A constrained optimization algorithm based on the simplex search method

Vivek Kumar Mehta & Bhaskar Dasgupta

To cite this article: Vivek Kumar Mehta & Bhaskar Dasgupta (2012) A constrained optimization algorithm based on the simplex search method, Engineering Optimization, 44:5, 537-550, DOI: 10.1080/0305215X.2011.598520

To link to this article: https://doi.org/10.1080/0305215X.2011.598520

Published online: 21 Sep 2011.

Submit your article to this journal

Article views: 351

Citing articles: 18 View citing articles
A constrained optimization algorithm based on the simplex search method

Vivek Kumar Mehta and Bhaskar Dasgupta*

Department of Mechanical Engineering, IIT, Kanpur – 208 016, India

(Received 8 July 2010; final version received 14 June 2011)

In this article, a robust method is presented for handling constraints with the Nelder and Mead simplex search method, which is a direct search algorithm for multidimensional unconstrained optimization. The proposed method is free from the limitations of previous attempts that demand the initial simplex to be feasible or a projection of infeasible points to the nonlinear constraint boundaries. The method is tested on several benchmark problems and the results are compared with various evolutionary algorithms available in the literature. The proposed method is found to be competitive with respect to the existing algorithms in terms of effectiveness and efficiency.

Keywords: constrained optimization; simplex search algorithm; constraint handling

1. Introduction

The Nelder–Mead algorithm, or simplex search algorithm (Nelder and Mead 1965), is one of the best known direct search algorithms for multidimensional unconstrained optimization. It was developed from the simplex method of Spendley (Spendley et al. 1962). In their original article, Nelder and Mead suggested two ways for handling constraints: (i) transforming the scale of the variables and (ii) modifying the function value so that it takes a high positive value in case any constraint is violated. Box (1965) found that this method of handling constraints leads to a degenerate simplex which collapses into a hypersurface at constraint boundaries. To handle this problem, he suggested the complex method. This method assumes the convexity of the feasible search space. Subrahmanyan (1989) proposed a modification in the simplex method using delayed reflection. He tested his algorithm on non-convex problems as well and found it to be successful. However, the major limitation of all these ways of handling constraints is the necessity for the initial simplex to lie in the feasible region.

Ghiasi et al. (2008) presented a backtracking (projection) procedure for handling nonlinear constraints. In this method, when a new point generated during the usual simplex iteration violates one of the constraints, the point is moved towards the original feasible point by a factor such that the distance between them reduces. This procedure is repeated till a feasible
point is found or the maximum number of iterations is reached. Again, this way of handling constraints demands a feasible set of the initial vertices of the simplex, which is its major limitation.

A methodology that successfully overcomes the limitation of previous methods involves independent treatment of constraint violations and objective functions. This approach has been adopted by Takahama and Sakai (2005) for handling constraints with the simplex algorithm. They extend the basic simplex algorithm using $\alpha$ level comparison, which compares points on the basis of their constraint violations, boundary mutation operator and multiple simplices.

In this article, the same methodology of treating constraint violations and objective functions separately has been followed. The effectiveness of the basic simplex search algorithm equipped with the proposed constraint handling method, which uses the default comparison scheme (without using mutation and multiple simplices) has been demonstrated by comparing its performance with the method of Takahama and Sakai (2005) and other evolutionary methods. The following sections describe the unconstrained simplex method and the proposed constraint handling method to be used with the simplex search algorithm, before presenting the results.

2. The basic simplex method

The simplex search method (Nelder and Mead 1965) is a direct method for function minimization. A particular version of the simplex algorithm as outlined by Dasgupta (2006) has been adopted here. In an $N$-dimensional design space a simplex is a polytope having $(N + 1)$ vertices. Let each vertex of the simplex be denoted by $x_k$, $k = 1, 2, \ldots, N + 1$. Starting with an initial simplex, the algorithm determines the worst $x_w$, second worst $x_s$ and the best $x_b$ vertices of the current simplex. Next, the algorithm attempts to move the simplex in a direction which is away from the worst vertex in the simplex. It does so by exploring the design space with three different operations, namely reflection, expansion and contraction and replacing the worst vertex with a new point in the design space.

To determine a new point, the centroid $x_c$ of all the other points except the worst point is calculated

$$x_c = \frac{1}{N} \sum_{k \neq w} x_k,$$

and point $x_w$ is reflected as $x_r = 2x_c - x_w$. This reflected point is taken as the default new point. However, depending upon the conditions given below, it may be revised.

1. If $f(x_r) < f(x_b)$, then $x_r$ potentially gives a good direction to move. So, an expansion of the simplex is performed to get a new point $x_{new}$ as $x_{new} = x_c + \alpha(x_c - x_w)$, $\alpha > 1$. If $f(x_{new}) < f(x_r)$, $x_{new}$ is taken as the new point.

2. If $f(x_r) \geq f(x_w)$, then moving in the suggested direction is not good. So, a negative contraction is performed to get a new point $x_{new}$, as $x_{new} = x_c - \beta(x_c - x_w)$, $0 < \beta < 1$.

3. If $f(x_s) < f(x_r) < f(x_w)$, then $x_r$ looks like a move in the right direction, but apparently too much. So, a positive contraction is performed to get a new point $x_{new}$, as $x_{new} = x_c + \beta(x_c - x_w)$, $0 < \beta < 1$.

A new simplex is formed by replacing the worst vertex in the current simplex with the new point so selected and the process is repeated, till convergence up to the required tolerance.
3. Proposed constraint handling method

A general nonlinear constrained optimization problem is defined as follows.

Minimize $f(\mathbf{x})$
subject to $g_i(\mathbf{x}) \leq 0$ for $i = 1, \ldots, p$,
$h_j(\mathbf{x}) = 0$ for $j = 1, \ldots, q$,
$l_k \leq x_k \leq u_k$, $k = 1, \ldots, n$,

where $\mathbf{x} = [x_1, x_2, x_3, \ldots, x_n]^T$ is the $n$-dimensional vector of decision variables, $f(\mathbf{x})$ is the function to be minimized, $g_i(\mathbf{x}) \leq 0$, $i = 1, \ldots, p$, are the $p$ inequality constraints, $h_j(\mathbf{x}) = 0$, $j = 1, \ldots, q$, are the $q$ equality constraints. The values $l_k$ and $u_k$ are the lower and upper bounds, respectively, for the decision variable $x_k$.

As shown in Figure 1, while moving in a constrained domain, a simplex can lie in three qualitatively different regions: (i) Infeasible region, when all the vertices of the simplex are infeasible, (ii) Feasible region, when all the vertices of the simplex are feasible and (iii) Boundary region, when some of the vertices are feasible and others infeasible.

From a practical point of view, it is unnecessary (at times impossible) to calculate the function value for an infeasible point. Hence, it is important to check the feasibility of all the vertices of the simplex. Once the feasibility check is performed and the region of location of the simplex is established, the constrained minimization problem is handled in the following way.

(i) If a simplex lies in the infeasible region entirely, the assigned value

$$F(\mathbf{x}) = CV = \sum_{j=1}^{p} \max\{0, g_j(\mathbf{x})\}$$

for each vertex is only its constraint violation (CV), which is the sum total of the absolute values of all the individual constraint violation equations.\(^1\) Hence, when the simplex is in the infeasible region, the algorithm minimizes the constraint violation and tries to move the simplex towards the feasible region.

(ii) If a simplex lies in the feasible region entirely, the assigned value

$$F(\mathbf{x}) = f(\mathbf{x})$$

for each vertex is only the function value of the original problem. Hence, when the simplex lies in the feasible region, the algorithm tries to improve upon the function value by guiding the simplex towards a lower function value.

(iii) If a simplex lies in the boundary region, there are some vertices which are feasible and others infeasible. For each feasible vertex, the value assigned is the function value of the original
problem. For an infeasible vertex, as mentioned earlier, it is unnecessary to calculate the
original function value and hence a value is assigned to the vertex which depends upon its
constraint violation and the value of the worst feasible vertex of the simplex. So, for a simplex
lying in the boundary region,

\[ F(x) = \begin{cases} f(x) & \text{if the vertex is feasible,} \\ f_{\text{max}} + CV & \text{if the vertex is infeasible,} \end{cases} \]  

(3)

where \( f_{\text{max}} \) is the function value of the worst feasible vertex. Hence, at any iteration, the
vertex with the highest constraint violation will always have the maximum value and will be
identified as the worst vertex of the simplex and the algorithm will try to move away from
that particular vertex.

For the problems with equality constraints, the constraint violation (CV) is modified as

\[ CV = \sum_{j=1}^{p} \max\{0, g_j(x)\} + \sum_{j=1}^{q} \max\{0, (|h_k(x)| - \delta)\} + R \sum_{j=1}^{q} |h_k(x)|^2. \]  

(4)

Among the two terms related to equality constraints \( h_k(x) \), the first allows the algorithm to
move in a narrow band, defined by using a suitable value for \( \delta \), around the manifold defined by
the equality constraints, and the second having a mild penalty parameter \( R \) directs the algorithm
towards the true solution of the problem. In the absence of the term with penalty parameter \( R \),
the accuracy of the obtained result would only depend upon the narrow band provided around the
constraint surface. However, a very small value of \( \delta \) poses difficulties for the simplex to move
freely and reach the solution. Again, a comparatively larger value of \( \delta \) would favour the easy
movement of the simplex, but with poor accuracy.

The effect of the value of \( \delta \) on the accuracy (defined as the distance of true minima from the
obtained minimum) and the cost (defined as the number of function evaluations required to reach
the minima) for the simplex algorithm, in the case of a test problem as depicted in figure 2, is
shown in figure 3. It is clearly evident from figure 3(a) that the accuracy of the minima obtained with a band defined with $\delta = 10^{-1}$ is lower in comparison to that of the minima obtained with a much narrower band defined with $\delta = 10^{-3}$. Though this improvement in accuracy is of the order of $10^{-2}$, the corresponding increase in the number of function evaluations, as shown in figure 3(b), is very large. It is quite clear that the simplex algorithm finds it difficult to move in a narrower band, so much so that for a band defined with $\delta = 10^{-4}$, the algorithm converges to a point which is far away from the true minima (see figure 3(a)).

Hence, to handle this difficulty, a term with a penalty parameter is introduced. In the presence of this term, the algorithm can work with a wider band, as in that case, for a simplex, lying entirely or partially in the band region, with the help of this term the algorithm will be able to detect and favour the vertex which is better than the others in terms of feasibility.

A similar strategy of assigning values to an infeasible point, in a set of some feasible and some infeasible points, has been exploited in the context of GA by Deb (2000). He has handled equality constraints by converting them into inequality constraints as $|h_k(x)| - \delta \leq 0$, where $\delta = 10^{-3}$, without using any penalty term.

4. Results and discussion

The present algorithm is tested on various benchmark problems that have been studied in the literature. Several other researchers have also tested their algorithms on these problems. These works employ a variety of techniques, as enumerated in table 1. In this section, the results obtained in the present study are compared with these approaches.

4.1. P1: Himmelblau’s nonlinear constrained optimization problem

This problem was proposed by Himmelblau (1972) and involves minimization of a quadratic function of five design variables subject to six inequality constraints. The problem is as follows.

Minimize $f(x) = 5.3578547 x_2^2 + 0.835689 x_1 x_5 + 37.293239 x_1 - 40792.141,$
subject to
$g_{1,2} = 0 \leq 85.334407 + 0.005685 x_2 x_5 + 0.000626 x_1 x_4 - 0.002205 x_3 x_5 \leq 92,$
Table 1. A survey of different algorithms.

<table>
<thead>
<tr>
<th>Authors</th>
<th>Algorithm/Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Takahama and Sakai (2005)</td>
<td>Constrained Method with Mutation</td>
</tr>
<tr>
<td>Deb (2000)</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>Ray and Liew (2003)</td>
<td>Algorithm based on Simulation of Social Behaviour</td>
</tr>
<tr>
<td>Coello (2000b)</td>
<td>GA + Self Adaptive Penalty Approach</td>
</tr>
<tr>
<td>Mezura and Coello (2005)</td>
<td>Evolutionary Algorithm</td>
</tr>
<tr>
<td>Coello (2000a)</td>
<td>Evolutionary Multi-Objective Optimization</td>
</tr>
<tr>
<td>Ragsdell and Phillips (1976)</td>
<td>Geometric Programming</td>
</tr>
<tr>
<td>Lee and Geem (2005)</td>
<td>Meta-heuristic based on Harmony Search</td>
</tr>
<tr>
<td>Fesanghary et al. (2008)</td>
<td>Hybrid Method (Harmonic search + SQP)</td>
</tr>
<tr>
<td>Homaifar et al. (1994)</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>Mahdavi et al. (2007)</td>
<td>Harmony Search</td>
</tr>
<tr>
<td>Pham et al. (2009)</td>
<td>Bees Algorithm</td>
</tr>
<tr>
<td>Cagnina et al. (2008)</td>
<td>Particle Swarm Optimizer</td>
</tr>
</tbody>
</table>

Table 2. Comparison of different algorithms for Himmelblau’s constrained problem. $N_f$ is the number of function evaluations.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>78.0</td>
<td>–</td>
<td>–</td>
<td>77,999,999</td>
</tr>
<tr>
<td>$x_2$</td>
<td>33.0</td>
<td>–</td>
<td>–</td>
<td>32,999,999</td>
</tr>
<tr>
<td>$x_3$</td>
<td>29.995</td>
<td>–</td>
<td>–</td>
<td>29,995,256</td>
</tr>
<tr>
<td>$x_4$</td>
<td>45.0</td>
<td>–</td>
<td>–</td>
<td>44,999,999</td>
</tr>
<tr>
<td>$x_5$</td>
<td>36.776</td>
<td>–</td>
<td>–</td>
<td>36,775,813</td>
</tr>
<tr>
<td>$f^*$</td>
<td>-30,665.5</td>
<td>-30,665.5</td>
<td>-30,665.538,671.8</td>
<td>-30,665,538,741,92</td>
</tr>
<tr>
<td>$N_f$</td>
<td>65,000</td>
<td>250,050</td>
<td>74,081</td>
<td>62,748</td>
</tr>
</tbody>
</table>

$g_{3,4} = 90 \leq 80,512.49 + 0.007,131.7x_2x_5 + 0.002,995,5x_1x_2 - 0.002,181,3x_3^2 \leq 110, \\
g_{5,6} = 20 \leq 9,300,961 + 0.004,702,6x_3x_5 + 0.001,254,7x_1x_3 + 0.001,908,5x_3x_4 \leq 25, \\
78 \leq x_1 \leq 102, \\
33 \leq x_2 \leq 35, \\
27 \leq x_3 \leq 45, \\
27 \leq x_4 \leq 45, \\
27 \leq x_5 \leq 45.$

The optimal solution for this problem is

$$x^* = [78.000, 33.000, 29.995, 45.000, 36.776]^T,$$

$$f(x^*) = -30665.5.$$

Several researchers (Deb 2000, Lee and Geem 2005, Takahama and Sakai 2005) have attempted this problem and have found the optimal solution as given in table 2. As is clearly evident, with the proposed method both the accuracy of the obtained solution and the efficiency are superior.

Some of the authors have tested their algorithm in another variation of this problem, where a parameter $0.000,626,2$ (typeset bold in the constraint $g_{1,2}$) has been taken as $0.000,26$. Fesanghary et al. (2008) and Coello (2000b) have got the optimum function value as $-31,024,316$ and $-31,020,859$, respectively, for this variation.
Table 3. Comparison of different algorithms for constrained problem P2.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>2.323 456 17</td>
<td>–</td>
<td>–</td>
<td>2.330 405 54</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1.951 242</td>
<td>–</td>
<td>–</td>
<td>1.951 320 50</td>
</tr>
<tr>
<td>$x_3$</td>
<td>– 0.448 467</td>
<td>–</td>
<td>–</td>
<td>– 0.477 330 69</td>
</tr>
<tr>
<td>$x_4$</td>
<td>4.361 919 9</td>
<td>–</td>
<td>–</td>
<td>4.365 874 40</td>
</tr>
<tr>
<td>$x_5$</td>
<td>– 0.630 075</td>
<td>–</td>
<td>–</td>
<td>– 0.624 464 24</td>
</tr>
<tr>
<td>$x_6$</td>
<td>1.038 665</td>
<td>–</td>
<td>–</td>
<td>1.038 255 60</td>
</tr>
<tr>
<td>$x_7$</td>
<td>1.605 384</td>
<td>–</td>
<td>–</td>
<td>1.594 152 25</td>
</tr>
<tr>
<td>$f^*$</td>
<td>680.641 357 4</td>
<td>680.634 46</td>
<td>680.630 057 4</td>
<td>680.630 058 1</td>
</tr>
<tr>
<td>$N_f$</td>
<td>160 000</td>
<td>350 070</td>
<td>850 73</td>
<td>798 09</td>
</tr>
</tbody>
</table>

4.2. P2

This problem involves seven design variables and four nonlinear constraints.

Minimize

$$f(x) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7$$

subject to

$$g_1 = 2x_1^2 + 3x_2 + x_3 + 4x_4^2 + 5x_5^5 - 127 \leq 0,$$

$$g_2 = 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282 \leq 0,$$

$$g_3 = 23x_1 + x_2^2 + 6x_6^2 - 8x_7 - 196 \leq 0,$$

$$g_4 = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0,$$

$$-10 \leq x_i \leq 10, \quad i = 1, \ldots, 7.$$

The optimal solution for this problem is

$$x^* = [2.330 499 \quad 1.951 372 \quad -0.477 541 4 \quad 4.365 726 \quad -0.624 487 0 \quad 1.038 131 \quad 1.594 227]^T,$$

$$f(x^*) = 680.630 057 3.$$

This is an interesting problem to solve as the feasible region for this problem occupies only 0.5% of the whole search space (Deb 2000), from the consideration of variable bounds. The result obtained from the present algorithm is compared with previous works in Table 3. Clearly, the result obtained by simplex based algorithms (Takahama and Sakai 2005 and the present algorithm) are much closer to the optimum as reported in the literature, in comparison to others. However, the number of function evaluations in the present case is less than that in Takahama and Sakai (2005).

4.3. P3

This problem involves five variables and three nonlinear equality constraints.

Minimize

$$f(x) = e^{x_1x_2x_3x_4x_5}$$

subject to

$$h_1(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0,$$
Table 4. Comparison of different algorithms for constrained problem P3.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-1.71070054$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$-$</td>
<td>$-$</td>
<td>$1.58824178$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-1.83917408$</td>
</tr>
<tr>
<td>$x_4$</td>
<td>$-$</td>
<td>$-$</td>
<td>$0.76226756$</td>
</tr>
<tr>
<td>$x_5$</td>
<td>$-$</td>
<td>$-$</td>
<td>$-0.76641280$</td>
</tr>
<tr>
<td>$f^*$</td>
<td>0.053950</td>
<td>0.053941</td>
<td>0.053969</td>
</tr>
<tr>
<td>$N_f$</td>
<td>350050</td>
<td>73527</td>
<td>73306</td>
</tr>
</tbody>
</table>

$h_2(x) = x_2x_3 - 5x_4x_5 = 0,$

$h_3(x) = x_1^3 + x_2^3 + 1 = 0,$

$- 2.3 \leq x_i \leq 2.3, \quad i = 1, 2,$

$- 3.2 \leq x_j \leq 3.2, \quad j = 3, 4, 5.$

The optimal solution for this problem is

$$x^* = [-1.717143 \quad 1.595709 \quad 1.827247 \quad -0.7636413 \quad -0.763645]^T,$$

$$f(x^*) = 0.053950.$$ 

While solving this problem, equality constraints are handled by converting them into inequality constraints as $|h_k(x)| - \delta \leq 0,$ where $\delta = 10^{-3}$ and taking the penalty parameter $R$ as $10^{-2}$. This is a difficult problem to solve as not all the runs of the algorithm leads to the optimum. The best solution obtained is comparable to the previously obtained solutions. However, the algorithm performs better with respect to the number of function evaluations (table 4).

4.4. P4

This problem involves minimization of a quadratic function of ten variables, subject to eight inequality constraints.

Minimize

$$f(x) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2$$

$$+ 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + 2(x_{10} - 7)^2 + 45$$

subject to

$$g_1 = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \leq 0,$$

$$g_2 = 10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0,$$

$$g_3 = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0,$$

$$g_4 = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0,$$

$$g_5 = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0,$$
Table 5. Comparison of different algorithms for constrained problem P4.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$f^*$</td>
<td>24.37248</td>
<td>24.306209</td>
<td>24.3075</td>
</tr>
<tr>
<td>$N_f$</td>
<td>350070</td>
<td>86121</td>
<td>240509</td>
</tr>
</tbody>
</table>

$$g_6 = x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \leq 0,$$

$$g_7 = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0,$$

$$g_8 = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0,$$

$$-10 \leq x_i \leq 10, \quad i = 1, \ldots, 10.$$

The optimal solution for this problem is

$$\mathbf{x}^* = [2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927],$$

$$f(\mathbf{x}^*) = 24.306209.$$

The comparison of the results obtained by different algorithms for this problem is given in table 5. As is evident from the results, with respect to Deb’s method (Deb 2000), which uses the same strategy of assigning values to an infeasible point as described in section 3, the present algorithm performs better both in terms of minimum value achieved and function evaluations. However, the $\alpha$ constrained method (Takahama and Sakai 2005) has achieved slightly better optima with much fewer function evaluations.

4.5. P5

Maximize $f(\mathbf{x}) = \frac{\sin^3(2\pi x_1) \sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$,

subject to $g_1 = x_1^2 - x_2 + 1 \leq 0$,

$$g_2 = 1 - x_1 + (x_2 - 4)^2 \leq 0,$$

$$0 \leq x_i \leq 10, \quad i = 1, 2.$$

The optimal solution for this problem is $\mathbf{x}^* = (1.2279713, 4.2453733)$ and $f(\mathbf{x}^*) = 0.095825$.

4.6. P6

Maximize $f(\mathbf{x}) = (\sqrt{n})^n \prod_{i=1}^{n} x_i$,

subject to $h_1 = \sum_{i=1}^{n} x_i^2 - 1 = 0$,

$$0 \leq x_i \leq 10, \quad (i = 1, 2, \ldots, n), \quad n = 10.$$

The optimal solution for this problem is $x_i^* = (1/\sqrt{n})$ for $i = 1, 2, \ldots, n$ and $f(\mathbf{x}^*) = 1$. 
Table 6. Comparison of results for constrained problems P5, P6, P7 and P8.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Takahama and Sakai (2005)</th>
<th>Present study</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f(x^*)$</td>
<td>$N_f$</td>
</tr>
<tr>
<td>P5</td>
<td>0.095825</td>
<td>128315</td>
</tr>
<tr>
<td>P6</td>
<td>1</td>
<td>85426</td>
</tr>
<tr>
<td>P7</td>
<td>0.75</td>
<td>84891</td>
</tr>
<tr>
<td>P8</td>
<td>-6961.81387</td>
<td>37492</td>
</tr>
</tbody>
</table>

4.7. P7

Minimize $f(x) = x_1^2 + (x_2 - 1)^2$, subject to $h_1 = x_2 - x_1^2 = 0$, $-1 \leq x_i \leq 1$, $(i = 1, 2)$.

The optimal solutions for this problem are $x^* = \left( \pm 1/\sqrt{2}, 1/2 \right)$ and $f(x^*) = 0.75$.

4.8. P8

Minimize $f(x) = (x_1 - 10)^3 + (x_2 - 20)^3$, subject to $g_1 = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0$, $g_2 = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$, $13 \leq x_1 \leq 100$, $0 \leq x_2 \leq 100$.

The optimal solution for this problem is $x^* = (14.095, 0.84296)$ and $f(x^*) = -6961.81388$.

A comparison of results obtained for problems P5, P6, P7 and P8 is given in table 6. While solving these problems, the $\delta$ and $R$ values are taken as $10^{-3}$ and $10^{-2}$, respectively. Clearly, the present algorithm performs much better in terms of function evaluations. The number of function evaluations in the case of 2-D problems (P5, P7, P8) is quite large with the $\alpha$ constrained method in comparison to the present algorithm.

4.9. Welded beam design problem

This problem involves a beam $B$ which is to be welded on a rigid support $A$, as shown in figure 4. There is a cost $C$ associated with the fabrication of the welded beam which needs to be minimized. The minimization of the cost associated, subject to constraints on shear stress ($\tau$), bending stress ($\sigma$), deflection ($\delta$) and buckling load ($P_c$), is the objective of this problem. The dimensions of the beam and weld $h$, $l$, $t$ and $b$ are the independent design variables.


Interestingly, different versions of this particular problem have been found in the literature. The main difference lies in the number of constraints and the way constraints are defined. Unfortunately, authors in the literature (Coello 2000a,b, Mezura and Coello 2005, Fesanghary et al. 2008) have compared the results of these different problems, while claiming their algorithms to be superior.
Figure 4. Welded beam.

In the present study, the proposed algorithm has been tested on all these versions. The mathematical formulation of the first version, WB1, is as follows.

Minimize \( C = f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2) \), \( (5) \)
subject to \( g_1(x) = \tau(x) - \tau_{\text{max}} \leq 0 \), \( (6) \)
\( g_2(x) = \sigma(x) - \sigma_{\text{max}} \leq 0 \), \( (7) \)
\( g_3(x) = x_1 - x_4 \leq 0 \), \( (8) \)
\( g_4(x) = -x_2 \leq 0 \), \( (9) \)
\( g_5(x) = -x_3 \leq 0 \), \( (10) \)
\( g_6(x) = P - P_c(x) \leq 0 \), \( (11) \)
\( g_7(x) = 0.125 - x_1 \leq 0 \), \( (12) \)
\( g_8(x) = \delta(x) - 0.25 \leq 0 \), \( (13) \)

where

\[
x = [x_1 \ x_2 \ x_3 \ x_4]^T = [h \ l \ t \ b]^T,
\]
\[
\tau(x) = \sqrt{(\tau')^2 + (\tau'')^2 + 2\tau'\tau''x_2^2/2R}, \quad \sigma(x) = \frac{6PL}{x_3x_4^2}, \quad \delta(x) = \frac{4PL^3}{E x_3^2 x_4},
\]
\[
P_c(x) = \frac{4.013}{L^2} \sqrt{EG \frac{x_1^2 x_3^2}{36}} \left(1 - \frac{x_3}{2L} \sqrt{\frac{E}{4G}}\right),
\]
\[
\tau' = \frac{P}{\sqrt{2x_1x_2}}, \quad \tau'' = \frac{MR}{J}, \quad M = P \left(L + \frac{x_2}{2}\right),
\]
\[
R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \quad J = 2 \left\{0.707x_1x_2 \left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},
\]
\[
P = 6000 \text{ lb}, \quad L = 14 \text{ in}, \quad E = 30 \times 10^6 \text{ psi}, \quad G = 12 \times 10^6 \text{ psi},
\]
\[
\tau_{\text{max}} = 13600 \text{ psi}, \quad \sigma_{\text{max}} = 30000 \text{ psi}.
\]

The best optimized solution (Ravindran et al. 2007) reported in the literature for this problem is \( h^* = 0.2444, l^* = 6.217, t^* = 8.2915 \) and \( b^* = 0.2444 \) with a function value \( f^* = 2.38116 \). Several authors (Ragsdell and Phillips 1976, Ravindran et al. 2007, Deb 2000, Ray and Liew 2003, Pham...


Table 7. Comparison of different algorithms for welded beam design problem WB1.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( h )</td>
<td>0.244 32</td>
<td>0.244 438 276</td>
<td>–</td>
<td>0.244 4</td>
<td>0.245 5</td>
<td>0.244 368 95</td>
</tr>
<tr>
<td>( t )</td>
<td>8.296 6</td>
<td>8.288 576 143</td>
<td>–</td>
<td>8.291 5</td>
<td>8.273 0</td>
<td>8.291 472 56</td>
</tr>
<tr>
<td>( b )</td>
<td>0.244 35</td>
<td>0.244 566 182</td>
<td>–</td>
<td>0.244 4</td>
<td>0.245 5</td>
<td>0.244 368 95</td>
</tr>
<tr>
<td>( f^* )</td>
<td>2.381 5</td>
<td>2.385 19</td>
<td>2.381 19</td>
<td>2.381 16</td>
<td>2.386</td>
<td>2.381 134 1</td>
</tr>
<tr>
<td>( N_f )</td>
<td>–</td>
<td>33 095</td>
<td>40 080</td>
<td>–</td>
<td>–</td>
<td>20 500</td>
</tr>
</tbody>
</table>

Table 8. Comparison of different algorithms for welded beam design problem WB2.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( h )</td>
<td>0.205 720</td>
<td>0.205 730</td>
<td>0.205 729 63</td>
</tr>
<tr>
<td>( l )</td>
<td>3.470 600</td>
<td>3.470 490</td>
<td>3.253 119 67</td>
</tr>
<tr>
<td>( t )</td>
<td>9.036 820</td>
<td>9.036 620</td>
<td>9.036 626 17</td>
</tr>
<tr>
<td>( b )</td>
<td>0.205 720</td>
<td>0.205 730</td>
<td>0.205 729 63</td>
</tr>
<tr>
<td>( f^* )</td>
<td>1.724 8</td>
<td>1.724 8</td>
<td>1.695 247 4</td>
</tr>
<tr>
<td>( N_f )</td>
<td>90 000</td>
<td>200 000</td>
<td>20 036</td>
</tr>
</tbody>
</table>

et al. 2009) have tried their algorithms on this version of the welded beam design problem. A comparison of the present work with some of the previous studies is presented in table 7.

Clearly, the present algorithm based on the simplex method performs much better in comparison to the other algorithms as both the optimal function value obtained and the number of function evaluations (\( N_f \)) in the present study are lower than in the previous studies.

In a recent study (Fesanghary et al. 2008), the authors have solved another version of the welded beam design problem (WB2). While defining constraints they have taken deflection \( \delta(x) \), buckling load \( P_c(x) \) and polar moment of inertia \( J \) as

\[
\delta(x) = \frac{6PL^3}{Ex_3^2x_4},
\]

\[
P_c(x) = \frac{4.013EL^{1/2}}{L^2} \left[ \left( 1 - \frac{x_3}{2L} \right)^{1/2} \right],
\]

\[
J = 2 \left\{ \sqrt{2}x_1x_2 \left[ \frac{x_2^2}{4} + \left( \frac{x_1 + x_3}{2} \right)^2 \right] \right\}.
\]

Mahdavi et al. (2007) have also solved the same version and a comparison of the present study with these studies is given in table 8. Clearly, the algorithm of the present study performs much better in comparison to previous algorithms, as it has achieved the lowest function value with fewer function evaluations.

Yet another version WB3 of the same welded beam design problem, takes into consideration an additional constraint

\[
g_9(x) = 1.104 71 x_1^2 + 0.048 11 x_3 x_4 (14 + x_2) - 5 \leq 0.
\]

In addition, this version uses the expression for \( P_c \) given in equation (15). Several researchers (Coello 2000a,b, Mezura and Coello 2005, Cagnina et al. 2008) have treated this problem and a comparison of their studies with the present work is given in table 9. The present study yields the same result as the earlier ones, but with fewer function evaluations.
Table 9. Comparison of different algorithms for welded beam design problem WB3.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(h)</td>
<td>0.205 729</td>
<td>0.205 730</td>
<td>0.208 8</td>
<td>0.182 9</td>
<td>0.205 728 85</td>
</tr>
<tr>
<td>(l)</td>
<td>3.470 488</td>
<td>3.470 489</td>
<td>3.420 5</td>
<td>4.048 3</td>
<td>3.470 505 67</td>
</tr>
<tr>
<td>(b)</td>
<td>0.205 729</td>
<td>0.205 729</td>
<td>0.210 0</td>
<td>0.205 9</td>
<td>0.205 729 64</td>
</tr>
<tr>
<td>(f^*)</td>
<td>1.724 8</td>
<td>1.724 8</td>
<td>1.748 3</td>
<td>1.824 2</td>
<td>1.724 855</td>
</tr>
<tr>
<td>(N_f)</td>
<td>24 000</td>
<td>30 000</td>
<td>90 000</td>
<td>5000</td>
<td>21 995</td>
</tr>
</tbody>
</table>

Table 10. Experimental results of algorithm for 20 independent runs. \(N_{f_{avg}}\) and \(N_{c_{avg}}\) are the average numbers of function and constraint evaluations, respectively.

<table>
<thead>
<tr>
<th></th>
<th>Optimal/Known best</th>
<th>Best</th>
<th>Mean</th>
<th>Median</th>
<th>Worst</th>
<th>(N_{f_{avg}})</th>
<th>(N_{c_{avg}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>-30 665.5</td>
<td>-30 665.538 6</td>
<td>-30 665.538 6</td>
<td>-30 665.538 6</td>
<td>-30 665.536 5</td>
<td>68 835.7</td>
<td>145 468.2</td>
</tr>
<tr>
<td>P2</td>
<td>680.033 05</td>
<td>680.630 058 1</td>
<td>680.630 095 0</td>
<td>680.630 064</td>
<td>680.630 356</td>
<td>81 079.5</td>
<td>135 675</td>
</tr>
<tr>
<td>P3</td>
<td>0.053 950</td>
<td>0.053 969</td>
<td>0.441 133</td>
<td>0.354 716</td>
<td>0.919 539</td>
<td>96 536</td>
<td>207 584.3</td>
</tr>
<tr>
<td>P5</td>
<td>0.095 825</td>
<td>0.095 825</td>
<td>0.095 825</td>
<td>0.095 825</td>
<td>0.095 825</td>
<td>1267.1</td>
<td>1293.4</td>
</tr>
<tr>
<td>P6</td>
<td>1</td>
<td>0.999 818</td>
<td>0.999 527</td>
<td>0.999 539</td>
<td>0.999 094</td>
<td>79 421.4</td>
<td>130 448.1</td>
</tr>
<tr>
<td>P7</td>
<td>0.75</td>
<td>0.749 99</td>
<td>0.749 99</td>
<td>0.749 99</td>
<td>0.749 99</td>
<td>2074.8</td>
<td>3135</td>
</tr>
<tr>
<td>P8</td>
<td>-6961.813 8</td>
<td>-6961.813 87</td>
<td>-6961.813</td>
<td>-6961.813 85</td>
<td>-6961.813 82</td>
<td>5714.1</td>
<td>11 405.5</td>
</tr>
<tr>
<td>WB1</td>
<td>2.381 16</td>
<td>2.381 134</td>
<td>2.381 178.6</td>
<td>2.381 164 1</td>
<td>2.381 261 4</td>
<td>20 903.4</td>
<td>34 679.1</td>
</tr>
<tr>
<td>WB2</td>
<td>1.724 8</td>
<td>1.695 247</td>
<td>1.695 2695</td>
<td>1.695 255 2</td>
<td>1.695 318 5</td>
<td>21 502.4</td>
<td>35 212.6</td>
</tr>
<tr>
<td>WB3</td>
<td>1.724 8</td>
<td>1.724 855</td>
<td>1.724 865</td>
<td>1.724 861</td>
<td>1.724 89</td>
<td>20 994.9</td>
<td>34 822.1</td>
</tr>
</tbody>
</table>

To show the effectiveness of the present algorithm, 20 independent runs of the algorithm were tried on all of the above problems, a summary of which is presented in table 10. For each problem, the best, mean, median and the worst of the obtained function values among 20 independent runs are presented along with the average number of function and constraint evaluations.

5. Conclusions

The simplex algorithms as proposed by Nelder and Mead is a well-known method for unconstrained optimization. In this article, a modified simplex algorithm that is capable of handling constraints has been proposed. As evident from the results, the proposed method performs much better than the several evolutionary algorithms as suggested in the literature.

It has been shown that the strategy of assigning the values to an infeasible point, as adopted in the present work, performs better with the simplex method in comparison to the GA based approach (Deb 2000). Moreover, is has also been demonstrated that, for most of the benchmark problems, the proposed method, which uses a simple algorithm for handling constraints, is comparable to the \(\alpha\) constrained method of Takahama and Sakai (2005), which utilizes mutation and multiple simplices.

Note

1. The equation here involves only the inequality constraints. Later in this section, the definition of constraint violation (CV) will be modified to include equality constraints as well.
References


