The Sixth International Symposium on Operations Research and Its Applications (ISORA'06)
Xinjiang, China, August 8–12, 2006

The Branch and Bound Algorithm for Solving a Sort of Non-smooth Programming on Simplex*

Zai-En Hou[1,†] Fu-Jian Duan[2]

[1] Shaanxi University of Science & Technology, Xianyang, Shaanxi 712081, China
[2] Guilin University of Science & Technology, Guilin, Guangxi 541004, China

Abstract An approximation algorithm for solving a sort of non-smooth programming is proposed. In the programming, the objective function is the Hölder function and the feasible region is contained in a simplex (viz. hyper-simplex). To establish the algorithm, the properties of the Hölder function and an approximation of the function by using Bernstein α-polynomial are studied. According to the properties of the approximation polynomial and the algorithm for solving geometric programming, the strategy for branching and bounding and the branch-and-bound algorithm are constructed to solve the programming. The convergence of the algorithm can be guaranteed by the exhaustive of the bisection of the simplex. The feasibility of the algorithm is validated by solving an example.

Keywords Hölder function; non-smooth programming; approximation polynomial; geometrical programming; branch-and-bound algorithm

1 Introduction

The following model is called as non-smooth programming

$$(\text{NSP})\begin{cases}\min & f(x)\\ \text{s.t.} & g_i(x)\le 0,(i=1,2,\cdots,m)\\ & h_j(x)=0,(j=1,2,\cdots,n),\end{cases}$$

where there exists one non-smooth function in $f(x)$, $g_i(x)$ $(i=1,2,\cdots,m)$ and $h_j(x)$ $(j=1,2,\cdots,n)$. The study for solving the programming has attracted much attention since the programming is widely existed in practical engineering and the penalty function of a constrained smooth programming is generally a non-smooth function. According to the properties of the programming and the study on smooth programming, the necessary and sufficient conditions and the basic methods are established to

*This work is supported by (Doctor) Scientific Research Startup Foundation of Shaanxi University of Science & Technology: BJ06-2.
†E-mail: hkze-00795@163.com

solve the programming (see [1–3]). And some modern methods for solving smooth programming are applied to solve the programming, such as bundle method, trust region method, reformed Newton method and SQP method, which is a successive approximation method with quadratic programming (see [4–9]). Some of these methods have requests or restrictions on smoothness or convexity to the function in the programming. The typical example is the Lipschitz optimization in which the Lipschitzian continuity is assumed to the objective and constraint function (see [10,11]). Can these restrictions be weakened in model (NSP)? These motivate us to study the properties of the Hölder function, the approximation of the function by using Bernstein α-polynomial and the strategy for branching and bounding. Our goal is to develop a new method for solving the programming where the objective function and constrained functions are all Hölder continues (see [12]).

The content of this paper is as follows. In section 2 we review the properties of the Hölder function and the approximation of the function by using Bernstein α-polynomial. In section 3 we construct the algorithm and study the convergence of the algorithm for solving the programming. In section 4 we give a numerical example to illuminate the feasibility of the algorithm. We end the paper with the conclusion of the paper in section 5.

2 The Properties of Hölder Function

2.1 Hölder Function

In this section we review the concept of Hölder function and study several properties of the function.

Definition 1. Let $f(x)$ be a real function on $P(\subset \mathbb{R}^n)$, $f(x)$ is called a Hölder function on P(or Hölder continuous) if there exist constant $L = L(f,P) > 0$ and $\gamma > 0$ such that

$$|f(x_2) - f(x_1)| \leq L \parallel x_2 - x_1 \parallel^{\gamma}, \text{ for all } x_1, x_2 \in P. \tag{1}$$

Where constants $L = L(f,P) > 0$ and $\gamma > 0$ are called Hölder constants of $f(x)$.

In the definition the norm $\parallel \cdot \parallel$ is the general Euclidean norm. In practice, the following l_p- norm is often adopted

$$\parallel x \parallel_p = \left(\sum_{i=1}^{s} |x_i|^p \right)^{\frac{1}{p}}, \ (1 \leq p \leq \infty), \tag{2}$$

where $\parallel x \parallel_\infty = \max_{i=1,\cdots,s} |x_i|$.

Obviously, Hölder function $f(x)$ is a Lipshchitz function on P when $\gamma = 1$, that is, Lipshchitz function is a special kind of Hölder function. Even so, the continuity of them is alike, and other properties of them are analogous as well.

Lemma 1. *Let $f(x)$,$g(x)$,$f_i(x)$ $(i = 1,\cdots,m)$ be Hölder functions on the compact set $P \subset \mathbb{R}^s$, then for any $x \in P$, there exists a neighborhood $U(x)$ at x, such that*

(i) *Every linear combination of* $f_i(x)(i = 1, \cdots, m)$ *is a Hölder function on* $U(x) \cap P$*;*

(ii) $\max_{i=1,\cdots,m} f_i(x)$ *and* $\min_{i=1,\cdots,m} f_i(x)$ *are Hölder functions on* $U(x) \cap P$*;*

(iii) $f(x) \cdot g(x)$ *is a Hölder function on* $U(x) \cap P$*.*

Proof. The constructive method may be used in the proofs of $(i) \sim (iii)$. The proof of $\max_{i=1,\cdots,m} f_i(x)$ in (ii) is given here, the others are similar.

Suppose L_i and γ_i $(i = 1, \cdots, m)$ are Hölder constants of function $f_i(x)$ $(i = 1, \cdots, m)$, respectively, by the definition 1, we have following inequalities

$$|f_i(x_2) - f_i(x_1)| \leq L_i \|x_2 - x_1\|^{\gamma_i}, \text{ for all } x_1, x_2 \in P, (i = 1, \cdots, m). \tag{3}$$

Let

$$F(x) = \max_{i=1,\cdots,m} f_i(x) \tag{4}$$

and $U(x) = \{\bar{x} | \|\bar{x} - x\| < d\}$, for any $x \in P$, we choose d such that $0 < d \leq 1$, then for all $x_1, x_2 \in U(x) \cap P$, we have following inequalities

$$\begin{aligned} |F(x_2) - F(x_1)| =& |\max_i f_i(x_2) - \max_i f_i(x_1)| \\ \leq& \max_i |f_i(x_2) - f_i(x_1)| \\ \leq& \max_i \{L_i \|x_2 - x_1\|^{\gamma_i}\} \\ \leq& (\max_i L_i) \max_i \{\|x_2 - x_1\|^{\gamma_i}\} \\ \leq& (\max_i L_i) \|x_2 - x_1\|^{(\min_i \gamma_i)}. \end{aligned} \tag{5}$$

Further, let $L = \max_i L_i$, $\gamma = \min_i \gamma_i$, the inequality (5) can be denoted as follows

$$|F(x_2) - F(x_1)| \leq L \|x_2 - x_1\|^{\gamma}. \tag{6}$$

This shows that $\max_{i=1,\cdots,m} f_i(x)$ is a Hölder function on $U(x) \cap P$. □

Lemma 2. *Let* $f_i(x)$ $(i = 1, \cdots, m)$ *be a Hölder function on the compact set* $P \subset \mathbb{R}^s$ *and* L_i, γ_i $(i = 1, \cdots, m)$ *be the Hölder constants respectively, then for any* $x \in P$*, there exists a neighborhood* $U(x)$ *at* x*, such that for any* p $(1 \leq p \leq \infty)$ *the function* $f(x) = (\sum_{i=1}^m |f_i(x)|^p)^{\frac{1}{p}}$ *is a Hölder function on* $U(x) \cap P$ *and the constants* $L = \sum_{i=1}^m L_i$ *and* $\gamma = \min_i \gamma_i$ *are its Hölder constants.*

Proof. Let $F(x) = (f_1(x), \cdots, f_m(x))^T$, then $f(x)$ is the l_p-norm of $F(x)$, that is $f(x) = \|F(x)\|_p$. Let $U(x) = \{\bar{x} | \|\bar{x} - x\| < d\}$, for any $x \in P$, where we choose d

such that $0 < d \leq 1$, then for all $x_1, x_2 \in U(x) \cap P$, we have following inequalities

$$\begin{aligned} |\|F(x_2)\|_p - \|F(x_1)\|_p| \leq & \|F(x_2) - F(x_1)\|_p \\ \leq & \|F(x_2) - F(x_1)\|_1 \\ = & \sum_{i=1}^{m} |f_i(x_2) - f_i(x_1)| \\ \leq & \sum_{i=1}^{m} L_i \|x_2 - x_1\|^{\gamma_i} \\ \leq & \sum_{i=1}^{m} L_i \|x_2 - x_1\|^{(\min_i \gamma_i)} \\ = & \left(\sum_{i=1}^{m} L_i\right) \|x_2 - x_1\|^{(\min_i \gamma_i)} \end{aligned} \tag{7}$$

Therefore, let $L = \sum_{i=1}^{m} L_i$, $\gamma = \min_i \gamma_i$, the inequality (7) can be expressed as follows

$$|f(x_2) - f(x_1)| \leq L\|x_2 - x_1\|^{\gamma}. \tag{8}$$

It is shown that $f(x)$ is a Hölder function on $U(x) \cap P$. $\square$

Actually, the functions in the results of the lemma are all local Hölder function.

2.2 The Approximation of Hölder Function

From the definition 1, Hölder function is continuous on P while $P(\subset \mathbb{R}^n)$ is simplex (viz. hyper-simplex). In this case the Hölder function can be approximated by Bernstein α- polynomials (like in paper [13]). But the labor of calculation of the approximation is increasing rapidly when the degree of the Bernstein α- polynomials is increasing in higher order. The higher degree Bernstein α- polynomials can not be used in practical computation. We pay attention to the process of branching and bounding, in which the region is divided smaller and smaller. Can the approximation by Bernstein α- polynomials be applied in this process, that is, can the heightening of the degree of the polynomials be substituted by the dwindling of the region? If it is possible, we can conclude that the Hölder function can be approximated by Bernstein α- polynomials in the process mentioned. We explain the feasibility of this approximation in the section below, then we will give the application of the approximation to the approximation algorithm in the next section.

Lemma 3. *Let $f(x)$ be a Hölder function on the simplex $P \subset \mathbb{R}^s$, $\{P_m\}$ be a sequence of simplexs such that*

$$P = P_0 \supset P_1 \supset \cdots \supset P_m \supset \cdots,$$

and $d_m = \sup\{\|x_2 - x_1\| \,|\, x_2, x_1 \in P_m\}$ be the diameter of simplex $P_m (m = 0, 1, 2, \cdots)$, such that $d_m \to 0$, as $m \to \infty$. And let

$$B_n^m(f, x, \alpha) = \sum_{|k| \leq n} f((\frac{k}{n})^{\frac{1}{\alpha}}) B_{n,k}(x, \alpha) \tag{9}$$

be the Bernstein α-polynomials of $f(x)$ on $P_m(m=0,1,2,\cdots,\cdots)$, then we have the following limit

$$\lim_{m\to\infty}(B_n^m(f,x,\alpha)-f(x))=0. \tag{10}$$

Proof. It is easy to know the $f(x)$ is a Hölder function on the simplex $P_m \subset P(m=0,1,\cdots)$ from the assumption of the lemma. Therefore, we have following inequality:

$$|f(x_2)-f(x_1)| \le L\|x_2-x_1\|^\gamma, \quad x_2,x_1\in P_m(m=0,1,\cdots) \tag{11}$$

where L and $\gamma>0$ are the Hölder constants. We also have inequalities

$$\begin{aligned}|B_n^m(f,x,\alpha)-f(x)| &= \left|\sum_{|k|\le n}\left(f((\frac{k}{n})^{\frac{1}{\alpha}})-f(x)\right)B_{n,k}(x,\alpha)\right| \\ &\le \sum_{|k|\le n}\left|f((\frac{k}{n})^{\frac{1}{\alpha}})-f(x)\right|B_{n,k}(x,\alpha) \\ &\le \sum_{|k|\le n}L\left\|f(\frac{k}{n})^{\frac{1}{\alpha}}-x\right\|^\gamma B_{n,k}(x,\alpha) \\ &\le Ld_m^\gamma\sum_{|k|\le n}B_{n,k}(x,\alpha)=Ld_m^\gamma.\end{aligned} \tag{12}$$

From the assumption, when $m\to\infty$, $d_m\to 0$. Because of this, for any $\varepsilon>0$, there exists a positive number N, such that $d_m<\left(\frac{\varepsilon}{L}\right)^{\frac{1}{\gamma}}$ as $m>N$. Thus

$$|B_n^m(f,x,\alpha)-f(x)|<L\left(\left(\frac{\varepsilon}{L}\right)^{\frac{1}{\gamma}}\right)^\gamma=\varepsilon. \tag{13}$$

It shows the limit (10) holds. □

2.3 The Non-smooth Programming

For the non-smooth programming

$$(\text{NSPH})\begin{cases}\min & f(x)\\ \text{s.t.} & g_i(x)\le 0,(i=1,2,\cdots,m)\\ & h_j(x)=0,(j=1,2,\cdots,n),\end{cases}$$

where $f(x),g_i(x),h_j(x)(i=1,2,\cdots,m;j=1,2,\cdots,n)$ are Hölder functions, we define a Lagrange penalty function as follows

$$L(x,M)=f(x)+M(\sum_{i=1}^m\max\{g_i(x),0\}+\sum_{j=1}^s h_j^2(x)),$$

where M is a large positive number. Then we choose a huge simplex D which comprise the feasible region of the programming (NSPH). The programming can be solved by solving the problem

$$\begin{aligned} \min \quad & L(x,M) \\ \text{s.t.} \quad & x \in D. \end{aligned}$$

Therefore, only do we need to solve the problem

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in D, \end{aligned} \tag{14}$$

or the problem

$$\begin{aligned} \min \quad & B_n(f,x,\alpha) \\ \text{s.t.} \quad & x \in D, \end{aligned} \tag{15}$$

where the $L(x,M)$ is denoted as the new function $f(x)$ in problem (14) and (15).

3 The Algorithm and Its Convergence

The general branch and bound methodology is applicable to broad classes of optimization problems. The branch and bound algorithms are based upon operations of partition, sampling, and subsequent lower and upper bounding procedures, in which these operations are applied iteratively to the collection of active ('candidate') subsets within the feasible set. In the section we describe the algorithm for solving the non-smooth programming ((NSPH) or (14)), in which we combine the strategy of branch and bound with the approximation of Hölder function by Bernstein α-polynomials. Moreover, we study the convergence of the algorithm.

3.1 The Basic Algorithm

Compared with general branch and bound methodology, here we approximate the function $f(x)$ with its Bernstein α-polynomials $B_n(f,x,\alpha)$ and we have the inequality

$$\min_{x\in D} f(x) \leq \min_{x\in D} B_n(f,x,\alpha), \tag{16}$$

our method did not need the process of bounding in formally, it only needs the process of branching and pruning. Thus, rules of branching and pruning is the key of the method. According to these rules, we can preserve the solution in the sequence of the simplex, and we need solve lesser subproblem as well. Thus the labor of calculation needn't be added a great deal.

The Branching Rule. To guarantee the convergence of the algorithm, we partition the simplex P_k with bisection method, that is, we bisect the simplex P_k at the midpoint of its longest edge. Where P_k is partitioned as two simplex:P_{k1},P_{k2}. The diameter of simplex $P_{ki}(i=1,2)$ is dwindled step by step with the process of the partition continuing. Moreover, the diameter of the simplex $P_{ki}(i=1,2)$ converges to zero as $k \to \infty$.

The Pruning Rule. To set up the pruning rule, we need to determine the initial threshold value μ_0 and the degree n of the polynomials.

In theory, we can choose the initial threshold value $\mu_0 > 0$ and the degree n, such that

$$|B_n(f,x,\alpha) - f(x)| \le \mu_0, \ \ x \in P_0 \tag{17}$$

where $P_0 = D$, since we have the following limit 由于

$$\lim_{n\to\infty} B_n(f,x,\alpha) = f(x), \ \ x \in P_0. \tag{18}$$

But the degree n may be too large to be used in actual computation.

In practical computation, we can first choose proper n according to the properties of the function, where a smaller n is chosen generally. Then we determine $\mu_0 > 0$ with the bisection method or 0.618 method (golden section method). Moreover, the following inequalities hold,

$$\min_{x\in P_0} B_n(f,x,\alpha) - \mu_0 \le \min_{x\in P_0} f(x) \le \min_{x\in P_0} B_n(f,x,\alpha). \tag{19}$$

Let $u = \min_{x\in P_0} B_n(f,x,\alpha)$ be the upper bound of the optimization value of objective function, according to the branching rule, we partition the simplex P_0 as: P_0^1, P_0^2, and solve $\min_{x\in P_0^1} B_n(f,x,\alpha), \min_{x\in P_0^2} B_n(f,x,\alpha)$ at the same time. Then Comparing the optimization values of them with u, if $u > \min_{x\in P_0^i} B_n(f,x,\alpha)$, we update the upper bound u as $u = \min_{x\in P_0^i} B_n(f,x,\alpha)$. Actually, that can be expressed as follows,

$$u = \min\left\{\min_{x\in P_0^1} B_n(f,x,\alpha), \min_{x\in P_0^2} B_n(f,x,\alpha)\right\} \le \min_{x\in P_0} B_n(f,x,\alpha),$$

where the Bernstein α-polynomial $B_n(f,x,\alpha)$ in $\min_{x\in P_0^1} B_n(f, x, \alpha)$, $\min_{x\in P_0^2} B_n(f, x, \alpha)$ and $\min_{x\in P_0} B_n(f,x,\alpha)$ is the Bernstein α-polynomial $B_n(f,x,\alpha)$ on corresponding simplex, respectively. For convenience, we denote them as the same front though they are different. This is the same in lemma 4 and hereinafter.

Like the proof of the theorem 3 in paper [13], we have following inequality

$$|B_n(f,x,\alpha) - f(x)| < \frac{M}{n}.$$

Therefore, we can take $\mu_1 = \frac{\mu_0}{2}$ as a new threshold value and pruning with the value. If $\min_{x\in P_0^i} B_n(f,x,\alpha) \ge u + \mu_1$, then we pruning the branch on P_0^i ($i = 1$ or 2), in which there does not exist the optimization solution; otherwise we save the branch.

When we complete the work above, we collect the reserved branches to form a set and number the branches in the set over again. For instance, if two branches above did not be pruned, we have the set $P_1 = \{P_{11}, P_{12}\}$, which is called as the set of reserved branches and the diameter of which is defined as follows

$$d_1 = \max\{d_{1j} | d_{1j} \text{is the diameter of } P_{1j}, P_{1j} \in P_1\}.$$

Afterwards, we do the process of branch and bound on P_1, and gain the new set of reserved branches P_2. Generally, repeating this process, we can obtain the set of reserved branches

$$P_m = \{P_{m1}, P_{m2}, \cdots, P_{ml_m}\}, \tag{20}$$

and corresponding upper bound u, threshold value μ_m diameter d_m.

Until satisfactory precision being reached, the above process is repeated.

The Terminate Rule. For given precision $\varepsilon > 0$, if

$$\mu_m < \varepsilon \text{ or } d_m < \varepsilon,$$

then stop the process, and solve the optimization solution and value.

Algorithm 1. (*Basic Algorithm*)

Step 0: *Let an accuracy $\varepsilon > 0$ be given. Initialization: Initialize iterative number $m := 0$, initialize the set of reserved branches $P_0 = \{P_{01}\}, P_{01} = D$, choose proper integer n, and solve*

$$\min_{x \in P_{01}} B_n(f, x, \alpha)$$

to obtain the solution x_0 and initial upper bound

$$u_0 = \min_{x \in P_{01}} B_n(f, x, \alpha),$$

and determine the initial threshold value μ_0. Let d_0 be the diameter of P_{01}, which is called as diameter of P_0 too.

Step 1: *If $\mu_m < \varepsilon$ or $d_m < \varepsilon$, then stop the iterative process. We obtain u_m is the optimization value and the corresponding solution is the optimization solution. Otherwise, go to step 2.*

Step 2: *Denote the set of reserved branches as:*

$$P_m = \{P_{m1}, P_{m2}, \cdots, P_{ml_m}\}. \tag{21}$$

Bisect every $P_{mj} (1 \le j \le l)$ as : $P_{mj}^1, P_{mj}^2 (1 \le j \le l_m)$.

Step 3: *Solve problems*

$$\min_{x \in P_{mj}^i} B_n(f, x, \alpha), \quad j = 1, 2, \cdots, l_m;\ i = 1, 2.$$

Get the solutions $x_{mj}^i (j = 1, 2, \cdots, l_m;\ i = 1, 2)$; determine the upper bound

$$u_{m+1} = \min_{1 \le j \le l_m; i=1,2} \left\{ \min_{x \in P_{mj}^i} B_n(f, x, \alpha) \right\},$$

and corresponding solution x_{m+1}.

Step 4: *Let* $\mu_{m+1} = \frac{\mu_m}{2}$ *be the new threshold value, if*

$$\min_{x \in P^i_{mj}} B_n(f, x, \alpha) > u + \mu_{m+1},$$

we abnegate the corresponding branch P^i_{mj}. *Otherwise, we save the corresponding branch, which constitute the new set of reserved branches*

$$P_{m+1} = \{P_{(m+1)1}, P_{(m+1)2}, \cdots, P_{(m+1)l_{m+1}}\}. \tag{22}$$

Let

$$d_{m+1} = \max_{1 \le j \le l_{m+1}} \left\{ d_{(m+1)j} | d_{(m+1)j} \textit{is the diameter of } P_{(m+1)j},\ 1 \le j \le l_{m+1} \right\}$$

be the diameter of P_{m+1}.

Step 5: $m+1 \Rightarrow m$, *go to step* 1.

3.2 The Convergence of the Algorithm

To give the convergence of the algorithm we review the following concept.

Definition 2. Let P be a simplex, then partition the simplex. If this partition produce a sequence $\{P_m\}$ of partition set of the simplex P such that

$$P = P_0 \supset P_1 \supset \cdots \supset P_m \supset \cdots,$$

and $\lim_{m\to\infty} d(P_m) = 0$, 且$\lim_{m\to\infty} P_m = \bigcap_{m=0}^{\infty} P_m = \{\hat{P}\}$, where $d(P_m)$ is the diameter of P_m and $\hat{P}$ is a point in the simplex P, then this partition $\{P_m\}$ of P is called as exhaustive.

In the aforesaid algorithm, the set of partition is the partition of simplex, this simplex partition is exhaustive from the reference [14]. Therefore, the branch and bound algorithm is exhaustive. So the algorithm is convergent from the exhaustive of the algorithm and lemma (3).

Theorem 4. *If the process of branch and bound algorithm can be done infinitely, and this process produce every sequence of the partition sets, such that*

$$P_{m_1} \supset P_{m_2} \supset \cdots \supset P_{m_k} \supset \cdots,$$

and it is exhaustive, then

$$u = \lim_{k\to\infty} u_{m_k} = \lim_{k\to\infty} f(x_{m_k})$$

and the accumulation point x^* *of the sequence* $\{x_m\}$ *is the optimization solution of the programming* (14).

Proof. There exists a accumulation point of the sequence $\{x_m\}$ since the set D is a compact set in the assumption of the theorem. Let x^* be the accumulation point, then there exists a subsequence $\{x_{m_k}\}$ of sequence $\{x_m\}$, such that the subsequence $\{x_{m_k}\}$ converge to the point x^* and corresponding sequence $\{P_{m_k}\}$

$$P_{m_1} \supset P_{m_2} \supset \cdots \supset P_{m_k} \supset \cdots$$

is exhaustive. Combining the continuity of the function $f(x)$ and the results of lemma 4, we have the limit

$$\lim_{k\to\infty} f(x_{m_k}) = f(x^*),$$

and the limit

$$\lim_{k\to\infty} u_{m_k} = \lim_{k\to\infty} f(x_{m_k}).$$

Thus from the meaning of u_{m_k}, we know the point x^* is the optimization solution of the programming (14) and the corresponding limit $u = \lim_{k\to\infty} u_{m_k}$ is the optimization value. □

Clearly, for given $\varepsilon > 0$, the process of branch and bound will be stopped at finite number of steps under the terminate rule and results of the theorem. In other words, the algorithm possesses the property of finite termination.

4 Numerical Example

Example 1. Solve the problem

$$\max_{x\in S} \quad f(x) \tag{23}$$

where $S = \{x : x = (x_1,x_2,x_3)^T \in \mathbb{R}^3, 0 \le x_i \le 1, i = 1,2,3, x_1+x_2+x_3 \le 1\}$, and

$$\begin{aligned} f(x) =& \max\{f_1(x), f_2(x)\},\\ f_1(x) =& -1.0 + 8x_1 + 8x_2 - 32x_1x_2,\\ f_2(x) =& 3.6 - 12x_1 - 4x_3 + 4x_1x_3 + 10x_1^2 + 2x_3^2. \end{aligned}$$

This is a maximum-minimum problem, in which the maximum of two functions or several functions is no longer smooth. Otherwise, if the primary two functions or several functions are Hölder functions, the obtained function also is a Hölder function. This property of Hölder function is the same as that of Lipschitz function.

As mentioned before, the larger the integer n is, the larger the labor of computation Bernstein α-polynomial is. So when we solve the problem

$$\max_{x\in S} \quad B_n(f,x,\alpha), \tag{24}$$

with the above algorithm, we choose $n = 3$ and $\alpha = (1.5, 1, 1.25)^T$.

For convenience in computation, we choose a rectangle $\overline{S} = \{x : x = (x_1,x_2,x_3)^T \in \mathbb{R}^3, 0 \le x_i \le 1, i = 1,2,3\} \supset S$ and solve the problem $\max_{x\in\overline{S}} f(x)$, that is, we solve the problem $\max_{x\in\overline{S}} B_n(f,x,\alpha)$ with the algorithm in which the rectangle $\overline{S}$ substitute

Table 1: Results of example 1

Sequence number	Rectangle S_m	Extreme point x^m	$\min B_m$	$f(x^m)$
0	$[0,1]^3$	$(.79622;.96929;.00000)$	.93130	.3850
1	$[0,.5][0,1]^2$	$(.49438;1.00000;.37119)$	$-.102800$	$-.3636$
2	$[.5,1][0,1]^2$	$(.50000;.99999;.50000)$	$-.101280$	$-.4000$
3	$[0,.5]^2[0,1]$	$(.50000;.50000;.28167)$	$-.00794$	1.0000
4	$[0,.5][.5,1][0,1]$	$(.49438;1.00000;.37119)$	$-.10280$	$-.3636$
5	$[.5,1][0,.5][0,1]$	$(.50000;.50000;.28167)$	$-.00790$	1.0000
6	$[.5,1]^2[0,1]$	$(.50000;1.00000;.35931)$	$-.10128$	$-.36040$
7	$[0,.5][.5,1][0,.5]$	$(.49070;1.00000;.38142)$	$-.22548$	$-.3666$
8	$[0,.5][.5,1]^2$	$(.48529;1.00000;.50000)$	$-.19528$	$-.3978$
9	$[.5,1]^2[0,.5]$	$(.50000;1.00000;.36889)$	$-.22062$	$-.3656$
10	$[.5,1]^3$	$(.50000;.50000;1.00000)$	$-.18140$	1.0000
11	$[0,.25][.5,1][0,.5]$	$(.2500;1.00000;.50000)$	$-.31322$	3.0000
…	…	……	…	…
18	$[.75,1][.5,1]^2$	$(.75000;1.00000;.50000)$	.76724	.2250
…	…	……	…	…
28	$[.5,.75][.75,1][.5,1]$	$(.5000;1.00000;.50000)$	$-.18140$	$-.4000$
…	…	……	…	…

for the simplex S in the above algorithm. Observingly, $B_n(f,x,\alpha)$ in (25) is different from that in (24). The partial processes and results are listed in the table bellow.

In table 1, x^m is the extreme point of the corresponding Bernstein α- polynomial in the rectangle S_m, $\min B_m$ is corresponding extreme value, $f(x^m)$ is the function value of $f(x)$ at the extreme point. Every segment in table 1 consist of the partitions of the reserved branches in the segment above its. We continue the process in the table 1, as the results, the optimization solution of problem

$$\max_{x\in\bar{S}} \quad f(x) \tag{25}$$

$x^* = (0.50000, 1.00000, 0.50000)^T$ and the optimization value $f(x^*) = -0.4000$ are obtained.

5 Conclusion

In our approach, we give a algorithm for solving the nonsmooth programming. Except the usual strategy of branch and bound, we mainly utilize on the approximation properties of Hölder function by Bernstein α- polynomial. The dwindling of the region substitute the heightening of the degree of the polynomials in the approximation. This technique largely cut down on the labor of the computation. The convergence of the algorithm is guaranteed in theory and the feasibility of algorithm is validated from the example. At the end of the paper, we point out there are two difficulties while solving a problem with the algorithm, one is the choosing of the parameter α in polynomial $B_n(f,x,\alpha)$, another is the solving of the subproblems $\min_{x\in P_{mj}} B_n(f,x,\alpha)$, though we solve it as a geometric programming [15,16]. Therefore, this study will be continued.

References

[1] Z. C. Xuan and P. G. Shao. A programming approach for nonsmooth structural optimization. *Advances in Engineering Software*, 31(2), 75–81, 2000.

[2] A. J. V. Brandão, M. A. Rojas-Medar and G. N. Silva. Nonsmooth continuous-time optimization problems: necessary conditions. *Computers and Mathematics with Applications*, 41(12), 1477–1486, 2001.

[3] Marco Castellani. A necessary second-order optimality condition in nonsmooth mathematical programming. *Operations Research Letters*, 19(2), 79–86, 1996.

[4] José Mario Martínez and Antonio Carlos Moretti. A trust region method for minimization of nonsmooth functions with linear constraints. *Mathematical Programming*, 76(3), 431–449, 1997.

[5] Peiping Shen, Kecun Zhang, Yanjun Wang. Applications of interval arithmetic in non-smooth global optimization. *Applied Mathematics and Computation*, 144(2–3), 413–431, 2003.

[6] P. E. Gill, W. Murray and M. A. Saunders. SNOPT: An SQP algorithm for large scale constrained optimization. *SIAM Journal on Optimization*, 12, 979–1006, 2002.

[7] John E. Dennis Jr, Shou-Bai B. Li and Richard A. Tapia. A unified approach to global convergence of trust region methods for nonsmooth optimization. *Mathematical Programming*, 68(3), 319–346, 1995.

[8] Claude Lemaréchal, Arkadii Nemirovskii, Yurii Nesterov. New variants of bundle methods. *Mathematical Programming*, 69(1), 111–147, 1995.

[9] M. J. Kontoleon, D. N. Kaziolas, M. D. Zygomalas and C. C. Baniotopoulos. Analysis of steel bolted connections by means of a nonsmooth optimization procedure. *Computers and Structures*, 81(26–27), 2455–2465, 2003.

[10] A. E. Csallner. Lipschitz continuity and the termination of interval methods for global optimization. *Computers and Mathematics with Applications*, 42(8–9), 1035–1042, 2001.

[11] Ya. I. Alber, A. N. Iusem and M. V. Solodov. On the projected subgradient method for nonsmooth convex optimization in a Hilbert space. *Mathematical Programming*, 81(1), 23–35, 1998.

[12] Stéphane Seuret and Jacques Lévy Véhel. The local Hölder function of a continuous function. *Applied and Computational Harmonic Analysis*, 13(3), 263–276, 2002.

[13] Zai-En Hou and Ke-Cun Zhang. Approximation of multivariable function by using new multivariable Bernstein α-polynomials. *Applied Mathematics and Computation*, 154(2), 335–345, 2004.

[14] R. Horst, P. M. Pardalos and N. V. Thoai. *Introduction to Global Optimization*, Second Edition, Kluwer Academic Publishers, 1–368, 2000.

[15] J. Rajgopa and D. L. Bricker. Solving posynomial geometric programming problems via generalized linear programming. *Computational Optimization and Application*, 21, 95–100, 2002.

[16] Y. J. Wang, K. C. Zhang and P. P. Shen. A new type of condensation curvilinear path algorithm for unconstrained generalized geometric programming. *Mathematical and Computer Modelling*, 35(11–12), 1209–1219, 2002.