(z)$ given by Definition 2. When the map $m(\cdot,\cdot,\cdot)$ is introduced, Algorithm 2 assumes the following specific form.

Algorithm 4.

Step 0: Compute a point z_0 in T and set i = 0.

Step 1: Compute a point y_i in $U(z_i)$ and let z_{i+1} be any cluster point of the sequence $\{m(z_i, y_i, j)\}_{j=0}^{\infty}$.

Step 2: If $\xi(z_{i+1}) < \xi(z_i)$, set i = i + 1 and go to Step 1, otherwise stop.

In view of Proposition 1, the following proposition is obvious.

Proposition 3. If the maps $\alpha(\cdot)$, $\gamma(\cdot)$, $\xi(\cdot)$, $U(\cdot)$ satisfy Hypotheses 3 and 4, the map $m(\cdot,\cdot,\cdot)$satisfies Hypothesis 7, Algorithm 4 is convergent.

Remark. Obviously, Algorithm 4 cannot be implemented on a digital computer since it would inevitably jam up in Step 1.

We now consider a subprocedure for computing points in A(z) when the map $A(\cdot)$ is defined as in Definition 3. Thus, suppose that we have a map $m(\cdot,\cdot,\cdot)$ from T x U x N into T which satisfies the following assumption.

Hypothesis 8.

(i) m(z,y,j) belongs to U for all z in T, y in U and j in N;

(ii) The sequence $\{\beta(z,m(z,y,j))\}_{j=0}^{\infty}$ converges to $\alpha(z)$ for all z in T and y in U;

When such a map is introduced, Algorithm 3 assumes the following specific form.

Algorithm 5.

Step 0: Compute a point z_0 in T and set $i = 0$.

Step 1: Compute a point y_i in U and let w_i be any cluster point of the sequence $\{m(z_i, y_i, j)\}_{j=0}^{\infty}$.

Step 2: Set $z_{i+1} = b(z_i, w_i)$.

Step 3: If $\xi(z_{i+1}) < \xi(z_i)$ let $i = i + 1$ and go to Step 0 otherwise stop.

In view of Proposition 2, the following proposition is obvious.

Proposition 4. If the maps $\alpha(\cdot)$, $\beta(\cdot,\cdot)$, $\xi(\cdot)$ and $b(\cdot,\cdot)$ satisfy Hypotheses 5 and 6 and the map $m(\cdot,\cdot,\cdot)$ satisfies Hypothesis 8, Algorithm 5 is convergent.

Again it is clear that if implemented on a digital computer, Algorithm 5 would inevitably jam up in Step 1.

II. Truncation Methods.

As we have just seen, Algorithms 4 and 5 will inevitably jam up in Step 1 since it is impossible to compute cluster points of infinite sequences in a finite time by means of a digital computer. Even if one relies on the finite word length of a digital computer to stop calculations after a finite time, this finite time will usually be prohib-

itively long. Consequently some sort of truncation procedure must be used in converting these algorithms into a more realistic form.

We begin by defining a class of maps from N into N (the set of all positive integers) which will be called truncation functions.

Definition 4. We shall say that a map $\ell(\cdot)$ from N into N is a truncation function if given any m in N, there exists a k in N such that

$$\ell(i) \geq m \quad \text{for all } i \geq k, \quad i \text{ in } N.$$

We first use truncation functions in Algorithm 4 which then takes on the following form.

Algorithm 6. Let $\ell(\cdot)$ be a given truncation function.

Step 0: Compute a point z_0 in T and set $i = 0$.

Step 1: Compute a point y_i in $U(z_i)$ and set $z_{i+1} = m(z_i, y_i, \ell(i))$.

Step 2: If $\xi(z_{i+1}) < \xi(z_i)$, set $i = i + 1$ and go to Step 1, otherwise stop.

Remark. In Step 1 of Algorithm 6, it is required that a point y_i be found in $U(z_i)$. We suppose that this task is easy. In fact, in almost all applications, z belongs to U(z) and a natural choice for y_i in $U(z_i)$ consists in letting $y_i = z_i$. In order to ensure that cluster

points of infinite sequences generated by Algorithm 6 are desirable the following assumption must be made.

Hypothesis 9.

(i) The sequence $\{\gamma(m(z,y,j))\}_{j=0}^{\infty}$ is monotonocally decreasing for all z in T and y in U(z);

(ii) Given any z in T, y in U(z), j in N and $\delta > 0$, there exists an $\varepsilon > 0$, possibly depending on z, y, j and δ, such that $\gamma(m(z',y',j)) - \gamma(m(z,y,j)) \leq \delta$, for all z' in T such that $||z'-z|| \leq \varepsilon$, for all y' in U(z') such that $||y' - y|| \leq \varepsilon$.

Proposition 5. If the maps $\alpha(\cdot)$, $\gamma(\cdot)$, $\xi(\cdot)$, $U(\cdot)$ satisfy Hypotheses 3 and 4, the map $m(\cdot,\cdot,\cdot)$ satisfies Hypothesis 7 and 9 and the map $\ell(\cdot)$ is a truncation function, then every cluster point of an infinite sequence generated by Algorithm 6 is desirable.

Proof. Consider an infinite sequence $\{z_i\}$ generated by Algorithm 6 and let z^* be a cluster point of this sequence i.e. let K_2 be a subset of the integers such that the subsequence $\{z_i\}_{K_2}$ converges to z^*. Consider the sequences $\{z_{i+1}\}_{K_2}$ and $\{y_i\}_{K_2}$ in T. The boundedness of T implies that there exists K_1 a subset of K_2 such that the subsequence $\{z_{i+1}\}_{K_1}$ converges to z^{**} and the subsequence $\{y_i\}_{K_1}$ converges to y^*. The property of convergent sequences implies that the subsequence $\{z_i\}_{K_1}$ also converges to z^*.

In order to show that z^* is desirable it is enough to establish the three following facts:

(i) z^{**} belongs to $U(z^*)$

(ii) $\alpha(z^*) = \gamma(z^{**})$

(iii) $\xi(z^{**}) - \xi(z^*) \geq 0$

By construction $z_{i+1} = m(z_i, y_i, \ell(i))$ and y_i is in $U(z_i)$. It now follows from (i) of Hypothesis 7 that z_{i+1} is in $U(z_i)$. Next, since the map $U(\cdot)$ is closed, we must have z^{**} and y^* in $U(z^*)$.

Let $\delta > 0$, then part (ii) of Hypothesis 7 implies that there exists k_2 in N such that $\gamma(m(z^*,y^*,k_2)) - \delta/2 \leq \alpha(z^*)$. From Hypothesis 9 it follows that there exists an $\varepsilon > 0$ such that $\gamma(m(z',y',k_2)) - \gamma(m(z^*,y^*,k_2)) \leq \delta/2$ for all z' in T such that $||z' - z^*|| \leq \varepsilon$, for all y' in $U(z')$ such that $||y' - y^*|| \leq \varepsilon$. Therefore there exists k_1 in N such that $\gamma(m(z_i,y_i,k_2)) - \delta \leq \alpha(z^*)$ for all $i \geq k_1$, i in K_1. Since the map $\ell(\cdot)$ is a truncation function, there exists a k_0 in N such that $\ell(i) \geq k_2$ for all $i \geq k_0$. It follows from part (i) of Hypothesis 9 that $\gamma(m(z_i,y_i,\ell(i))) - \delta \leq \alpha(z^*)$ for all $i \geq k$, i in K where $k = \max(k_0,k_1)$. But $\gamma(\cdot)$ is continuous and therefore $\gamma(z^{**}) - \delta \leq \alpha(z^*)$. Since this is true for any $\delta > 0$, it follows that $\gamma(z^{**}) \leq \alpha(z^*)$.

Now given any z_i and y_i in $U(z_i)$, part (ii) of Hypothesis 9 implies that $\gamma(m(z_i,y_i,j)) \geq \alpha(z_i)$ for all j in N i.e. $\gamma(m(z_i,y_i,\ell(i))) \geq \alpha(z_i)$. It follows that $\gamma(z^{**}) \geq \alpha(z^*)$ and therefore $\gamma(z^{**}) = \alpha(z^*)$.

Now suppose that $\xi(z^{**}) - \xi(z^*) < 0$. The continuity of $\xi(\cdot)$ implies

that there exist a $\delta > 0$ and an $\varepsilon > 0$ such that

$$\xi(z'') - \xi(z') \leq -\delta < 0$$

for all z'' in T such that $||z'' - z^{**}|| \leq \varepsilon$, for all z' in T such that $||z' - z^{*}|| \leq \varepsilon$. It follows that there exists a k such that $\xi(z_{i+1}) - \xi(z_i) \leq -\delta$ for all $i \geq k$, i in K_1. The sequence $\{\xi(z_i)\}$ is monotonically decreasing, the set T is bounded and therefore $\{\xi(z_i)\}$ converges to ξ^* which contradicts the fact that $\xi(z_{i+1}) - \xi(z_i) \leq -\delta$ for all $i \geq k$, i in K_1. It follows that $\xi(z^{**}) - \xi(z^{*}) \geq 0$ and z^* is desirable.

Note that as stated, Algorithm 6 is not necessarily convergent because it may stop in Step 2 at a point z_i which may be not desirable. To make it convergent we only need to remove the stop condition in Step 2. However, in many practical instances, as we shall see in the examples in Section III, the stop condition in Step 2 detects desirable points only, and therefore we have included it in the statement of Algorithm 6.

We now use truncation functions in Algorithm 5 to construct the following form.

Algorithm 7. Let $\ell(\cdot)$ be a given truncation function.

Step 0: Compute a point z_0 in T and set $i = 0$.

Step 1: Compute a point y_i in U and set $w_i = m(z_i, y_i, \ell(i))$

Step 2: Set $z_{i+1} = b(z_i, w_i)$.

Step 3: If $\xi(z_{i+1}) < \xi(z_i)$, let $i = i + 1$ and go to Step 1, otherwise stop.

Remark. In Step 1 of Algorithm 7, it is required that a point y_i be found in the set U. We suppose that this task is easy. In fact, in almost all applications, z belongs to U and a natural choice for y_i consists in letting $y_i = z_i$.

In order to ensure that cluster points of infinite sequences generated by Algorithm 7 are desirable, the following assumption must be made.

Hypothesis 10.

(i) The sequence $\{\beta(z,m(z,y,j))\}_{j=0}^{\infty}$ is monotonically decreasing for all z in T and y in U;

(ii) Given any z in T, y in U, j in N and $\delta > 0$, there exists an $\varepsilon > 0$, possibly depending on z, y, j and δ, such that $\beta(z', m(z',y',j))-\beta(z,m(z,y,j)) \leq \delta$ for all z' in T such that $||z' - z|| \leq \varepsilon$ and for all y' in U such that $||y' - y|| \leq \varepsilon$.

The following proposition can be proved easily, following the same type of argument as in the proof of Proposition 5.

Proposition 6. If the maps $\alpha(\cdot)$, $\beta(\cdot,\cdot)$, $\xi(\cdot)$, $b(\cdot,\cdot)$ satisfy Hypotheses 5 and 6, the map $m(\cdot,\cdot,\cdot)$ satisfies Hypotheses 8 and 10 and the map $\ell(\cdot)$ is a truncation function, then every cluster point of an infinite sequence generated by Algorithm 7 is desirable.

As in the case of Algorithm 6, Algorithm 7 as stated is not necessarily convergent because of the stop condition in Step 2. This

stop condition is included for the same reasons we gave for including it in Algorithm 6.

III. Applications

In order to clarify the concepts and methods exposed in the preceding sections we are now going to examine two specific problems and the algorithms usually used to solve them.

III-1. Unconstrained Minimization Problems.

In this subsection we shall examine the following classical problem.

Problem 1. Find a $\hat{z}$ in R^n such that

$$f(\hat{z}) \leq f(z) \quad \text{for all } z \text{ in } R^n$$

where $f(\cdot)$, a convex map from R^n into R^1 is continuously differentiable with the property that the set $\{z \mid f(z) \leq \alpha\}$ is bounded for every α in R^1.

Suppose that we have a point z_0 in R^n. We define T as

$$T=\{y \in R^n \mid f(y) \leq f(z_0)\}$$

In this case a point $\hat{z}$ in T is desirable iff it minimizes $f(z)$ over T, i.e. $f(z) \leq f(\hat{z})$ for all z in T. Since $f(\cdot)$ is convex and continuously differentiable, we recognize z to be desirable iff $\nabla f(\hat{z}) = 0$.

We propose to use the Steepest Descent Method for solving Problem 1, i.e. the following algorithm.

Algorithm 8.

Step 0: Let z_0 be a point in R^n and let $i = 0$.

Step 1: Let z_{i+1} be any point satisfying

(i) z_{i+1} belongs to $U(z_i)$;

(ii) $f(z_{i+1}) = \min \{f(y) \mid y \in U(z_i)\}$; where $U(z_i) = \{y \in R^n \mid y = z_i + \nu \nabla f(z_i), \nu \in [-1,0]\}$.

Step 2: If $f(z_{i+1}) < f(z_i)$ set $i = i + 1$ and go to Step 1, otherwise stop.

Algorithm 8 can be seen to be of the form of Algorithm 2 with the maps $\alpha(\cdot)$, $\gamma(\cdot)$, $\xi(\cdot)$, $U(\cdot)$ in Algorithm 2 defined as follows.

Definition 5. Let the maps $\alpha(\cdot)$, $\gamma(\cdot)$, $\xi(\cdot)$ from T into R^1 and the map $U(\cdot)$ from T into all the subsets of T be defined as:

(i) $U(z) = \{y \in T \mid y = z + \nu \nabla f(z), \quad \nu \in [-1,0]\}$.

(ii) $\alpha(z) = \min \{f(y) \mid y \in U(z)\}$;

(iii) $\gamma(z) = f(z)$;

(iv) $\xi(z) = f(z)$.

Since by inspection the maps $\alpha(\cdot)$, $\gamma(\cdot)$, $\xi(\cdot)$ and $U(\cdot)$ satisfy Hypotheses 3 and 4, Algorithm 8 is convergent when applied to Problem 1.

At this point we see the first advantage of using Abstract Models of Algorithms as a tool for proving convergence of algorithms. On one hand they provide us with a pattern to follow, while one the other, we find that the proof of convergence of a specific algorithm becomes decomposed into fairly simple and independent parts.

In Algorithm 8, the computation of z_{i+1} from z_i is not explicit. We therefore proceed as indicated in Section II in order to produce truncations in the calculation of z_{i+1} from z_i .
Making use of Proposition 3, we now define a subalgorithm which we shall use in order to modify Algorithm 8.

For every z in R^n, y in R^n and j a positive integer, consider the following subprocedure.

Algorithm 9. Let $\alpha \in (0,1)$ be given.

Step 0: Set $y_0 = y, i = 0$ and $\nu = 0$.

Step 1: If $< \nabla f(z), \nabla f(y_i) > = 0$ go to Step 6, else let $\nu = -\operatorname{sgn} < \nabla f(z), \nabla f(y_i) >$.

Step 2: Set $\tilde{y} = y_i + \nu \nabla f(z)$.

Step 3: Compute Θ defined by $\Theta = f(\tilde{y}) - f(y_i) + \alpha \nu < \nabla f(z), \nabla f(y_i) >$.

Step 4: If $\Theta \leq 0$, let $y_{i+1} = \tilde{y}$ and go to Step 5, else let $\nu = \nu/2$ and go to Step 2.

Step 5: If $i < j$ set $i = i + 1$ and go to Step 1, otherwise go to step 6.

Step 6: Let $\tilde{\nu} = \text{sat } \nu$[+], set $y_j = y_0 + \nu \nabla f(z)$ and stop.

Remark. We choose this subalgorithm because it has a nontrivial

---

[+]Note: The function sat (.): $R^1 \to R^1$ is defined by

Sat $(\nu) = \nu$ if $|\nu| \leq 1$;

Sat $(\nu) = 1$ if $\nu > 1$;

Sat $(\nu) = -1$ if $\nu < -1$.

amount of structure rather than because it is the best computationally. The structure of this subalgorithm should serve the purpose of illustrating the complexity that can be found in a subalgorithm for computing the values of the map $m(\cdot,\cdot,\cdot)$.

Definition 6. Let $m(\cdot,\cdot,\cdot)$ be the map from T x T x N into T defined by: $m(z,y,j) = y_j$ where y_j is given by Algorithm 9. It can be verified easily that the map $m(\cdot,\cdot,\cdot)$ given by Definition 6 satisfies Hypotheses 7 and 9. Using the map $m(\cdot,\cdot,\cdot)$ and a truncation function $\ell(\cdot)$ we obtain from Algorithm 8 the following "explicit" algorithm.

Algorithm 10. Let $\ell(\cdot)$ be a given truncation function.

Step 0: Compute a z_0 in R^n and set $i = 0$.

Step 1: Let $z_{i+1} = m(z_i, z_i, \ell(i))$.

Step 2: If $f(z_{i+1}) < f(z_i)$ set $i = i + 1$ and go to Step 1, otherwise stop.

In view of Proposition 5, the following is obvious.

Proposition 7. If Algorithm 10 generates an infinite sequence of points $\{z_i\}$ when applied to Problem 1 then every cluster point of the sequence is desirable.

Remark. It can be shown that if the sequence of points generated by Algorithm 10 when applied to Problem 1 is finite i.e. $\{z_i\} = \{z_0, z_1, \ldots z_k\}$ then z_{k-1} is desirable. Consequently, Algorithm 10 is

convergent for Problem 1.

III-2. Constrained Minimization Problem.

In this subsection we shall examine the following classical problem.

Problem 2. Given T a closed, bounded convex subset of R^n and t a point in R^n find $\hat{z}$ in T such that

$$||t - \hat{z}|| \leq ||t - z|| \text{ for all } z \text{ in } T.$$

We shall suppose that T is of the form

$$T = \{z \in R^n \mid f^i(z) \leq 0 \quad i = 1, 2, \ldots, m\}$$

where the maps $f^i(\cdot)$ from R^n into R^1 are continuously differentiable.

Suppose that we try to solve Problem 2 by means of the Frank-Wolfe Algorithm.

Algorithm 11.

Step 0: Compute a point z_0 in T and set i = 0.

Step 1: Compute a point w_i in T satisfying $\langle t - z_i, w_i \rangle \geq \langle t - z_i, w \rangle$ for all w in T.

Step 2: Let z_{i+1} be the point in $[z_i, w_i]$ satisfying $||t - z_{i+1}|| \leq ||t - z||$ for all z in $[z_i, w_i]^+$.

+Note: Given two points x_1 and x_2 in R^n, the set $\{y \in R^n \mid y = \nu x_1 + (1-\nu) x_2, 0 \leq \nu \leq 1\}$ is denoted by $[x_1, x_2]$.

Step 3: If $||t - z_{i+1}|| < ||t - z_i||$ let $i = i + 1$ and go to Step 1, otherwise stop.

Algorithm 11 can be seen to be of the form of Algorithm 3 by defining the following maps.

Definition 7. Let the set $U = T$ and the maps $\alpha(\cdot)$, $\xi(\cdot)$ from T into R^1, $\beta(\cdot,\cdot)$ from $T \times U$ into R^1, $b(\cdot,\cdot)$ from $T \times U$ into T be defined as follows:

(i) $\alpha(z) = \max \{\alpha \mid \alpha = < t - z, w >, \ w \in T\}$;

(ii) $\xi(z) = ||t - z||$;

(iii) $\beta(z,w) = < t - z, w >$;

(iv) $b(z,w)$ is defined by:

(a) $b(z,w)$ belongs to $[z,w]$;

(b) $||t - b(z,w)|| \leq ||t - y||$ for all y in $[z,w]$.

In order to show that Algorithm 11 is convergent for Problem 2, it suffices to show that the mappings $\alpha(\cdot)$, $\xi(\cdot)$, $\beta(\cdot,\cdot)$ and $b(\cdot,\cdot)$ satisfy Hypotheses 5 and 6. It is easily verified that this is indeed so and we therefore conclude that Algorithm 11 is convergent for Problem 2.

Once again we see the advantage of using an abstract model in proving the convergence of a specific algorithm.

In Algorithm 11, both the point w_i and the point $b(z_i,w_i)$ are defined by implicit relations. However, since the computation of $b(z_i,w_i)$ from z_i and w_i is extremely simple, we shall consider that $z_{i+1} = b(z_i,w_i)$ is an "explicit" function of z_i and w_i. Thus we shall consider that the only real difficulty lies in the computation of w_i.

To obtain from Algorithm 11 an "explicit" algorithm of the form of Algorithm 7, we must introduce a map $m(\cdot,\cdot,\cdot)$ satisfying Hypotheses 8 and 10. For example, one can use a method of feasible directions to define such a map.

We now state a method of feasible directions in the required truncated form.

For every z in T, y in T and positive integer j, consider the following algorithm.

Algorithm 12. Suppose that S a compact neighborhood of the origin in R^n, ε and ε' positive scalars such that $\varepsilon > \varepsilon' > 0$ are given. Let $f^0(\cdot)$ be the map from R^n into R^1 defined by $f^0(x) = \langle z - t, x \rangle$ for all x in R^n.

Step 0: Set $y_0 = y$ and $i = 0$.

Step 1: Set $\varepsilon_i = \varepsilon$.

Step 2: Compute ϕ_{ε_i} and h_{ε_i} by solving the following:

$$\phi_{\varepsilon_i} = \min_{h \in S} \max_{j \in J_{\varepsilon_i}} \langle \nabla f^j(y_i), h \rangle$$

h_{ε_i} is any vector in S such that $\phi_{\varepsilon_i} = \max_{j \in J_{\varepsilon_i}} \langle \nabla f^j(y_i), h_{\varepsilon_i} \rangle$.

where $J_{\varepsilon_i} = \{j \in \{1, 2, \ldots, m\} \mid f^j(y_i) + \varepsilon_i \geq 0\} \cup \{0\}$.

Step 3: If $\phi_{\varepsilon_i} < -\varepsilon_i$ set $h_i = h_{\varepsilon_i}$ and go to Step 4.

If $\phi_{\varepsilon_i} \geq -\varepsilon_i$ and $\varepsilon_i \leq \varepsilon'$ compute ϕ_0 defined by

$\phi_0 = \min_{h \in S} \max_{j \in J_0} \langle \nabla f^j(y_i), h \rangle$ where

$J_0 = \{j \in \{1, 2, \ldots, m\} \mid f^j(y_i) = 0\} \cup \{0\}$.

If $\phi_0 = 0$, set $y_{i+1} = y_i$, $i = i + 1$ and go to Step 1.

If $\phi_0 < 0$, set $\varepsilon_i = \frac{\varepsilon_i}{2}$ and go to Step 2.

Step 4: Compute $\lambda_i \geq 0$ such that

$\lambda_i = \max \{\lambda \mid f^j(y_i + \lambda h_i) \leq 0$ for all $j = 1, 2, \ldots, m\}$

Step 5: Compute μ_i in $[0, \lambda_i]$ such that

$f^0(y_i + \mu_i h_i) \leq f^0(y_i + \mu h_i)$ for all μ in $[0, \lambda_i]$.

Step 6: If $i < j$, set $y_{i+1} = y_i + \mu_i h_i$, $i = i + 1$ and go to Step 1, otherwise stop.

Definition 8. Let $m(\cdot,\cdot,\cdot)$ be the map from T x T x N into T defined by $m(z,y,j) = y_j$ where y_j is given by Algorithm 12. It can be verified that the map $m(\cdot,\cdot,\cdot)$ given by Definition 8 satisfies Hypothesis 8 and 10 (see [9]). Using the map $m(\cdot,\cdot,\cdot)$ and a truncation function $\ell(\cdot)$ in Algorithm 11 we obtain the following "explicit" method.

Algorithm 13. Let $\ell(\cdot)$ be a given truncation function.

Step 0: Compute a point z_0 in T and set $i = 0$.

Step 1: Set $w_i = m(z_i, z_i, \ell(i))$.

Step 2: Let z_{i+1} be the point in $[z_i, w_i]$ satisfying $||t - z_{i+1}|| \leq ||t - z||$ for all z in $[z_i, w_i]$.

Step 3: If $||t - z_{i+1}|| < ||t - z_i||$ let $i = i + 1$, and go to Step 1, otherwise stop.

In view of Proposition 6, the following is obvious.

Proposition 8. If Algorithm 13 generates an infinite sequence of points $\{z_i\}$ when applied to Problem 2, then every cluster point of the sequence is desirable.

Remark. It can be shown that if the sequence of points generated by Algorithm 13 is finite i.e. $\{z_i\} = \{z_0, z_1, \ldots, z_k\}$ then z_{k-1} is desirable. Consequently, Algorithm 13 is convergent for Problem 2.

The use of the approach defined in this paper may show relations between different well known algorithms. For example, if a function $\ell(\cdot)$ from N into N defined by $\ell(i) = 1$ for all i is used in Algorithm 13 instead of a truncation function, then Algorithm 13 is a method of feasible directions. On the other hand if a "function" $\ell(\cdot)$ from N into N "defined" by $\ell(i) = \infty$ for all i is used in Algorithm 13 instead of a truncation function, then Algorithm 13 is the Frank-Wolfe Algorithm. The use of truncation function in Algorithm 13 thus produce algorithms which are "between" a method of feasible directions and the Frank-Wolfe Algorithm.

IV. ε - approximations for a Class of Iterative Procedures

The approach to the synthesis of algorithms described in Sections I and II is by no means the only one possible. In this section we shall show that an alternative approach exists and gives extremely interesting results.

Throughout this section we shall consider maps $A(\cdot)$ and $\xi(\cdot)$ satisfying Hypothesis 1 in Section I and we will suppose that it is impossible to use them in iterative procedures of the form of Algorithm 1 due to the fact that the computation of points z_{i+1} in $A(z_i)$ is impossible (or that we are not aware of ways of doing it).

The main idea developed in this section consists in using a map $B(\cdot,\cdot)$ from $R_+^1 \times T$ into all the subsets of T such that the set $B(0, z)$ is identical to the set $A(z)$ for all z.

We shall suppose that the task of finding a y in $B(\varepsilon,z)$ for $\varepsilon > 0$ and z in T is relatively easy.

Consider the following algorithm.

Algorithm 14. Let $\bar{\varepsilon} > 0$ be given

Step 0: Compute z_0 in T and set i = 0.

Step 1: Let $\varepsilon = \bar{\varepsilon}$.

Step 2: Find a point y in $B(\varepsilon, z_i)$.

Step 3: If $\xi(y) - \xi(z_i) \leq -\varepsilon$ set $\varepsilon_i = \varepsilon$, $z_{i+1} = y$, i=i+1 and go to Step 1, otherwise let $\varepsilon = \varepsilon/2$ and go to Step 2.

We remark that if we let $\bar{\varepsilon} = 0$ then Algorithm 14 is almost of the

form of Algorithm 1. The following assumption on $B(\cdot,\cdot)$ ensures that cluster points of infinite sequences generated by Algorithm 14 are desirable.

Hypothesis 11.

(i) $B(0, z) = A(z)$ for all z in T;

(ii) $B(\cdot,\cdot)$ is jointly closed in both its arguments i.e. for any sequence $\{\varepsilon_i\}$ converging to ε^*, for any sequence $\{z_i\}$ converging to z^*, for any sequence $\{y_i\}$ converging to y^*, with y_i in $B(\varepsilon_i, z_i)$, y^* belongs to $\beta(\varepsilon^*, z^*)$.

Theorem 1. If the maps $A(\cdot)$ and $\xi(\cdot)$ satisfy Hypothesis 1, the map $B(\cdot,\cdot)$ satisfies Hypothesis 11, then every cluster point of an infinite sequence generated by Algorithm 14 is desirable.

Proof. Let $\{z_i\}$ be an infinite sequence generated by Algorithm 14 and let z^* be a cluster point of this sequence. Thus, for some subset K_1 of the integers, the subsequence $\{z_i\}_{K_1}$ converges to z^*. Consider the sequence $\{z_{i+1}\}_{i+1 \in K_1}$ in T and the sequence $\{\varepsilon_i\}_{i \in K_1}$ in $[0, \bar{\varepsilon}]$. The boundedness of T and of the interval $[0, \bar{\varepsilon}]$ ensure that there exists K, an infinite subset of K_1 such that the subsequences $\{z_{i+1}\}_{i+1 \in K}$ and $\{\varepsilon_i\}_K$ converge to points z^{**} and ε^* in T and $[0, \bar{\varepsilon}]$ respectively. The properties of convergent sequence ensure that the subsequence $\{z_i\}_K$ also converges to z^*.

Now suppose that $\varepsilon^* > 0$, then the form of Algorithm 14 implies that there exists an integer k such that $\xi(z_{i+1}) - \xi(z_i) \leq -\frac{\varepsilon^*}{2}$ for

all $i \geq k$, i in K and this contradicts part (ii) of Hypotheses 1. Consequently $\varepsilon^* = 0$.

The map $B(\cdot,\cdot)$ is jointly closed in both its arguments, and z_{i+1} belongs to $B(\varepsilon_i, z_i)$ for all i. It follows that z^{**} is in $B(\varepsilon^*, z^*)$ i.e. in $A(z^*)$. Consequently z^* is desirable.

Hypothesis 12.

(i) $B(0, z) = A(z)$ for all z in T;

(ii) If $\xi(a) - \xi(z) < 0$ for all a in $A(z)$ then there exist $\varepsilon > 0$, $\delta > 0$ and $\gamma > 0$, possibly depending on z, such that

$$\xi(b') - \xi(z') \leq -\gamma < 0$$

for all z' in T such that $||z' - z|| \leq \varepsilon$ and for all b' in $B(\nu, z')$, $0 \leq \nu \leq \delta$.

The proof of the following theorem can be carried out by using the same type of arguments as were used to prove Theorem 1, and is therefore omitted.

Theorem 2. If the maps $A(\cdot)$ and $\xi(\cdot)$ satisfy Hypothesis 2, the map $B(\cdot,\cdot)$ satisfies Hypothesis 12, then every cluster point of an infinite sequence generated by Algorithm 14 is desirable.

Remark. It can be shown that if maps $A(\cdot)$, $\xi(\cdot)$ and $B(\cdot,\cdot)$ satisfy Hypotheses 1 and 11, they satisfy Hypotheses 2 and 12 (see [1]).

For examples of how the ε - procedure is used in the synthesis of algorithms, see E. Polak, "Computational Methods in Optimization: A Unified Approach," Academic Press, 1970.

V. CONCLUSION

To conclude, we should like to highlight the two most important aspects of the theory we have presented in this paper. The first is that by using models one can separate out the essential properties of an algorithm from the non essential one. Thus, for example, in a gradient method one need not specify in advance exactly which procedure one will use to search along the direction of steepest descent, one only has to specify that the search procedure will have certain properties. The second point that we wish to emphasize is that given a "conceptual" algorithm in which one has to perform in sequence several operations each of which requires an infinite number of iterations, one can obtain an "implementable" algorithm by "shutling" between these infinite operations, combining them into a single infinite operation. Thus, our methods of obtaining an "implementable" algorithm from a plurally infinitely iterative "conceptual" algorithm consists in "parallelizing" the infinite operations of the conceptual algorithm.

Obviously, this paper does not exhaust the study of models for computational methods nor of the possibilities of constructing "implementable" algorithms from "conceptual" ones. We hope that this paper will lead to further work, in particular in the study of algorithms with memory.

REFERENCES

[1] E. Michael, "Topologies on Spaces of Subsets," American Mathematical Society Transactions, Vol. 71 (1951), pp. 152-182.

[2] M. Frank and P. Wolfe, "An Algorithm for Quadratic Programming," U. S. Naval Res. Logist. Quart., Vol. 3 (1956), pp. 95-110.

[3] G. Zoutendijk, Methods of Feasible Directions, Elsevier Publ. Co., Amsterdam, 1960.

[4] B. T. Polyak, "Gradient Methods for the Minimization of Functionals," USSR Computational Mathematics and Mathematical Physics, Vol. 3, No. 4 (1963), pp. 864-878. (Translation of Zh. Vychisl. Mat. i Mat. Fiz., Vol. 3, No. 4 (1963), pp. 643-653.)

[5] E. S. Levitin and B. T. Polyak, "Constrained Minimization Methods," USSR Computational Mathematics and Mathematical Physics, Vol. 6, No. 5 (1966), pp. 1-50. (Translation of Zh. Vychisl. Mat. i Mat. Fiz., Vol. G, No. 5 (1966), pp. 787-823.)

[6] W. I. Zangwill, "Convergence Conditions for Nonlinear Programming Algorithms," Working Paper No. 197, Center for Research in Management Science, University of California, Berkeley, November 1966.

[7] D. M. Topkis and A. Veinott "On the Convergence of Some Feasible Directions Algorithms for Nonlinear Programming," J. SIAM Control, Vol. 5, No. 2 (1967), pp. 268-279.

[8] E. Polak, "On Primal and Dual Methods for Solving Discrete Optimal Control Problems," Proceedings, Second International Conference on Computing Methods in Optimization Problems, San Remo, Italy, Sept. 9-13, 1968, Academic Press, 1969.

[9] E. Polak, "On the Convergence of Optimization Algorithms," Revue Francaise d' Informatique et de Recherche Operationelle, Serie Rouge, No. 16, 1969, pp. 17-34.

[10] W. I. Zangwill, Nonlinear Programming: A Unified Approach, Prentice Hall Inc., Englewood Cliffs, N. J., 1969.