Optimization Letters (2023) 17:283–297
https://doi.org/10.1007/s11590-022-01953-y

ORIGINAL PAPER

Practical initialization of the Nelder–Mead method for computationally expensive optimization problems

Shintaro Takenaga[1,2] · **Yoshihiko Ozaki**[2,3] · **Masaki Onishi**[2]

Received: 27 May 2022 / Accepted: 6 November 2022 / Published online: 7 December 2022

Abstract
Black-box optimization (BBO) algorithms are widely employed by practitioners to address computationally expensive real-world problems such as automatic tuning of machine learning models and evacuation route planning. The Nelder–Mead (NM) method is a well-known local search heuristic for BBO that has been applied to solve many real-world problems from way back because of its promising performance. However, this method has a strong dependence on initialization due to its local search tendency. Nevertheless, a discussion on the proper initialization of the NM method is limited to the recent study by Wessing (Optim Lett 13(4):847–856, 2019), which is solely based on an analysis using the simple sphere function. In this study, we take a further step to improve Wessing's result by massively investigating how the initialization affects the search performance in views of the initial simplex size and shape and a constraint handling method that is employed on 24 BBO benchmarking problems. Based on the numerical results, we present the empirical best practice for the initialization of the NM method for cases involving a limited evaluation budget.

Keywords Nelder–Mead method · Initialization · Computationally expensive optimization · Black-box optimization

1 Introduction

Black-box optimization (BBO) is an approach for optimizing an objective function without any information regarding the analytic form and gradient of the objective. This approach has been applied to important real-world problems, including

✉ Shintaro Takenaga
shintaro-takenaga@aist.go.jp

[1] University of Tsukuba, Ibaraki, Japan

[2] AI Research Center, AIST, Tokyo, Japan

[3] GREE, Inc., Tokyo, Japan

automatic tuning of machine learning models [2, 18] and evacuation route planning [21]. In such real-world problems, it is generally important to obtain a promising solution with a limited evaluation budget because their objective functions usually involve computationally expensive operations, such as training of deep neural networks and crowd evacuation simulations.

The Nelder–Mead (NM) method [17] is a local search heuristic that uses a simplex, and it has been used to solve BBO problems for more than half a century. This method converges quickly with a relatively small number of function evaluations; thus, it can achieve preferable results for computationally expensive problems, such as the automatic tuning of machine learning models [18]. However, it is empirically known that the search performance of the NM method strongly depends on initialization, which concerns the generation of the initial simplex. Therefore, to achieve good optimization results, it is crucial to provide proper initialization. However, there has been only a very limited discussion on the proper initialization of the NM method so far.

Recently, Wessing investigated how initialization affects the search performance of the NM method, using the Sphere function [26]. After arguing for the necessity of determining a proper initial simplex, Wessing proposed generating an initial simplex that is as large as the normalized search space. However, it is currently unclear whether the obtained results can be generalized to a variety of computationally expensive problems because they are solely based on an analysis using the simple Sphere benchmark function in the large evaluation budget case. Therefore, further empirical assessments are required for practitioners to identify the best practice for the initialization of the NM method to solve computationally expensive problems.

Additionally, practitioners need to consider handling infeasible solutions to tackle constrained problems, even considering the simplest box-constrained case, because the NM method was originally designed for unconstrained optimization. The handling methods used to address constraints significantly affect the search performance in BBO [25]. Therefore, the effects of these methods on the proper initialization of the NM method should be investigated.

Motivated by the above discussion, in this study, we massively investigate how initialization affects the NM method on a proven benchmark suite, namely BBO benchmarking (BBOB) [8]. The main contributions and important findings of this study are summarized as follows:

- We empirically find that the search performance of the NM method highly depends not only on the size of the initial simplex but also on its shape.
- We also present a practical initialization heuristic to maximize the performance of the NM method for the limited-evaluation-budget case based on the experimental results. This involves normalizing the search space to unit hypercube and generating a regular-shaped simplex that is as large size as possible regardless of the constraint handling method that is employed.

2 Background

In this section, we describe the NM method, its initialization methods, and the methods for handling box constraints.

2.1 Nelder–Mead method

Algorithm 1 Nelder–Mead Method [17]

Input:
$\boldsymbol{Y}_0 = \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^n\}$ ▷ Initial simplex
$0 < \gamma < 1, -1 < \delta^{\mathrm{ic}} < 0 < \delta^{\mathrm{oc}} < \delta^{\mathrm{r}} < \delta^{\mathrm{e}}$ ▷ Coefficients
for $k \leftarrow 0, 1, \ldots$ **do**
  Order the vertices of $\boldsymbol{Y}_k$ so that $f^0 = f(\boldsymbol{y}^0) \leq f^1 = f(\boldsymbol{y}^1) \leq \cdots \leq f^n = f(\boldsymbol{y}^n)$ ▷ Order
  $\boldsymbol{y}^{\mathrm{c}} \leftarrow \frac{1}{n} \sum_{i=0}^{n-1} \boldsymbol{y}^i$
  $\boldsymbol{y}^{\mathrm{r}} \leftarrow \boldsymbol{y}^{\mathrm{c}} + \delta^{\mathrm{r}}(\boldsymbol{y}^{\mathrm{c}} - \boldsymbol{y}^n), f^{\mathrm{r}} \leftarrow f(\boldsymbol{y}^{\mathrm{r}})$ ▷ Reflection
  if $f^0 \leq f^{\mathrm{r}} < f^{n-1}$ **then**
    $\boldsymbol{Y}_{k+1} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{n-1}, \boldsymbol{y}^{\mathrm{r}}\}$
    continue
  if $f^{\mathrm{r}} < f^0$ **then** ▷ Expansion
    $\boldsymbol{y}^{\mathrm{e}} \leftarrow \boldsymbol{y}^{\mathrm{c}} + \delta^{\mathrm{e}}(\boldsymbol{y}^{\mathrm{c}} - \boldsymbol{y}^n), f^{\mathrm{e}} \leftarrow f(\boldsymbol{y}^{\mathrm{e}})$
    if $f^{\mathrm{e}} < f^{\mathrm{r}}$ **then**
      $\boldsymbol{Y}_{k+1} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{n-1}, \boldsymbol{y}^{\mathrm{e}}\}$
    else if $f^{\mathrm{r}} \leq f^{\mathrm{e}}$ **then**
      $\boldsymbol{Y}_{k+1} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{n-1}, \boldsymbol{y}^{\mathrm{r}}\}$
    continue
  if $f^{n-1} \leq f^{\mathrm{r}} < f^n$ **then** ▷ Outside Contraction
    $\boldsymbol{y}^{\mathrm{oc}} \leftarrow \boldsymbol{y}^{\mathrm{c}} + \delta^{\mathrm{oc}}(\boldsymbol{y}^{\mathrm{c}} - \boldsymbol{y}^n), f^{\mathrm{oc}} \leftarrow f(\boldsymbol{y}^{\mathrm{oc}})$
    if $f^{\mathrm{oc}} < f^{\mathrm{r}}$ **then**
      $\boldsymbol{Y}_{k+1} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{n-1}, \boldsymbol{y}^{\mathrm{oc}}\}$
    else if $f^{\mathrm{r}} \leq f^{\mathrm{oc}}$ **then**
      go to Shrinkage
    continue
  if $f^{\mathrm{r}} \geq f^n$ **then** ▷ Inside Contraction
    $\boldsymbol{y}^{\mathrm{ic}} \leftarrow \boldsymbol{y}^{\mathrm{c}} + \delta^{\mathrm{ic}}(\boldsymbol{y}^{\mathrm{c}} - \boldsymbol{y}^n), f^{\mathrm{ic}} \leftarrow f(\boldsymbol{y}^{\mathrm{ic}})$
    if $f^{\mathrm{ic}} < f^n$ **then**
      $\boldsymbol{Y}_{k+1} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^{n-1}, \boldsymbol{y}^{\mathrm{ic}}\}$
    else if $f^n \leq f^{\mathrm{ic}}$ **then**
      go to Shrinkage
    continue
  $\boldsymbol{Y}_{k+1} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^0 + \gamma(\boldsymbol{y}^1 - \boldsymbol{y}^0), \boldsymbol{y}^0 + \gamma(\boldsymbol{y}^2 - \boldsymbol{y}^0), \ldots, \boldsymbol{y}^0 + \gamma(\boldsymbol{y}^n - \boldsymbol{y}^0)\}$ ▷ Shrinkage

The algorithm of the NM method for minimization is shown in Algorithm 1. To optimize an n-dimensional objective function, the NM method requires a simplex composed of affinely independent $n + 1$ vertices in an n-dimensional search space. For example, a two-dimensional simplex is a triangle, and a three-dimensional simplex is a tetrahedron. The NM method iteratively performs the five major operations of transforming the simplex—Reflection, Expansion, Outside Contraction, Inside Contraction, and Shrinkage-based on the objective function values of the solutions corresponding to each vertex of the simplex. Figure 1 shows these five operations in

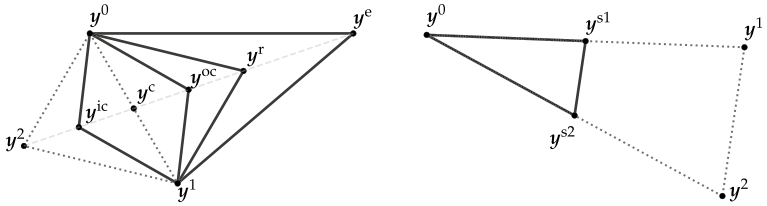


Fig. 1 Simplex transformations by the NM method: Reflection ($\boldsymbol{y}^{\mathrm{r}}$), Expansion ($\boldsymbol{y}^{\mathrm{e}}$), Outside Contraction ($\boldsymbol{y}^{\mathrm{oc}}$), Inside Contraction ($\boldsymbol{y}^{\mathrm{ic}}$), and Shrinkage ($\boldsymbol{y}^{\mathrm{s1}}$ and $\boldsymbol{y}^{\mathrm{s2}}$)

two-dimensional space. In this figure, $\boldsymbol{y}^0$, $\boldsymbol{y}^1$, and $\boldsymbol{y}^2$ are the vertices of the simplex before the operation such that $f(\boldsymbol{y}^0) < f(\boldsymbol{y}^1) < f(\boldsymbol{y}^2)$. For the coefficients of the NM method in Algorithm 1, we use the following standard settings [3]:

$$\delta^{\mathrm{r}} = 1,\ \delta^{\mathrm{e}} = 2,\ \delta^{\mathrm{oc}} = \frac{1}{2},\ \delta^{\mathrm{ic}} = -\frac{1}{2},\ \gamma = \frac{1}{2}. \tag{1}$$

2.2 Initialization methods for the Nelder–Mead method

A previous study [26] investigated the five initialization methods: Pfeffer [7], Nash [16], Han [9], Varadhan [22, 23], and Std basis [24]. Examples of the simplices generated using these methods are shown in Fig. 2. The shapes of the generated simplices can be classified into two types, "regular" and "standard," with a few exceptions. The former is a simplex that all of its side lengths are the same, and the latter is a simplex that its vertices correspond to the standard basis vectors. In Fig. 2, we observe that the Han and Varadhan methods generate regular simplices, whereas the Nash and Std basis methods generate standard simplices. For the Pfeffer method, the diagonally placed simplices are standard, but the remaining ones are sharper.

2.3 Handling methods for box constraints

In BBO, one of the most frequently appearing constraints is the box constraint, wherein a variable has specific lower and upper bounds. Several methods are available to handle box constraints, including the Extreme Barrier [1], Projection [10, 12–14], Reflection [25], and Wrapping [20] methods (see Fig. 3). All of these approaches transform constrained problems into unconstrained problems, to which the NM method can be applied.

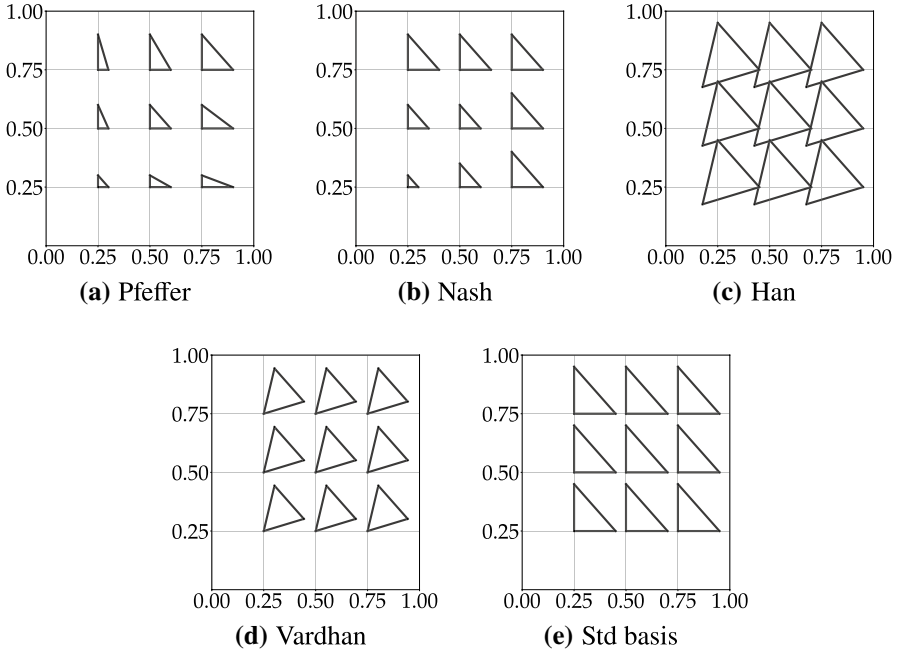


Fig. 2 Examples of two-dimensional simplices generated by the Pfeffer, Nash, Han, Varadhan, and Std basis methods

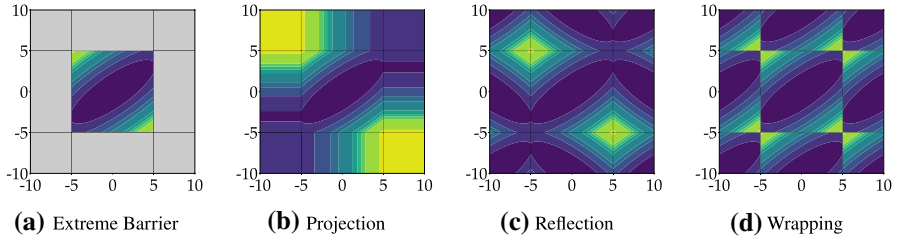


Fig. 3 Objective landscapes of the Attractive Sector function [8] ($D = [-5, 5]^2$) with each handling method. The gray area indicates $+\infty$

Assume $f : \mathbb{R}^n \to \mathbb{R}$ and $D = [l_0, u_0] \times [l_1, u_1] \times \cdots \times [l_{n-1}, u_{n-1}] \subset \mathbb{R}^n$. We consider the minimization problem $\min_{\boldsymbol{y} \in \mathbb{R}^n} f(\boldsymbol{y})$ subject to $\boldsymbol{y} \in D$. The Extreme Barrier approach defines a penalty function $f_{\mathrm{E}} : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ that assigns $+\infty$,[1] which is the penalty value, to the objective function value corresponding to an infeasible solution as follows:

[1] In practice, a large constant is usually employed rather than $+\infty$ in implementations.

(Extreme Barrier)

$$f_{\mathrm{E}}(\boldsymbol{y}) = \begin{cases} f(\boldsymbol{y}) & \boldsymbol{y} \in D \\ +\infty & \boldsymbol{y} \notin D. \end{cases} \tag{2}$$

Subsequently, we minimize the penalty function f_{E} instead of the original objective function f to solve the target problem. The Projection, Reflection, and Wrapping approaches define repair functions $f_{\mathrm{P}} : \mathbb{R}^n \to \mathbb{R}$, $f_{\mathrm{R}} : \mathbb{R}^n \to \mathbb{R}$, and $f_{\mathrm{W}} : \mathbb{R}^n \to \mathbb{R}$, respectively, which assign the objective function value corresponding to a specific feasible solution to that of an infeasible solution by applying a simple mapping rule:

(Projection)

$$f_{\mathrm{P}}(\boldsymbol{y}) = f([T_{\mathrm{P}_0}(y_0), \ldots, T_{\mathrm{P}_{n-1}}(y_{n-1})]), \tag{3}$$

$$T_{\mathrm{P}_i}(y) = \begin{cases} y & l_i \le y \le u_i \\ u_i & y > u_i \\ l_i & y < l_i. \end{cases} \tag{4}$$

(Reflection)

$$f_{\mathrm{R}}(\boldsymbol{y}) = f([T_{\mathrm{R}_0}(y_0), \ldots, T_{\mathrm{R}_{n-1}}(y_{n-1})]), \tag{5}$$

$$T_{\mathrm{R}_i}(y) = \begin{cases} y & l_i \le y \le u_i \\ T_{\mathrm{R}_i}(u_i + (u_i - y)) & y > u_i \\ T_{\mathrm{R}_i}(l_i + (l_i - y)) & y < l_i. \end{cases} \tag{6}$$

(Wrapping)

$$f_{\mathrm{W}}(\boldsymbol{y}) = f([T_{\mathrm{W}_0}(y_0), \ldots, T_{\mathrm{W}_{n-1}}(y_{n-1})]), \tag{7}$$

$$T_{\mathrm{W}_i}(y) = \begin{cases} y & l_i \le y \le u_i \\ T_{\mathrm{W}_i}(y - (u_i - l_i)) & y > u_i \\ T_{\mathrm{W}_i}(y + (u_i - l_i)) & y < l_i. \end{cases} \tag{8}$$

In these equations, $T_{\mathrm{P}_i} : \mathbb{R} \to [l_i, u_i]$, $T_{\mathrm{R}_i} : \mathbb{R} \to [l_i, u_i]$, and $T_{\mathrm{W}_i} : \mathbb{R} \to [l_i, u_i]$ $(i = 0, \ldots, n-1)$ are the auxiliary mapping functions. Similar to the Extreme Barrier approach, we minimize f_{P}, f_{R}, and f_{W} instead of f to solve the target problem using these approaches.

3 Investigating the effect of initialization

In this section, we empirically investigate the effect of the initial simplex of the NM method using comprehensive experiments. We focus on the effects of the initial simplex size and shape and the method employed to handle box constraints on the search performance for the limited-evaluation-budget case. Our research questions are described as follows:

Q.1 Is it better to generate a larger initial simplex as Wessing [26] previously reported?

Q.2 Which initial simplex shape is better, regular, or standard?

Q.3 Is the proper initial simplex dependent on the constraint handling method that is employed?

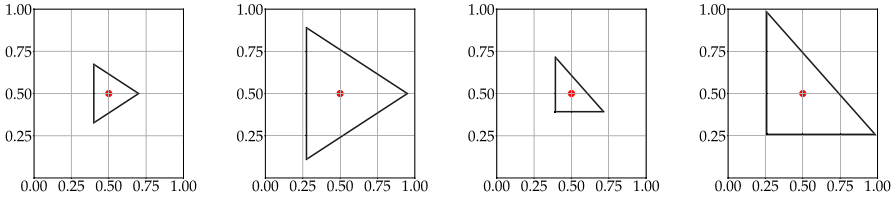
Regarding the effect of the simplex size, it can be quantitatively evaluated using the volume metric [1, 3]. Therefore, we evaluate initial simplices with different simplex volumes. The volume of the n-dimensional simplex $\boldsymbol{Y} = \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^n\}$ is defined as:

$$\mathrm{vol}(\boldsymbol{Y}) = \frac{|\det(\boldsymbol{L})|}{n!} \tag{9}$$

where $\boldsymbol{L}$ denotes a matrix such that:

$$\boldsymbol{L} = [(\boldsymbol{y}^1 - \boldsymbol{y}^0), (\boldsymbol{y}^2 - \boldsymbol{y}^0), \ldots, (\boldsymbol{y}^n - \boldsymbol{y}^0)]. \tag{10}$$

Regarding the effect of the simplex shape, we evaluate regular and standard simplices because these shapes are widely adopted by existing initialization methods (cf. Sect. 2.2). Finally, for the effect of the constraint handling methods, we evaluate the Extreme Barrier, Projection, Reflection, and Wrapping approaches (cf. Sect. 2.3).



(a) Regular: 0.2 **(b)** Regular: 0.45 **(c)** Standard: small **(d)** Standard: large

Fig. 4 Examples of two-dimensional simplices. 0.2 and 0.45 indicate the input γ values of the regular simplices. The red points indicate the centroids. The volume of **(c)** is the same as **(a)** and the volume of **(d)** is the same as **(b)**

3.1 Experimental setup

Algorithm 2 Generate regular simplex [19]

Input: $n \in \mathbb{N}$, γ = L2 norm, $\boldsymbol{p}$ = centroid

$\boldsymbol{Y} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^n\}$
for $i \leftarrow 0, 1, \ldots, n$ **do**
  for $j \leftarrow 0, 1, \ldots, n-1$ **do**
    $y^i_j \leftarrow 0$
$b \leftarrow 0$
for $i \leftarrow 0, 1, \ldots, n-1$ **do**
  $c \leftarrow \sqrt{1-b}$, $y^i_i \leftarrow c$, $r \leftarrow \frac{-\frac{1}{n}-b}{c}$
  for $j \leftarrow i+1, i+2, \ldots, n$ **do**
    $y^j_i \leftarrow r$
  $b \leftarrow b + r^2$
for $i \leftarrow 0, 1, \ldots, n$ **do**
  $\boldsymbol{y}^i \leftarrow \gamma \boldsymbol{y}^i + \boldsymbol{p}$
return $\boldsymbol{Y}$

Algorithm 3 Generate standard simplex

Input: $n \in \mathbb{N}$, v = criterion volume, $\boldsymbol{p}$ = centroid
$\boldsymbol{Y} \leftarrow \{\boldsymbol{y}^0, \boldsymbol{y}^1, \ldots, \boldsymbol{y}^n\}$
for $i \leftarrow 0, 1, \ldots, n$ **do**
  for $j \leftarrow 0, 1, \ldots, n-1$ **do**
    if $i = j$ **then**
      $y^i_j \leftarrow 1$
    else
      $y^i_j \leftarrow 0$
$\alpha \leftarrow 1.0, \gamma \leftarrow 1.0$
while True **do**
  $\gamma \leftarrow 0.9\gamma, \boldsymbol{Y}' \leftarrow \alpha\boldsymbol{Y}, \beta \leftarrow \alpha, v' \leftarrow \mathrm{vol}(\boldsymbol{Y}')$
  if $v' - v > 0$ **then**
    $\alpha \leftarrow \alpha(1-\gamma)$
  else
    $\alpha \leftarrow \alpha(1+\gamma)$
  if $|\alpha - \beta| < \epsilon$ **then** ▷ ϵ: a small constant
    break
$\boldsymbol{Y} \leftarrow \boldsymbol{Y}', \boldsymbol{c} \leftarrow \frac{1}{n}\sum_{i=0}^{n} \boldsymbol{y}^i$
for $i \leftarrow 0, 1, \ldots, n$ **do**
  $\boldsymbol{y}^i \leftarrow \boldsymbol{y}^i + (\boldsymbol{p} - \boldsymbol{c})$
return $\boldsymbol{Y}$

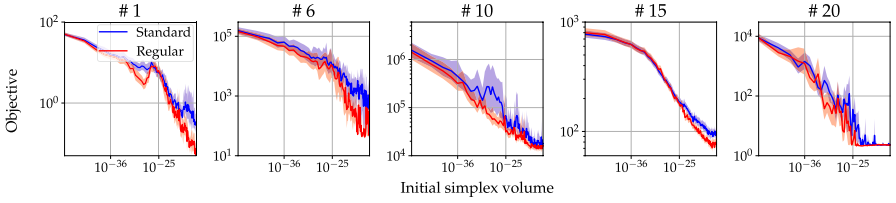
In our experiments, the search space was always assumed to be normalized to the n-dimensional unit hypercube $[0, 1]^n$ in advance according to Wessing [26]. To generate regular and standard simplices, we prepared Algorithm 2 [19] and Algorithm 3, respectively. Algorithm 2 requires the dimension n, the L2 norm γ to determine the generated simplex size, and the centroid $\boldsymbol{p}$ of the simplex as the input parameters. The larger the γ, the larger the volume of the generated simplex. Conversely, Algorithm 3 requires the dimension n, the criterion simplex volume v, which allows us to generate a standard simplex with the same volume as a regular simplex to compare, and the centroid $\boldsymbol{p}$ of the simplex as input parameters. Figure 4 shows examples of two-dimensional simplices, in which their centroids are $\mathbf{0.5}$.

We evaluated the search performance of the NM method initialized with a variety of simplices on 24 benchmark functions described in detail later by employing each of the four constraint handling methods introduced in Sect. 2.3. By using the simplex generation algorithms, we generated 200 types of initial simplices with centroids randomly placed at $[0.1, 0.9]^n$ for each benchmark function instance. First, 100 types of regular simplices were generated by using Algorithm 2 with the L2 norm $\gamma = 0.45 \times 0.01, 0.45 \times 0.02, \ldots, 0.45 \times 1.00$. The remaining 100 types of standard simplices were generated by using Algorithm 3 with the criterion volumes that are the same as the volumes of the 100 types of regular simplices.

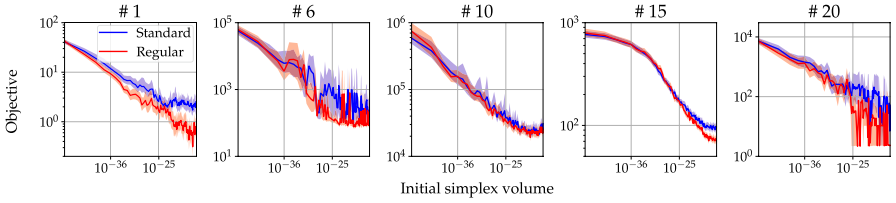
As a benchmark suite, we employed BBOB [8], which is one of the most popular benchmarks for evaluating the performance of BBO algorithms. BBOB contains 24 artificial functions with a box-constrained search space $[-5, 5]^n$. These functions are classified into five groups based on their landscape features: 1. separable (#1–5), 2. low or moderate-conditioning (#6–9), 3. high-conditioning and unimodal

Table 1 List of BBOB functions [8]

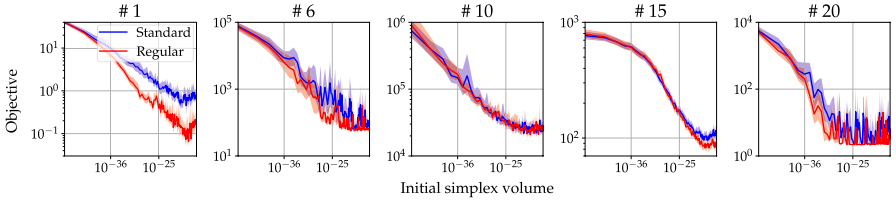
#	Name	Primal landscape feature	Other landscape features
1	Sphere	Separable	Unimodal
2	Ellipsoidal		Unimodal, high-conditioning
3	Rastrigin		Multimodal
4	Büche–Rastrigin		Multimodal, asymmetric
5	Linear slope		Unimodal
6	Attractive sector	Low or moderate-conditioning	Unimodal, asymmetric
7	Step ellipsoidal		Unimodal, non-separable
8	Rosenbrock		Unimodal, strong inter-variable dependence
9	Rosenbrock, rotated		Unimodal, strong inter-variable dependence
10	Ellipsoidal	High-conditioning and unimodal	Non-separable
11	Discus		
12	Bent cigar		
13	Sharp ridge		
14	Different powers		
15	Rastrigin	Multimodal with adequate global structure	Non-separable
16	Weierstrass		Several global optima, repetitive
17	Schaffers F7		Low-conditioning, asymmetric
18	Schaffers F7		Moderate-conditioning
19	Composite Griewank–Rosenbrock		Strong inter-variable dependence
20	Schwefel	Multimodal with weak global structure	
21	Gallagher's Gaussian 101-me peaks		Low-conditioning
22	Gallagher's Gaussian 21-hi peaks		Moderate-conditioning
23	Katsuura		Highly repetitive
24	Lunacek bi-Rastrigin		



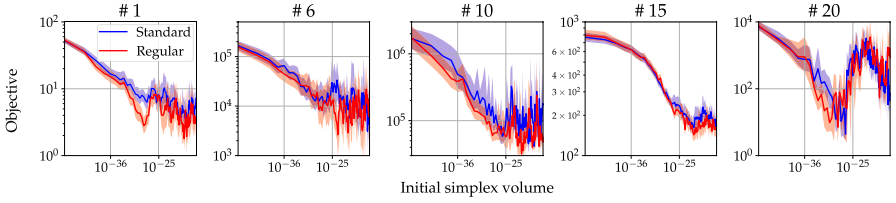
(a) Constraint Handling Method: Extreme Barrier



(b) Constraint Handling Method: Projection



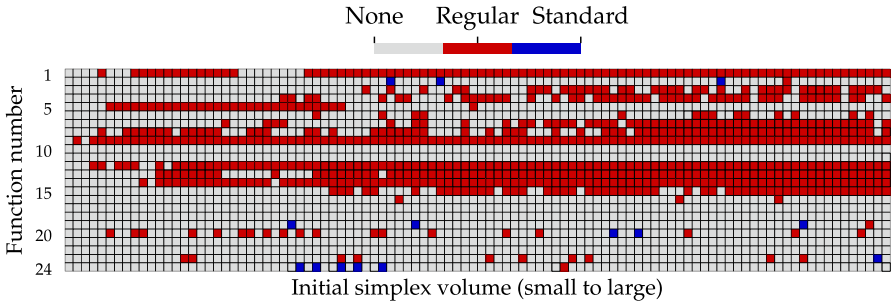
(c) Constraint Handling Method: Reflection



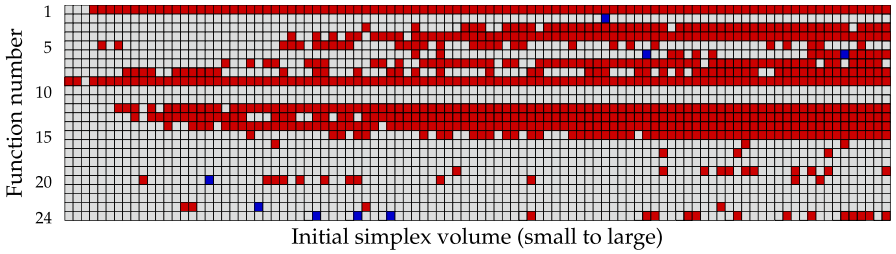
(d) Constraint Handling Method: Wrapping

Fig. 5 Results: mean achieved objective value versus initial simplex volume for $n = 15$. The shadings represent the 95%-confidence intervals computed by the bootstrapping [5] and bias-corrected and accelerated [6] methods

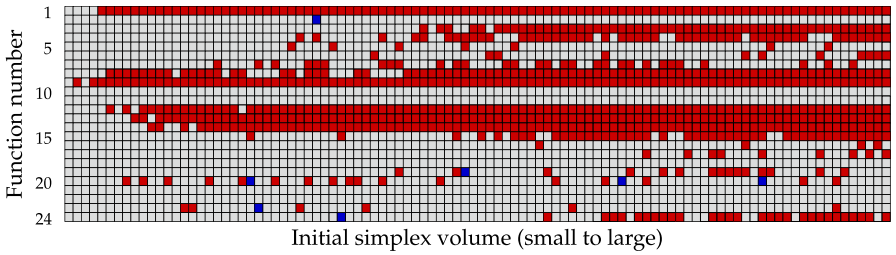
(#10–14), 4. multimodal with an adequate global structure (#15–19), and 5. multimodal with a weak global structure (#20–24), as shown in Table 1. All benchmark functions of BBOB are parameterized, that is, different instances of the same function are available (e.g., low/high-dimensional, translated, and shifted versions) [8]. We prepared the three kinds of dimensions for each benchmark function: $n = 5, 10, 15$. The evaluation budget was set to 400 (including initialization) because, in this study, we assumed that the problems were computationally expensive. Note that, with the Extreme Barrier approach, evaluations of out-of-search-space solutions were not counted because the corresponding actual objective function evaluations were not needed (i.e., their computational costs were negligible) [4]. We evaluated each setting on 100 translated-and-shifted versions of each benchmark function



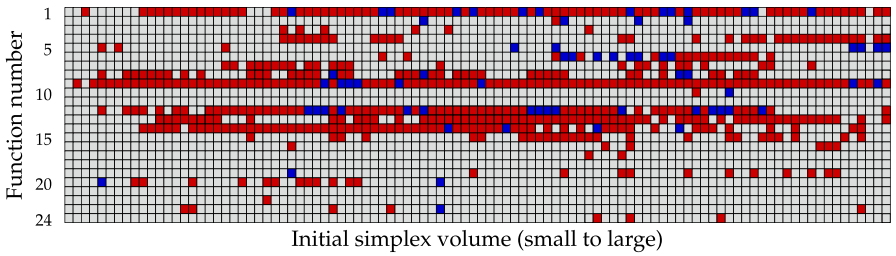
(a) Constraint Handling Method: Extreme Barrier



(b) Constraint Handling Method: Projection



(c) Constraint Handling Method: Reflection



(d) Constraint Handling Method: Wrapping

Fig. 6 Results: regular shape vs. standard shape based on the Wilcoxon rank sum test ($\alpha = 0.05$) for $n = 15$. The color of each square shows a statistically significantly better shape. Gray indicates that the performance difference is not statistically significant. The horizontal and vertical axes denote the volume of the initial simplex and BBOB function number, respectively

to obtain the average performance of the setting and the corresponding 95%-confidence interval. In summary, we collected 24 (benchmark functions) × 3 (dimensions) × 100 (translated-and-shifted versions) × 200 (100 regular + 100 standard initial simplices) × 4 (constraint handling methods) = 5, 760, 000 optimization results.

3.2 Results and discussion

First, we discuss the effects of the initial simplex volume. Figure 5 shows a subset of the experimental results ($n = 15$) of the effect of the initial simplex volume. We focus on the results shown in Fig. 5 because the results for the remaining problems and dimensions share similar trends to them. All the experimental results are available in the Supplementary Material. Five problems (#1 Sphere, #6 Attractive Sector, #10 Ellipsoidal, #15 Rastrigin, and #20 Schwefel) are selected as representatives of each group. We nearly consistently confirm that a larger initial simplex volume results in a better search performance regardless of the benchmark function, shape of the initial simplex, and constraint handling method that was employed. Therefore, regarding our research question Q.1, we conclude that a larger initial simplex is preferable, and the previous results obtained by Wessing [26] can be generalized to a wide range of problems for the limited-evaluation-budget case.

We next discuss the effects of the initial simplex shape. We performed the Wilcoxon rank sum test [15] ($\alpha = 0.05$) to evaluate which shape, regular or standard, is more preferable. Figure 6 shows the results of the statistical tests for $n = 15$. The results for $n = 5$ and 10 were similar to that for $n = 15$ and are available in the Supplementary Material. We find that, in many cases, a regular shape is statistically significantly better than a standard shape regardless of the benchmark function, the volume of the initial simplex, and the constraint handling method that was employed. In particular, this tendency becomes more apparent for unimodal functions and in higher dimensions. This result indicates that the performance of the NM method highly depends not only on the size of the initial simplex but also on its shape. In the end, regarding our research question Q.2, we conclude that a regular initial simplex is preferable for the NM method.

Finally, we discuss the effect of the constraint handling method. As we have observed in Figs. 5 and 6, the tendencies of the effects of the initial simplex volume and shape are nearly consistent, regardless of the constraint handling method that was employed. Therefore, regarding our research question Q.3, we conclude that the proper initial simplex for the NM method is not significantly dependent on the constraint handling method that is employed.

Based on the above discussion, we present a practical initialization heuristic for the NM method for a limited-evaluation-budget case. To maximize the search performance of the NM method, we should employ the initial simplex satisfying the following conditions:

- The size of the initial simplex is as large as possible in the normalized search space.
- The shape of the initial simplex is regular.

We consider this to be the current empirical best practice for practitioners.

4 Conclusion

In this study, we have empirically investigated the effect of the initialization on the NM method for a limited-evaluation-budget case. Our experimental results demonstrated that both the initial simplex size and shape significantly affect the performance of the NM method. We also determined the best practice for the initialization based on the preferable conditions, which practice is not seriously dependent on the constraint handling methods, as indicated by the numerical results.

A possible future direction of this study is to find a practical initialization method for multi- and re-starting cases [11]. In these cases, it may be necessary to generate a variety of simplices rather than a set of regular-shaped large simplices in order to achieve a preferable performance.

We believe that our findings will help practitioners to address real-world problems more efficiently and effectively than previously possible.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11590-022-01953-y.

Acknowledgements We are grateful to the anonymous reviewers for their greatly helpful comments and suggestions. This study is based on the results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO). The computational resources of the AI Bridging Cloud Infrastructure (ABCI) provided by the National Institute of Advanced Industrial Science and Technology (AIST) were used.

Data Availability All experimental data in this study are available from the corresponding author upon request.

References

1. Audet, C., Hare, W.: Derivative-Free and Blackbox Optimization. Springer (2017)
2. Cohen, G., Ruch, P., Hilario, M.: Model selection for support vector classifiers via direct simplex search. In: FLAIRS Conference, pp. 431–435 (2005)
3. Conn, A.R., Scheinberg, K., Vicente, L.N.: Introduction to Derivative-Free Optimization. SIAM (2009)
4. Digabel, S.L., Wild, S.M.: A taxonomy of constraints in simulation-based optimization (2015). arXiv preprint arXiv:1505.07881
5. Efron, B.: The Jackknife, the Bootstrap and Other Resampling Plans. SIAM (1982)
6. Efron, B.: Better bootstrap confidence intervals. J. Am. Stat. Assoc. **82**(397), 171–185 (1987)

7. Fan, E.: Global optimization of the Lennard–Jones atomic cluster. Master's thesis, McMaster University (2002)
8. Finck, S., Hansen, N., Ros, R., Auger, A.: Real-parameter black-box optimization benchmarking 2009. Presentation of the noiseless functions. Technical report. Research Center PPE (2009)
9. Gao, F., Han, L.: Implementing the Nelder–Mead simplex algorithm with adaptive parameters. Comput. Optim. Appl. **51**(1), 259–277 (2012)
10. Ghiasi, H., Pasini, D., Lessard, L.: Constrained globalized Nelder–Mead method for simultaneous structural and manufacturing optimization of a composite bracket. J. Compos. Mater. **42**(7), 717–736 (2008)
11. Hansen, N.: Benchmarking the Nelder-Mead downhill simplex algorithm with many local restarts. In: Proceedings of 11th Annual Conference on Companion Genetic Evolution Computation Conference, Late Breaking Papers, pp. 2403–2408 (2009)
12. Hansen, N., Niederberger, A.S., Guzzella, L., Koumoutsakos, P.: A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Trans. Evol. Comput. **13**(1), 180–197 (2008)
13. Langdon, W.B., Poli, R.: Evolving problems to learn about particle swarm optimizers and other search algorithms. IEEE Trans. Evol. Comput. **11**(5), 561–578 (2007)
14. Lewis, R.M., Torczon, V.: Pattern search algorithms for bound constrained minimization. SIAM J. Optim. **9**(4), 1082–1099 (1999)
15. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. 50–60 (1947)
16. Nash, J.C.: Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation. CRC Press (1990)
17. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**(4), 308–313 (1965)
18. Ozaki, Y., Yano, M., Onishi, M.: Effective hyperparameter optimization using Nelder–Mead method in deep learning. IPSJ Trans. Comput. Vis. Appl. **9**(1), 1–12 (2017)
19. Parks, H.R., Wills, D.C.: An elementary calculation of the dihedral angle of the regular n-simplex. Am. Math. Mon. **109**(8), 756–758 (2002)
20. Purchla, M., Malanowski, M., Terlecki, P., Arabas, J.: Experimental comparison of repair methods for box constraints. In: Proceedings of 7th National Conference on Evolution is a Global Optimization, pp. 135–142. Warsaw Univ. of Technology Publishing House, Warsaw (2004)
21. Shigenaka, S., Takami, S., Ozaki, Y., Onishi, M., Yamashita, T., Noda, I.: Evaluation of optimization for pedestrian route guidance in real-world crowded scene. In: Proceedings of 18th International Conference on Autonomous Agents and Multi-Agent Systems, pp. 2192–2194 (2019)
22. Spendley, W., Hext, G.R., Himsworth, F.R.: Sequential application of simplex designs in optimisation and evolutionary operation. Technometrics **4**(4), 441–461 (1962)
23. Varadhan, R., Borchers, H.: Package dfoptim: derivative-free optimization (2016)
24. Wessing, S.: dfoalgos:derivative-free optimization algorithms (2018). https://pypi.org/project/dfoalgos/
25. Wessing, S.: Repair methods for box constraints revisited. In: EvoApplications, pp. 469–478. Springer, Berlin (2013)
26. Wessing, S.: Proper initialization is crucial for the Nelder–Mead simplex search. Optim. Lett. **13**(4), 847–856 (2019)