Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computers and Structures 83 (2005) 1991–2002

Computers & Structures

www.elsevier.com/locate/compstruc

Development of the unlocking mechanisms for the complex method

Ssu-yuan Hu, Jung-Ho Cheng *

Department of Mechanical Engineering, National Taiwan University No. 1 Roosevelt Rd. Sec. 4, Taipei 106, Taiwan, ROC

Received 13 February 2004; accepted 15 February 2005
Available online 19 April 2005

Abstract

The locking phenomenon, or the degeneracy, has long been the potential defectiveness of the Complex Method. A locked optimization process leads to not only unreliable optimum but waste of time. In this paper, a modified Complex algorithm is suggested and assessed by four test functions. The results showed that the improved Complex Method equipped with the new unlocking mechanisms demonstrates excellent abilities to avoid locking phenomenon. Besides, it reduces the time of processing to about 80% to reach the optimum with the same accuracy. Moreover, a practical example of a common engineering problem is also given to show its applicability.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Optimization; Complex Method; Raplex Method; Degeneracy; Locking phenomenon

1. Introduction

Since the Complex Method utilizes direct search rather than gradient search, it has been extensively applied and continuously improved to optimize non-analytical functions, such as experimental optimizations [1]. After decades of development, this method has been comprehensively adopted in various fields of biology [2], geology [3], electronics [4], chemistry, chemical engineering [5,6], material engineering [7], parallel processing and communication [8], numerical control and information processing [9]. Besides, the "simple elegance" of this method also shows great compatibility of combination with other algorithms, like the Evolutionary Algorithm [10], Genetic Algorithm [2,11] and Game Theory [12]. However, the defectiveness of the Complex Method, known as the locking phenomenon or the degeneracy, might emerge abruptly during the search process and interrupt the optimization. According to the algorithm of the Complex Method, an optimization process that has fallen into a locked position can neither be resumed nor redeemed, and all previous experiments or evaluations are nullified. This is quite a waste of time and resources, and a variety of modifications against this shortcoming have been proposed by researchers. However, it is already confirmed by Keefer that most of them introduced complications into what is basically a very simple procedure without significantly improving the efficiency [13]. Only few of them have been proved effective to ameliorate the situation.

Van der Wiel introduced a symmetry restriction to control the degeneracy of the simplex to prevent it from repeatedly making steps that will create an elongated

* Corresponding author. Tel.: +886 2 2362 2191; fax: +886 2 2363 1755.
E-mail address: jhcheng@ntu.edu.tw (J.-H. Cheng).

and degenerate simplex [14]. A few years later, Hedlund adopted the idea and used degeneracy constraints to improve the converging ability of the Simplex Method [15]. Simply applying constraints to avoid locking might seem straightforward. However, extra constraints not only retard the convergence by complicating the process, but sometimes yield inaccurate results.

To solve all the problems above, the aim of this paper is to develop efficient and reliable algorithms to avoid locking phenomenon. The improved method will be assessed by common test functions and compared with the original Complex Method to demonstrate its feasibility.

2. Theories

The predecessor of the Complex Method is the Simplex Method, which was invented by Spendley et al. in 1962 [16]. Later in 1965, Box devised his own method to deal with constrained optimization problems, and called it the Complex Method [17]. However, essentially it is just a modification of the Extended Simplex Method proposed by Nelder and Mead in 1965 [18].

2.1. Complex method

When using the Complex Method to solve an optimization problem of N variables, an initial simplex with $2N$ nodes, or vertices, must first be constructed. Each of the nodes represents a set of design variables, either specified by user or generated randomly without violating constraint equations. All nodes will be evaluated for their objective function values and accordingly sorted, and then the iteration begins.

The iterative procedure of the Complex Method can be simplified into only two actions: expansion and contraction. The node with the largest function value, or the "worst" node, will be abandoned, and then a new node is located away from the old node with respect to the centroid of other nodes with a reflection factor of 1.3, as suggested by Box. This procedure is called expansion, because the reflection factor is usually larger than one.

After the expansion, the new node must be checked for its validity. If it violates any constraint equations, or its function value is still the largest, suggesting that the new node is not better than the old one, a midway node will be located to replace the new node with a reflection factor of 0.5, as suggested by Box. Because the midway node is located backwards, this procedure is thus called contraction. The simplex keeps contracting until the new node fulfills all constraint equations and possesses smaller function value than the old node. This makes a whole iteration and generates a new simplex with one node modified.

Then the new simplex is expanded again, and the iterative process goes over and over until all nodes occupy almost the same location, and the simplex shrinks to one point. This point represents the best node that can be found, i.e., the optimum [19].

2.2. Locking phenomenon and degeneracy

When dealing with smooth objective functions, the Complex Method seldom fails. But for some special cases, such as the well-known Rosenbrock's function, there is a great chance that the worst node of the simplex gets trapped in particular locations and loses the abilities to expand or contract. Because this was not anticipated by the original algorithm, the worst node will be forced to contract over and over again, creating an infinite loop, or so-called locking phenomenon.

The most common reason of such case is that the worst node coincides with the centroid of other nodes. Since the worst node is reflected with respect to itself, the simplex can never change its shape. Because such a simplex usually has a slender or distorted shape, the locking phenomenon is also designated as a degeneracy.

3. Improvement

Here we demonstrate a typical example of the locking phenomenon. By observing the history of the iterative process step by step, we hope to reveal the crux that leads to the locking phenomenon.

We choose the well-known Rosenbrock's function $f(x,y) = 100(y - x^2)^2 + (1 - x)^2$ as the objective function, with four initial nodes $(0,0)$, $(-3.63, 23.2)$, $(-20.55, 19.0)$, $(-25, 25)$. The design space is $-25 \leqslant x \leqslant 25$, $-25 \leqslant y \leqslant 25$, and the optimum is 0 at $(1,1)$. The iterative history is shown in Table 1.

It is clear that finally the worst node coincided with the centroid of other nodes in the last iteration caused the locking phenomenon. In closer observation we noticed three things that might help to eliminate the locking phenomenon.

3.1. Pre-locking determination

The first thing we noticed is that successive contractions augur the locking phenomenon. From iterations 1–8, every node gets no more than two contractions. But then node 3 gets five contractions in iteration 9 and soon it is locked. Gustavsson and Sundkvist once concluded in their work that repeated failed contractions had to be minimized in order to prevent false convergence [20]. It suggests that successive contractions on a single node gives the warning that the locking is about to occur. We call this a pre-locking state.

The determination of the locking occurrence is a converging process, and therefore requires considerable evaluations of the objective function. Wasting so many

Table 1
Iterative history of a locking phenomenon

Iter. No.	Node 1		Node 2		Node 3		Node 4		No. of contractions
	x	y	x	y	x	y	x	y	
0	0	0	−20.55	19.0	−3.63	23.2	−25	25	
1	↓	↓	↓	↓	↓	↓	13.96	−0.15	0
2	↓	↓	14.22	0.33	↓	↓	↓	↓	1
3	↓	↓	−10.6	17.25	↓	↓	↓	↓	0
4	↓	↓	↓	↓	↓	↓	−9.8	16.4	2
5	↓	↓	3.44	7.9	↓	↓	↓	↓	0
6	↓	↓	↓	↓	↓	↓	6.27	6.5	1
7	↓	↓	↓	↓	↓	↓	−4.18	12.9	1
8	↓	↓	↓	↓	1.96	−3.6	↓	↓	1
9	↓	↓	↓	↓	−0.33	7.37	↓	↓	5
10	↓	↓	↓	↓	−0.13	6.38	↓	↓	0
11	↓	↓	↓	↓	**−0.24**	**6.94**	↓	↓	33

resources just to declare failure of the process is obviously absurd. Therefore, the pre-locking warning becomes very important. It tells us that if we want to do something to prevent the locking phenomenon, we should do it right away. According to our experiments, it is recommended that five successive contractions are sufficient to be determined as a pre-locking state.

3.2. Abdication

Abdication is the main unlocking mechanism we have developed to prevent the process from locking. The idea is simple. When the worst node of the simplex is in pre-locking state, we judge it improper to be reflected in this iteration. Naturally, the second worst node makes the best candidate to substitute for the worst node, for it will become the new worst node one way or the other. Since the worst node gives way to the priority of expansion, we termed this mechanism “abdication”.

Although abdication somewhat violates the basic concept of the Complex Method, there are two reasons that make it permissible. One reason is that expanding the second worst node helps to change the shape of the simplex. After the second worst node expands successfully, the real worst node will no longer be on the centroid of the other nodes, because one of them has moved. The other reason is that we assume the worst node and the second worst node will expand toward almost the same direction with or without abdication, because at least they are always reflected towards the better nodes. In short, abdication helps to reshape the simplex for easier expansion without altering the converging direction, and thus prevents locking.

In some special cases, the second worst node is also in the pre-locking state, while the third worst node must carry on. This may keep going until all worse nodes get in pre-locking state, and abdication fails to unlock the situation. When this happens, there are two ways to resume the operation. One is to increase the number of contractions for pre-locking criteria, but this does not always work. The other is to add extra nodes to the simplex, and this will be discussed in the next section.

3.3. Adjustable number of nodes

Nelder and Mead recommended $N + 1$ nodes for an N-variable optimization, while Box recommended $2N$. However, there is no reasonable explanation why they decided thus. According to our experiments, a $2N$-node simplex leads to more failed processes, while an $N + 1$-node simplex results in more inaccurate optima (Tables 2–5). This implies that too many nodes might interfere with one another and obstruct the convergence, while insufficient nodes may lack the ability to cover the whole design space, and thus generate an inaccurate optimum.

We adopted a compromised means to determine the number of nodes of a simplex. Based on our experiments, an initial simplex of $1.5N$ nodes seems an adequate choice. Besides, an extra random node is added to the simplex when all worse nodes within the simplex are in pre-locking state. This may keep going until the total number of nodes reaches $2N$. If all worse nodes within a $2N$-node simplex are in pre-locking state, the whole process is determined as failed.

The idea of an adjustable number of nodes is motivated by two assumptions. First, too many nodes require more time of evaluation, so less nodes seem more efficient, and extra nodes are added only when needed. Second, a newly added node not only makes a failed convergence resumable, but helps to find new search directions since it is not involved in previous iterations.

Table 2
Comparison of different algorithms tested by Powell's function

	Algorithm C	Eight initial nodes		Algorithm R	Six initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	0	0	0	0	0	0
Failed	0	0	0	0	0	0
S.D.	55.5	73.8	95.7	76.7	84.9	88.5
Avg.	674.3	904.6	1343.8	554.7	720.8	1068.6
Eff_U	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Eff_E	100.0%	100.0%	100.0%	111.7%	117.8%	122.2%
Eff	100.0%	100.0%	100.0%	111.7%	117.8%	122.2%
	Algorithm R_{2N}	Eight initial nodes		Algorithm R_{N+1}	Five initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	0	0	0	0	0	1
Failed	0	0	0	0	0	0
S.D.	69.8	70.4	83.5	83.6	95.8	116.0
Avg.	676.3	894.4	1329.2	511.7	664.2	987.6
Eff_U	100.0%	100.0%	100.0%	100.0%	100.0%	99.0%
Eff_E	95.7%	101.9%	103.8%	118.7%	124.3%	125.4%
Eff	95.7%	101.9%	103.8%	118.7%	124.3%	124.1%

Table 3
Comparison of different algorithms tested by Rosenbrock's function

	Algorithm C	Four initial nodes		Algorithm R	Three initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	1	0	0	3	2	1
Failed	47	55	43	3	2	3
S.D.	27.1	28.5	41.0	29.3	35.1	48.9
Avg.	225.4	265.4	357.7	162.7	204.4	283.4
Eff_U	52.0%	45.0%	57.0%	94.0%	96.0%	96.0%
Eff_E	100.0%	100.0%	100.0%	129.4%	120.1%	117.7%
Eff	52.0%	45.0%	57.0%	121.6%	115.3%	113.0%
	Algorithm R_{2N}	Four initial nodes		Algorithm R_{N+1}	Three initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	0	0	0	8	0	2
Failed	5	8	8	0	2	3
S.D.	33.0	32.0	39.6	31.4	30.4	36.1
Avg.	218.5	267.0	344.3	169.5	201.5	271.2
Eff_U	95.0%	92.0%	92.0%	92.0%	98.0%	95.0%
Eff_E	99.0%	97.3%	103.8%	123.7%	124.1%	128.3%
Eff	94.1%	89.5%	95.5%	113.8%	121.6%	121.8%

R and R_{N+1} begin with the same number of nodes, so actually they are the same algorithm for this function.

3.4. The improved algorithm

All the unlocking mechanisms mentioned above were combined as one final algorithm, and a flow chart of the whole process is shown in Fig. 1. The three rhombi at right represent the pre-locking warning, abdication, and extra nodes, respectively. This final algorithm will be referred to as the Raplex Method hereafter, and Visual Basic 6.0 is adopted as the programming environment.

Table 4
Comparison of different algorithms tested by Fletcher's function

	Algorithm C	Six initial nodes		Algorithm R	Five initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	0	0	0	1	1	0
Failed	41	36	38	1	3	1
S.D.	108.6	98.9	116.9	120.7	118.4	102.1
Avg.	501.7	572.5	747.6	424.5	507.7	649.1
Eff_U	59.0%	64.0%	62.0%	98.0%	96.0%	99.0%
Eff_E	100.0%	100.0%	100.0%	112.6%	106.9%	115.0%
Eff	59.0%	64.0%	62.0%	110.3%	102.6%	113.9%
	Algorithm R_{2N}	Six initial nodes		Algorithm R_{N+1}	Four initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	0	0	0	5	1	0
Failed	0	2	3	3	1	2
S.D.	119.1	133.8	128.4	122.1	107.7	123.9
Avg.	518.2	598.7	756.5	361.4	439.5	565.8
Eff_U	100.0%	98.0%	97.0%	92.0%	98.0%	98.0%
Eff_E	95.7%	91.3%	97.3%	128.9%	122.6%	124.6%
Eff	95.7%	89.5%	94.3%	118.5%	120.1%	122.1%

Table 5
Comparison of different algorithms tested by Wood's function

	Algorithm C	Eight initial nodes		Algorithm R	Six initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	6	0	0	20	3	0
Failed	9	13	10	0	0	0
S.D.	245.2	279.7	279.0	174.0	245.0	240.7
Avg.	961.5	1100.5	1362.8	760.9	901.1	1144.2
Eff_U	85.0%	87.0%	90.0%	80.0%	97.0%	100.0%
Eff_E	100.0%	100.0%	100.0%	129.3%	120.5%	118.5%
Eff	85.0%	87.0%	90.0%	103.4%	116.9%	118.5%
	Algorithm R_{2N}	Eight initial nodes		Algorithm R_{N+1}	Five initial nodes	
Accuracy	10^{-2}	10^{-3}	10^{-5}	10^{-2}	10^{-3}	10^{-5}
Inaccurate	5	1	0	45	14	0
Failed	2	0	0	0	0	0
S.D.	254.8	269.6	288.6	185.1	174.0	218.4
Avg.	954.2	1082.2	1369.6	699.6	782.5	1036.0
Eff_U	93.0%	99.0%	100.0%	55.0%	86.0%	100.0%
Eff_E	99.9%	102.1%	98.9%	136.4%	144.7%	130.8%
Eff	92.9%	101.1%	98.9%	75.0%	124.4%	130.8%

4. Results and discussion

Our modifications were assessed by some well-known test functions to examine their reliabilities and efficiencies. Moreover, the Raplex Method was compared with unlocking mechanisms previously suggested by other researchers to manifest its outstanding applicability.

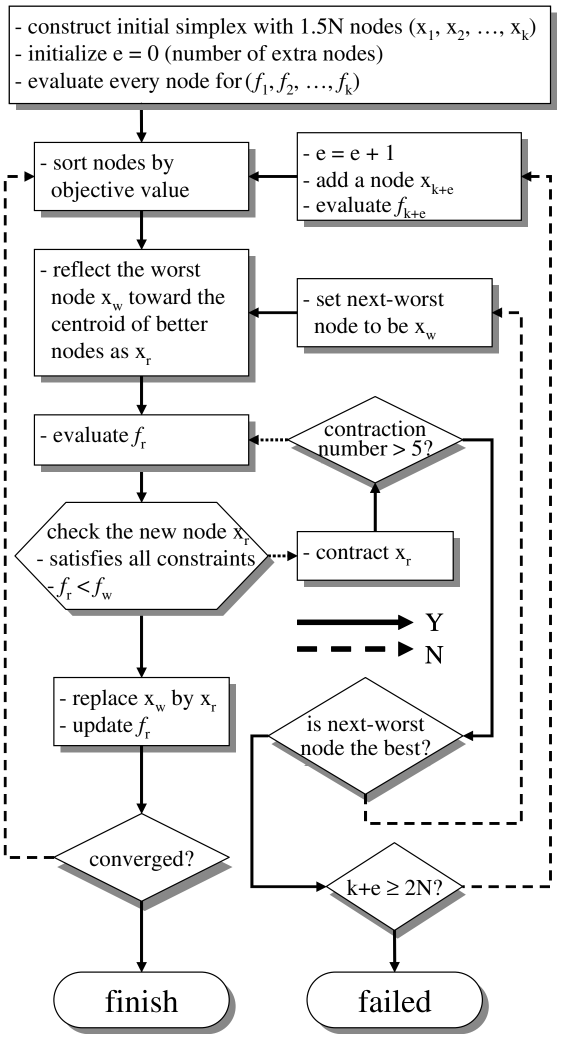

- construct initial simplex with 1.5N nodes (x1, x2, …, xk)
- initialize e = 0 (number of extra nodes)
- evaluate every node for (f1, f2, …, fk)
- sort nodes by objective value
- e = e + 1
- add a node xk+e
- evaluate fk+e
- reflect the worst node xw toward the centroid of better nodes as xr
- set next-worst node to be xw
- evaluate fr
contraction number > 5?
check the new node xr
- satisfies all constraints
- fr < fw
- contract xr
Y
N
- replace xw by xr
- update fr
is next-worst node the best?
converged?
k+e ≥ 2N?
finish
failed

Fig. 1. Flow chart of the Raplex Method.

4.1. Benchmark functions

We choose the following four test functions, which are suggested by brilliant mathematicians and have long been adopted to assess optimization algorithms [21].

1. Powell's function:
   $\text{Obj} = (x + 10y)^2 + 5(z - w)^2 + (y - 2z)^4$
   $+ 10(x - w)^4$
   region: $-100 \leqslant x,y,z,w \leqslant 100$
   optimum: 0 at $(0,0,0,0)$
   Powell's function is a relatively simple function. Even the original Complex Method can solve it correctly.
2. Rosenbrock's function:
   $\text{Obj} = 100(y - x^2)^2 + (1 - x)^2$
   region: $-25 \leqslant x,y \leqslant 25$
   optimum: 0 at $(1,1)$
   Complex Method often fails to solve the Rosenbrock's function, and gets locked on these two pseudo-minima: 1 at $(0,0)$ and 0.25 at $(0.5,0.25)$.
3. Fletcher's function:
   $\text{Obj} = 100(z - 10\theta(x,y))^2 + (\sqrt{x^2+y^2} - 1)^2 + z^2$
   where $\theta(x,y) = \frac{1}{2\pi}\tan^{-1}(\frac{y}{x})$
   region: $-100 \leqslant x,y,z \leqslant 100$
   optimum: 0 at $(1,0,0)$
   Fletcher's function is also hard to solve because the function surface is a steep helical valley.
4. Wood's function:
   $\text{Obj} = 100(y - x^2)^2 + (1 - x)^2 + 90(w - z^2)^2$
   $+ (1 - z)^2 + 10.1((y - 1)^2 + (w - 1)^2)$
   $+ 19.8(y - 1)(w - 1)$
   region: $-100 \leqslant x,y,z,w \leqslant 100$
   optimum: 0 at $(1,1,1,1)$
   Wood's function is another tricky function, while the $(y - x^2)$ and $(w - z^2)$ terms also induce $(-1,1,-1,1)$ rather than the real optimum.

4.2. Efficiency Indices

Before the assessment begins, we want to establish two indices as the reference of efficiency: the Evaluation Efficiency and the Unlocking Efficiency.

Many people take the number of converging iterations as the index of optimization efficiency, but this is not always adequate. In experimental optimizations, for example, each evaluation of the objective function requires one or more field experiments to be performed [6]. In such cases, the less experiments are needed to be carried out, the more time will be saved. Therefore, the Evaluation Efficiency is defined as the normalization with respect to the original Complex Method, counted within certain number of optimizations:

$$\text{Eff}_\text{E} = w_1\frac{\text{Avg}_0}{\text{Avg}} + w_2\frac{\text{SD}_0}{\text{SD}} \qquad (1)$$

where w_1, w_2 are weighting factors, Avg is the average value of the number of evaluations, and Avg_0 is that of the original Complex Method. Similarly, SD represents the standard deviation of the number of evaluations, and SD_0 is that of the Complex Method.

We choose to observe not only the average number of evaluations but also their standard deviation because we need the algorithm to perform steadily. If some algorithm takes the fewest evaluations to complete a process but with high standard deviation, it implies that this rarely occurs, and that is not what we want. In this paper, we choose $w_1 = 80\%$ and $w_2 = 20\%$.

The Unlocking Efficiency is simply defined as

$$\text{Eff}_\text{U} = 1 - \frac{F + I}{T} \qquad (2)$$

where F is the number of failed optimizations, and I is the number of optimizations that yield inaccurate results. In this paper, a result is defined as an inaccurate one if the difference between the resulted optimum and the exact optimum exceeds 50 times the convergence criteria. T represents the total number of optimizations.

Finally we define the total efficiency as the product of the two indices:

$$\mathrm{Eff} = \mathrm{Eff}_{\mathrm{U}} \cdot \mathrm{Eff}_{\mathrm{E}} \tag{3}$$

which is much stricter definition than the sum or the average value of those two indices.

4.3. Comparison between the R-series algorithms and the complex method

Four different algorithms will be assessed together for comparison: the original Complex Method, the proposed Raplex Method, and the Raplex Method initialized with $2N$ nodes and $N + 1$ nodes.

The procedure of the assessment is as follows. Each test function will be optimized to specific accuracy by every different algorithm for 100 times. The accuracy is controlled by the convergence criteria, which is set to 10^{-2}, 10^{-3}, and 10^{-5}. The converging status and number of evaluations will be recorded and calculated for their average value and standard deviation for further analyses. All the results are listed in Tables 2–5. Different algorithms are referred to as C, R, $\mathrm{R}_{2\mathrm{N}}$, $\mathrm{R}_{\mathrm{N+1}}$ for the original Complex Method, the Raplex Method, and the Raplex Method initialized with $2N$ nodes and $N + 1$ nodes, respectively.

The success of an optimization process is dependent on the ability of the algorithm to locate the optimum not only correctly but efficiently [15], and this can be verified by checking the Unlocking Efficiency and the Evaluation Efficiency.

First we examine the unlocking ability. From Table 3–5 it is clear that there is a 9–55% chance that the Complex Method fails to complete an optimization process, while our modifications reduce the rates to only 0–3%. The Unlocking Efficiencies of the R, $\mathrm{R}_{2\mathrm{N}}$ and $\mathrm{R}_{\mathrm{N+1}}$ algorithms are undoubtedly much higher than that of the Complex Method. This suggests that the abdication mechanism does possess good abilities of unlocking.

Then we check the Evaluation Efficiency. Because the pre-locking detection deflects the searching direction before it falls into a tedious locking convergence, and an adjustable number of nodes avoids unnecessary evaluations, these two unlocking mechanisms are also supposed to reduce the number of evaluations of the process. Just as expected, the R-series algorithms exceed the Complex Method in Evaluation Efficiency by approximately 20–30%.

If we take a closer look at the results, we can find that there is no particular best algorithm for all situations. The $\mathrm{R}_{\mathrm{N+1}}$ algorithm has better Evaluation Efficiency because it does not have too many nodes to evaluate. However, it also induces more inaccurate optima than other algorithms. The $\mathrm{R}_{2\mathrm{N}}$ algorithm has the best Unlocking Efficiency on Wood's function, but performs worse than any other R-series algorithm on Rosenbrock's function. The R algorithm seems efficient for most of the cases, especially when the accuracy demand is high. However, if the problem is relatively simpler, the $\mathrm{R}_{\mathrm{N+1}}$ algorithm might perform more efficiently than the R algorithm. These all suggest that the initial number of nodes should be determined not only by the number of variables but also by the complexity of the problem. Therefore, it is advised that this value should be open for users to modify when translating this algorithm into computer programs. Basically, we recommend $1.5N$ as a modest default for an average performance.

4.4. Comparison between the Raplex Method and other unlocking mechanisms

The Raplex Method is compared with other improvements proposed in [14,15]. The main unlocking mechanism in [14] is Symmetry Control, and that in [15] is Degeneracy Constraint.

Van der Wiel proposed a measure of simplex distortion termed "Symmetry", which is defined by the Nth root of the absolute value of the determinant of the simplex divided by the radius of the sphere passing through all vertices of the simplex, where N represents the number of variables. If the Symmetry of the simplex is below a preset criterion, the simplex will be reshaped until it meets the Symmetry criterion [14]. Although it is mentioned that the applicable number of variables range from 2 to 8, no experiments were demonstrated to show that this mechanism works for $N > 2$, probably due to the limitations of computing capacities 20 years ago. It should be noticed that the calculation of the matrix determinant and the circumcenter of the simplex requires considerable resources of CPU, especially when N is large. Besides, this mechanism is inapplicable when $N = 1$.

Hedlund proposed a relatively simpler measure of degeneracy, which is the minimum angle ratio of the smallest to largest angles of the simplex ($A_{\min}/A_{\max}$). When a simplex was regarded as too degenerate, the optimization was continued with a simplex that was regularly rescaled, retaining the previous best vertex, the best gradient direction and the area of the simplex [15]. It should come to one's attention that the term "area" implies that this mechanism is developed for 2-variable problems only. Besides, all experiments conducted in [15] are also of 2-variable test functions, which concur with the conjecture. The reason why N must not exceed 3 could be that an angle between two vectors in hyperspace has little physical meaning. In addition, this

Table 6
Comparison between the Raplex Method and other unlocking mechanisms

	Raplex Method		Symmetry Control		Degeneracy Constraint	
Accuracy	10^{-2}	10^{-5}	10^{-2}	10^{-5}	10^{-2}	10^{-5}
Rosenbrock function $(N = 2)$						
Activated						
Count	38	59	100	100	53	39
Inaccurate	1	1	7	0	49	0
Failed	3	3	17	21	0	0
Overall						
Inaccurate	3	1	7	0	50	0
Failed	3	3	17	21	0	0
S.D.	29.3	48.9	29.5	30.0	29.4	76.0
Avg.	162.7	283.4	147.4	237.1	142.2	311.9
Eff_U	94.0%	96.0%	76.0%	79.0%	50.0%	100.0%
Eff_E	129.4%	117.7%	140.8%	148.0%	145.4%	102.6%
Eff	121.6%	113.0%	107.0%	116.9%	72.7%	102.6%
Fletcher function $(N = 3)$						
Activated						
Count	53	53	100	100	2	1
Inaccurate	1	0	0	1	2	1
Failed	1	1	80	73	0	0
Overall						
Inaccurate	1	0	0	1	3	1
Failed	1	1	80	73	44	41
S.D.	120.7	102.1	82.5	127.3	98.3	99.0
Avg.	424.5	649.1	319.5	495.0	324.6	551.3
Eff_U	98.0%	99.0%	20.0%	26.0%	53.0%	58.0%
Eff_E	112.6%	115.0%	152.0%	139.2%	145.7%	132.1%
Eff	110.3%	113.9%	30.4%	36.2%	77.2%	76.6%
Wood function $(N = 4)$						
Activated						
Count	24	7	100	100	0	0
Inaccurate	3	0	10	0	0	0
Failed	0	0	82	94	0	0
Overall						
Inaccurate	20	0	10	0	30	0
Failed	0	0	82	94	16	8
S.D.	174.0	240.7	171.2	175.5	227.3	219.0
Avg.	760.9	1144.2	380.7	667.2	514.8	1029.7
Eff_U	80.0%	100.0%	8.0%	6.0%	54.0%	92.0%
Eff_E	129.3%	118.5%	230.7%	195.2%	171.0%	131.4%
Eff	103.4%	118.5%	18.5%	11.7%	92.3%	120.8%

mechanism also requires considerable computing resources when N is large, since the rescaling of the simplex requires complete update of evaluations of every simplex nodes except the best one.

The assessing procedures in the previous section were adopted again for these two mechanisms, and the results are listed in Table 6. Besides the previously defined indices, the number of optimizations in which the unlocking mechanism is activated are also listed, along with the number of corresponding inaccurate or failed results. This helps to identify the connections between the source of unreliable results and the unlocking mechanisms.

First it should come to one's notice that both the Symmetry Control and the Degeneracy Constraint show poor Unlocking Efficiencies (mostly 6–58%) for the

Fletcher and the Wood functions, implying that they are applicable only when $N = 2$. Moreover, the unreliable results is of different sources. For the Symmetry Control, the number of unreliable results from the optimizations which activates the unlocking mechanism agrees with the total number of unreliable results. This suggests that Symmetry Control is quite alert to the degeneracy of the simplex even though reshaping of the simplex is ineffective. However, the number of unreliable results from the optimizations which activates the Degeneracy Constraint does not match the total number of unreliable results at all, implying that this mechanism is unsuitable for $N > 2$ cases because it cannot even detect that the locking is happening, not to mention unlock it.

On the other hand, the Evaluation Efficiencies tend to substantially favor Symmetry Control and Degeneracy Constraint rather than the Raplex Method for the Fletcher and the Wood functions. However, this is because the Raplex Method begins with $1.5N$ nodes, while the other two mechanisms begin with $N + 1$ nodes. Consequently the number of evaluations required by the Raplex Method is doubtlessly larger when $N > 2$. Even so, the overall efficiencies of the Raplex Method are still averagely higher than other two mechanisms, because of its steady and reliable unlocking abilities.

As for the 2-variable Rosenbrock's function, Symmetry Control does exhibit abilities to unlock the degeneracy as compared to the Complex Method, but the Unlocking Efficiencies are still much lower than those of the Raplex Method. On the other hand, Degeneracy Constraint successfully unlocked all of the 100 tests. However, half of them yields inaccurate results when the accuracy demand is low, so the average Unlocking Efficiency is still lower than that of the Raplex Method, implying that Degeneracy Constraint works selectively to the convergence criterion.

4.5. Summary

Basically Symmetry Control and Degeneracy Constraint have similar deficiencies. They are both unsuitable for $N > 2$ cases, and require extra computer resources while calculating the measure of geometric distortion of the simplex. On the contrary, the Raplex Method has no such limitations at all, since it forecasts the occurrence of degeneracy long before the locking happens by monitoring the iterative history, rather than exhausting additional resources to detect and reduce the degeneracies, as the other two mechanisms do.

Overall, the Raplex Method not only prevents the locking phenomenon more effectively than other discussed mechanisms, but saves about 20–30% of the time required by the Complex Method. This makes it a successful modification to the original Complex algorithm.

5. Application

Deep drawing of sheet metal is an important manufacturing technique, and the selection of working conditions has always been complicated and tricky. Numerous techniques such as the Evolutionary Structural Optimization (ESO), Inverse Approach (IA), and the Large Time Incremental (LATIN) method have been proposed for optimization of the parameters [22–24]. Since manual optimization requires the competence of qualified engineers during a long period, and is thus unreliable and costly [25], we tried to translate the process into an optimization problem, where the history of drawing can be automatically simulated by finite element analysis. Then we can solve the problem with both the Complex Method and the Raplex Method, and determine from the results whether the Raplex Method is more capable of dealing with practical engineering problems than the Complex Method.

5.1. Description

A typical deep drawing process of a cylindrical cup usually consists of one piece of sheet metal, a punch, a die, and a holding clamp. The geometry and mesh for this problem are illustrated in Fig. 2 [26]. R_B, R_D, R_P and R_C represent the radii of the blank, the die, the punch, and the clamp, respectively, while r_D and r_P stand for the radii of the fillet of the die and the punch. R_B, R_D, and R_C have already been chosen, while R_P, r_D and r_P remain to be decided.

To obtain a successful deep drawing process, it is essential to control the slip between the blank, the clamp, and the die. If the slip is restrained too strongly, the material will undergo severe stretching, thus potentially causing necking and rupture. If the blank can slide

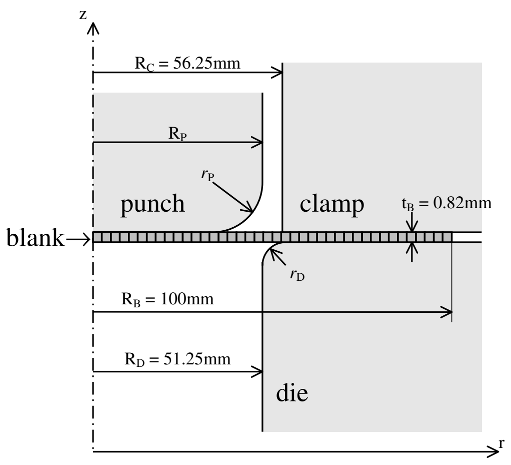


Fig. 2. Geometry and mesh for a deep drawing problem.

too easily, the material will be drawn in completely and high compressive circumferential stresses will develop, causing wrinkling in the product. Therefore, the friction coefficients between those parts are also to be optimized. They are referred to as f_D, f_C and f_P standing for the friction coefficients between the blank and the die, the blank and the clamp, the blank and the punch, respectively. That makes a total of six variables to be tuned for this deep drawing problem. Although some may argue that friction coefficients are not to be manipulated, the intention of this example is to reveal the optimized configuration for the process. Practically it can be achieved by selection of blank materials and lubricants to create conditions similar to the optimized configuration.

Since relatively small thickness changes are usually desired in deep drawing processes, we decided to choose the absolute value of the simulated maximum compressive strain in the thickness direction as the objective function of this optimization problem. Thus the optimization problem can be described as

$$\text{minimize} \quad f_{\text{obj}} = f(R_P, r_D, r_P, f_D, f_C, f_P)$$

subject to

$$45.0 < R_P < 55.0$$
$$3.0 < r_D < 25.0$$
$$3.0 < r_P < 25.0$$
$$0.01 < f_D < 0.5$$
$$0.01 < f_C < 0.5$$
$$0.15 < f_P < 0.5$$

with initial values $R_P = 50.0$, $r_D = 13.0$, $r_P = 5.0$, $f_D = 0.1$, $f_C = 0.1$ and $f_P = 0.25$, where the initial values of the variables are directly adopted from the configuration in [26], and the constraints represent the design space, which is decided by choosing a reasonable range around the given initial values.

5.2. Finite element modelling

The finite element analyses are conducted by the commercial software ABAQUS 6.3-1. Since the target shape is a cylindrical cup, the blank is modelled with 4-node bilinear axisymmetric element with reduced integration (CAX4R). The punch, the die, and the clamp are all modelled as rigid bodies, and the material of the blank (aluminum-killed steel) is assumed to satisfy the Ramberg–Osgood relation between true stress and logarithmic strain:

$$\varepsilon = \left(\frac{\sigma}{K}\right)^{1/n} \tag{4}$$

where the reference stress value K is 513 MPa, and the work-hardening exponent n is 0.223. The Young's modulus is 211 GPa, and the Poisson's ratio is 0.3.

The procedure for an evaluation is simple. Whenever an evaluation is requested by the optimizing algorithm, values of a set of parameters R_P, r_D, r_P, f_D, f_C and f_P should be given, and an input file of corresponding mesh and boundary conditions for a finite element analysis can be generated by filling the blanks reserved in a template file with given values. Then the FEM software ABAQUS will be executed to solve the simulation problem. The status of ABAQUS will be continuously monitored until the job is successfully completed, and the results will then be retrieved from the output files and searched for the maximum compressive strain in the direction of thickness as the objective. If the analysis fails to converge, a strain of -10 will be adopted instead as a penalty. All of the above procedures are programmed and thus automatically yield the value of evaluation for the objective function. A representative flow chart showing the flow of information among the Raplex Method, ABAQUS, and the interface program is illustrated in Fig. 3.

5.3. Results and discussion

The problem is optimized for 25 times by each algorithm with a convergence criteria of 10^{-2}, and the results are shown in Table 7. The optimized variables and the contour of the thickness strain for the optimal case located by the Raplex Method are shown in Fig. 4.

Basically both algorithms can reach an optimal compressive strain of about $-1.37\text{E}-2$. This is quite an improvement compared with the suggested configuration in [26], which yields a compressive strain of $-7.896\text{E}-2$. However, it is clear that in 19 out of 25 cases, the original Complex Method failed to converge to the specified criteria, while the Raplex Method successfully solved all of them without failure. The standard deviation of the Raplex Method for evaluation numbers

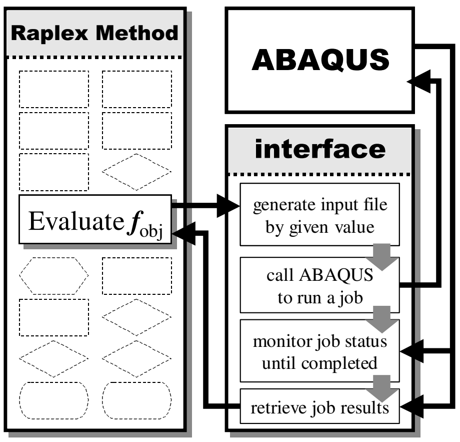
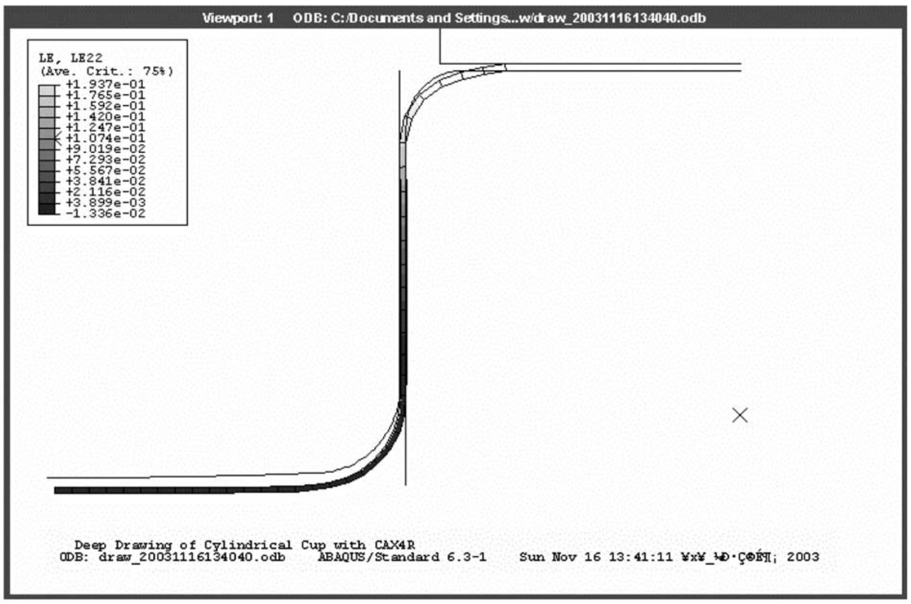

Raplex Method
Evaluate f_{obj}
ABAQUS
interface
generate input file by given value
call ABAQUS to run a job
monitor job status until completed
retrieve job results

Fig. 3. Flow chart of program interfaces.

Table 7
Comparison of different algorithms for the deep drawing problem

	Algorithm	
	Complex Method	Raplex Method
Accuracy	10^{-2}	10^{-2}
Failed	19	0
Evaluation		
S.D.	109.1	133.9
Avg.	352.8	284.1
Efficiency		
Eff_U	24.0%	100.0%
Eff_E	100.0%	115.6%
Eff	24.0%	115.6%
Best six converged objective values		
	1.3703E − 02	1.3773E − 02
	1.4152E − 02	1.4083E − 02
	1.4166E − 02	1.4098E − 02
	1.4526E − 02	1.4313E − 02
	1.6671E − 02	1.4496E − 02
	1.7988E − 02	1.4796E − 02


Viewport: 1 ODB: C:/Documents and Settings...w/draw_20031116134040.odb
LE, LE22
(Ave. Crit.: 75%)
+1.937e-01
+1.765e-01
+1.592e-01
+1.420e-01
+1.247e-01
+1.074e-01
+9.019e-02
+7.293e-02
+5.567e-02
+3.841e-02
+2.116e-02
+3.899e-03
-1.336e-02
Deep Drawing of Cylindrical Cup with CAX4R
ODB: draw_20031116134040.odb ABAQUS/Standard 6.3-1 Sun Nov 16 13:41:11 2003

$R_P^* = 50.3650707180328$ (mm), $r_P^* = 12.3792511130303$ (mm), $r_D^* = 8.3116017202614$ (mm), $f_D^* = 0.01$, $f_C^* = 0.01$, $f_P^* = 0.414896713309095$, Objective = 1.37729602099132E-02

Fig. 4. Contour of thickness strain for the optimal result.

is a little higher than that of the Complex Method, but this is because there are only six effective samples for the Complex Method. The overall efficiency of the Raplex Method is undoubtedly much higher than that of the Complex Method.

When we compare the best six results yielded by both algorithms, it is obvious that those by the Raplex Method are averagely smaller than those by the Complex Method. All six results by the Raplex Method are below 1.5E − 2, while those by the Complex Method range up to 1.8E − 2. Therefore, it can be concluded that the Raplex Method possesses not only higher unlocking efficiency, but also greater chance to locate results closer to the exact optimum.

6. Conclusions

In this paper, we developed an improved Complex Method with three unlocking mechanisms: pre-locking detection, abdication, and adjustable number of nodes. The new method is assessed by four common test functions, and the results demonstrated the following truths:

1. The new method has reliable unlocking abilities. It solves most of the cases that the original Complex Method fails to deal with.
2. The Evaluation Efficiency of the new method is averagely 25% higher than the original Complex Method, which means it needs only 80% resources to generate the same optimum.

We have shown that the new method is reliable and efficient, and thus termed it the Raplex Method for its rapidity.

References

[1] Huang Y, McColl WF. An improved simplex method for function minimization. Systems Man Cybernet 1996;3:1702–5.
[2] Yen J. A hybrid approach to modeling metabolic systems using a genetic algorithm and simplex method. IEEE Trans Syst Man Cybernet—Part B: Cybernet 1998;28(2):173–91.
[3] Vance JB, Hassani FP, Mottahed P. Improved determination of microseismic source location using a simplex technique. IEEE Trans Industry Appl 1988;24(4):666–71.
[4] Al-Mualla ME, Canagarajah CN, Bull DR. Simplex minimization for fast long-term memory motion estimation. Electron Lett 2001;37(5):290–2.
[5] Öberg T. Importance of the first design matrix in experimental simplex optimization. Chemom Intell Lab Syst 1998;44:147–51.
[6] Santos-Delgado MJ, Crespo-Corral E, Polo Díez LM. Determination of herbicides in soil samples by gas chromatography: optimization by the simplex method. Talanta 2000;53:367–77.
[7] Tsao L, Liu C. A comparison between two optimization methods on the stacking sequence of fiber-reinforced composite laminate. Comput Struct 1995;55(3):515–25.
[8] Wang D, Hu B, Peng Q, Tan Y. BSP-based parallel simplex method. Proc Fourth Int Conf/Exhibition High Perform Comput Asia-Pacific Region 2000;2:635–9.
[9] Song HY, Golomb SW. Some new constructions for simplex codes. IEEE Trans Inf Theory 1994;40(2):504–7.
[10] Smith S. The simplex method and evolutionary algorithms. The 1998 IEEE Int Conf Evolutionary Comput Proc 1998:799–804.
[11] Seront G, Bersini H. Simplex GA and hybrid methods. Proc IEEE Int Conf Evolutionary Comput 1996:845–8.
[12] Stojković NV, Stanivirомić PS. Two direct methods in linear programming. Eur J Oper Res 2001;131:417–39.
[13] Gill PE, Murray W. Numerical methods for constrained optimization. Academic Press; 1974.
[14] Van der Wiel, Maassen R, Kateman G. The symmetry-controlled simplex optimization procedure. Anal Chim Acta 1983;153:83–92.
[15] Hedlund P, Gustavsson A. Design and evaluation of an effective modified simplex method. Anal Chim Acta 1999;391:257–67.
[16] Spendley W, Hext GR, Himsworth FR. Sequential applications of simplex designs in optimization and evolutionary operation. Technometrics 1962;4:441–61.
[17] Box MJ. A new method of constrained optimization and a comparison with other methods. The Comput J 1965;8:42–52.
[18] Nelder JA, Mead R. A simplex method for function minimization. The Comput J 1965;7:308–13.
[19] Bunday BD. Basic optimisation methods. Edward Arnolds 1984:98–102.
[20] Gustavsson A, Sundkvist JE. Design and optimization of modified simplex methods. Anal Chim Acta 1985;167:1–10.
[21] Hedlund P, Gustavsson A. Design and evaluation of an improved simplex method. Anal Chim Acta 1998;371:9–21.
[22] Naceur H, Guo YQ, Batoz JL. Blank optimization in sheet metal forming using an evolutionary algorithm. J Mater Process Technol 2004;151:183–91.
[23] Guo YQ, Batoz JL, Naceur H, Bouabdallah S, Mercier F, Barlet O. Recent developments on the analysis and optimum design of sheet metal forming parts using a simplified inverse approach. Comput Struct 2000;78:133–48.
[24] Boucard PA, Ladevéze P, Poss M, Rougée P. A nonincremental approach for large displacement problems. Comput Struct 1997;64(14):499–508.
[25] Naceur H, Delaméziere A, Batoz JL, Guo YQ, Knopf-Lenoir C. Some improvements on the optimum process design in deep drawing using the inverse approach. J Mater Process Technol 2004;146:250–62.
[26] ABAQUS Example Problems Manual, Hibbitt, Karlsson & Sorenson, Inc., p. 1.3.4.1–1.3.4.10.