mathematics

MDPI

Article

Meta-Optimization of Dimension Adaptive Parameter Schema for Nelder–Mead Algorithm in High-Dimensional Problems

Žiga Rojec, Tadej Tuma, Jernej Olenšek, Árpád Bűrmen and Janez Puhan *

Department of Electronics, Faculty of Electrical Engineering, University of Ljubljana, SI-1000 Ljubljana, Slovenia; ziga.rojec@fe.uni-lj.si (Ž.R.); tadej.tuma@fe.uni-lj.si (T.T.); jernej.olensek@fe.uni-lj.si (J.O.); arpad.buermen@fe.uni-lj.si (Á.B.)

* Correspondence: janez.puhan@fe.uni-lj.si; Tel.: +386-(0)1-4768322

Abstract: Although proposed more than half a century ago, the Nelder–Mead simplex search algorithm is still widely used. Four numeric constants define the operations and behavior of the algorithm. The algorithm with the original constant values performs fine on most low-dimensional, but poorly on high-dimensional, problems. Therefore, to improve its behavior in high dimensions, several adaptive schemas setting the constants according to the problem dimension were proposed in the past. In this work, we present a novel adaptive schema obtained by a meta-optimization procedure. We describe a schema candidate with eight parameters subject to meta-optimization and define an objective function evaluating the candidate's performance. The schema is optimized on up to 100-dimensional problems using the Parallel Simulated Annealing with Differential Evolution global method. The obtained global minimum represents the proposed schema. We compare the performance of the optimized schema with the existing adaptive schemas. The data profiles on the Gao–Han modified quadratic, Moré–Garbow–Hilstrom, and CUTEr (Constrained and Unconstrained Testing Environment, revisited) benchmark problem sets show that the obtained schema outperforms the existing adaptive schemas in terms of accuracy and convergence speed.

Keywords: meta-optimization; Nelder–Mead algorithm; adaptive parameter schema; high-dimensional optimization problems

MSC: 65K05; 90C56

Citation: Rojec, Ž.; Tuma, T.; Olenšek, J.; Bűrmen, Á.; Puhan, J. Meta-Optimization of Dimension Adaptive Parameter Schema for Nelder–Mead Algorithm in High-Dimensional Problems. *Mathematics* **2022**, *10*, 2288. https://doi.org/10.3390/math10132288

Academic Editor: Ioannis G. Tsoulos

Received: 20 May 2022
Accepted: 27 June 2022
Published: 30 June 2022

1. Introduction

Today, we can find optimization algorithms in almost every field of science and technology. A number of optimization algorithms exist, invented to fulfill various requirements regarding convergence rate, precision, robustness, and more. One of them (the Nelder–Mead simplex algorithm [1]), although more than half a century old, is still extensively used for solving a wide range of continuous optimization problems. The algorithm's popularity is due to its simplicity and reasonably good performance observed in practical optimization cases.

Despite its popularity, the algorithm has proven convergence issues. McKinnon [2] presented a family of two-dimensional functions that cause the Nelder–Mead Algorithm (NMA) to converge to a non-stationary point. Galántai [3] provided a sufficient condition for repeated inside contractions in two dimensions, causing non-convergence. A relatively poor theoretical background on algorithm convergence is available for up to two-dimensional problems. Lagarias et al. [4] proved various limited convergence results for two-dimensional strictly convex objective functions. Further, Lagarias et al. [5] proved convergence for a restricted NMA, a version of the NMA without expansion steps, on two-dimensional strictly convex C^2 functions with bounded level sets.

Many modifications of the original NMA were proposed to avoid the algorithm's known deficiencies and provide convergence. Kelley [6] proposed an oriented restart

in case of detected stagnation. Tseng's [7] and Nazareth and Tseng's [8] versions of the algorithm guarantee convergence with a sufficient descent approach. Price et al. [9] used simplex reshaping to achieve convergence on C^1 functions, again satisfying sufficient descent conditions. Bűrmen et al. [10], on the other hand, introduced a grid-restrained version of the NMA, thus making the algorithm a pattern search method. The approach was generalized to a successive approximation of the objective function by Bűrmen and Tuma [11].

However, convergence analysis of the unmodified, original NMA stays a hard mathematical problem. No theoretical background is available above two dimensions. Torczon [12] discovered that the NMA fails because search direction and downhill gradient become orthogonal when the problem dimension is large enough. Wright [13] reported that several scholars observed how the NMA deteriorates with dimensionality but without any explanation. Further, Han and Neumann [14] showed that the NMA makes less and less progress per iteration with an increasing problem dimension. Gao and Han [15] suggested that poor performance in high dimensions is due to an increasing fraction of reflection steps.

Researchers addressed the poor performance of the original NMA in high dimensions in two ways. First, they proposed various algorithm modifications to improve the convergence rate in high-dimensional parameter spaces. Fajfar et al. [16] used genetic programming to evolve a direct search procedure using reflection, expansion, and contraction steps. Musafer's [17] modification adjusts simplex size and direction by performing different NMA steps on various axial combinations. Fajfar et al. [18] proposed random centroid perturbation for improving the search direction, to name a few.

The second approach deals with NMA parameters and does not modify the algorithm itself in any way. Gao and Han [15] proposed the first schema of dimension-dependent NMA parameters to maintain the algorithm's descent property in high dimensions. Kumar and Suri [19] suggested another schema obtained from parameter sensitivity analysis on five test functions. Mehta [20], on the other hand, observed that two schemas based on Chebyshev spaced points outperform Gao–Han's and Kumar–Suri's schemas.

This paper presents the global minimum of the meta-optimization of the NMA adaptive parameter schema. We used the Parallel Simulated Annealing with Differential Evolution (PSADE) robust global optimization method [21] running on a cluster of personal computers as the meta-optimization method. The subjects of meta-optimization are eight coefficients whose values define an individual adaptive parameter Schema Candidate (SC). The schema's mathematical formulation is set in advance and does not evolve during the procedure. We run the NMA using the SC on several test functions and evaluate the SC in each iteration of the meta-optimization. The global minimum represents the best adaptive parameter schema corresponding to the predefined mathematical formulation of the schema and used objective function. We compare the performance of the NMA using the optimized schema with the NMA using the existing schemas on modified quadratic, i.e., Gao–Han (GH) [15], Moré–Garbow–Hilstrom (MGH) [22], and Constrained and Unconstrained Testing Environment, revisited (CUTEr) [23], sets of benchmark problems. Since the proposed adaptive parameter schema results from a meta-optimization procedure, we do not provide a mathematical background explaining the schema's performance. However, the proposed schema outperforms all the other schemas and thus, to the best of our knowledge, currently represents the best dimension adaptive parameter schema for the NMA.

2. Adaptive Parameter Schemas for NMA

Each mathematical symbol used in Sections 2 and 3 is explained at first use. However, for clarification, all the symbols are also listed in the Abbreviations section at the end of the paper.

The original NMA [1,4] is well known, therefore we provide only a brief introduction. The NMA is an unconstrained minimization algorithm operating on an objective function

f of n_p variables handling (n_p+1) points $\mathbf{P}_i$, i.e., simplex vertices, with objective function values $f(\mathbf{P}_i)$, $i = 0, 1 \ldots n_p$. In one iteration of the NMA, the following steps are performed:

- Order and relabel simplex vertices to satisfy $f(\mathbf{P}_0) \leq f(\mathbf{P}_1) \leq \ldots \leq f(\mathbf{P}_{n_p})$. Calculate the centroid point $\overline{\mathbf{P}} = \frac{1}{n_p}\sum_{i=0}^{n_p-1}\mathbf{P}_i$ excluding the highest objective function value point.
- Reflect $\mathbf{P}_{n_p}$ over $\overline{\mathbf{P}}$ to obtain the reflected point $\mathbf{P}_r = \overline{\mathbf{P}} + \alpha(\overline{\mathbf{P}} - \mathbf{P}_{n_p})$, $\alpha > 0$.
- If $f(\mathbf{P}_r) < f(\mathbf{P}_0)$, expand $\mathbf{P}_r$ to obtain the expanded point
  $\mathbf{P}_e = \overline{\mathbf{P}} + \frac{\beta}{\alpha}(\mathbf{P}_r - \overline{\mathbf{P}}) = \overline{\mathbf{P}} + \beta(\overline{\mathbf{P}} - \mathbf{P}_{n_p})$, $\beta > \alpha$.
  If $f(\mathbf{P}_e) < f(\mathbf{P}_r)$, replace $\mathbf{P}_{n_p}$ with $\mathbf{P}_e$ and end the iteration.
- If $f(\mathbf{P}_r) < f(\mathbf{P}_{n_p-1})$, replace $\mathbf{P}_{n_p}$ with $\mathbf{P}_r$ and end the iteration.
- If $f(\mathbf{P}_r) < f(\mathbf{P}_{n_p})$, contract $\mathbf{P}_r$ towards $\overline{\mathbf{P}}$ to obtain the contracted point
  $\mathbf{P}_{rc} = \overline{\mathbf{P}} + \frac{\gamma}{\alpha}(\mathbf{P}_r - \overline{\mathbf{P}}) = \overline{\mathbf{P}} + \gamma(\overline{\mathbf{P}} - \mathbf{P}_{n_p})$, $\gamma < \alpha$.
  If $f(\mathbf{P}_{rc}) < f(\mathbf{P}_{n_p})$, replace $\mathbf{P}_{n_p}$ with $\mathbf{P}_{rc}$ and end the iteration.
- If $f(\mathbf{P}_r) \geq f(\mathbf{P}_{n_p})$, contract $\mathbf{P}_{n_p}$ towards $\overline{\mathbf{P}}$ to obtain the contracted point
  $\mathbf{P}_{nc} = \overline{\mathbf{P}} + \gamma(\mathbf{P}_{n_p} - \overline{\mathbf{P}}) = \overline{\mathbf{P}} - \gamma(\overline{\mathbf{P}} - \mathbf{P}_{n_p})$.
  If $f(\mathbf{P}_{nc}) < f(\mathbf{P}_{n_p})$, replace $\mathbf{P}_{n_p}$ with $\mathbf{P}_{nc}$ and end the iteration.
- Shrink the entire simplex towards $\mathbf{P}_0$, $\mathbf{P}_i := \mathbf{P}_0 + \delta(\mathbf{P}_i - \mathbf{P}_0)$, $\delta < 1$, $i = 1, 2, \ldots n_p$.

NMA iterations are repeated until convergence is achieved. Algorithm behavior depends on α (reflection), β (expansion), γ (contraction), and δ (shrink) parameter values. The NMA default values are

$$\alpha = 1, \quad \beta = 2, \quad \gamma = \frac{1}{2}, \quad \delta = \frac{1}{2} \ . \tag{1}$$

Adaptive parameter schemas define NMA parameter values as functions of the number of variables n_p. The existing schemas (Gao–Han Schema (GHS) [15], Kumar–Suri Schema (KSS) [19], Chebyshev Crude Schema (CCS) [20], and Chebyshev Refined Schema (CRS) [20]) considered in this paper are

$$\begin{aligned}
&\text{GHS:} && \alpha = 1, \quad \beta = 1 + \tfrac{2}{n_p}, \quad \gamma = \tfrac{3}{4} - \tfrac{1}{2n_p}, \quad \delta = 1 - \tfrac{1}{n_p} \\
&\text{KSS:} && \alpha = 1 + \tfrac{3}{5n_p}, \quad \beta = \tfrac{6}{5}, \quad \gamma = \tfrac{19}{20} - \tfrac{3}{n_p} - \tfrac{3}{n_p^2}, \quad \delta = 1 - \tfrac{1}{n_p} \\
&\text{CCS:} && \alpha = 1 + \cos\tfrac{(n_p - 1 - n_p\%2)\pi}{2n_p}, \quad \beta = 1 + \cos\tfrac{(n_p - 3 - n_p\%2)\pi}{2n_p}, \\
& && \gamma = 1 + \cos\tfrac{(n_p + 3 + n_p\%2)\pi}{2n_p}, \quad \delta = 1 + \cos\tfrac{(n_p + 1 + n_p\%2)\pi}{2n_p} \\
&\text{CRS:} && \alpha = 1 + \cos\tfrac{(n_c - 1)\pi}{2n_c}, \quad \beta = 1 + \cos\tfrac{(n_c - 3)\pi}{2n_c}, \\
& && \gamma = 1 + \cos\tfrac{(n_c + 5)\pi}{2n_c}, \quad \delta = 1 + \cos\tfrac{(n_c + 3)\pi}{2n_c}, \quad n_c = 2(9 + \lfloor\tfrac{n_p - 1}{5}\rfloor)
\end{aligned} \tag{2}$$

where % denotes modulo operation, and $\lfloor\rfloor$ the floor function.

In general, the initial simplex vertices $\mathbf{P}_i$ are random. In this paper, however, to assure repeatability of the results, the initial simplex is generated from the starting point $\mathbf{x}_0$ using Pfeffer's method [15]. The first vertex is starting point $\mathbf{P}_0 = \mathbf{x}_0$. The remaining vertices are generated by varying the ith component $\mathbf{P}_i = \mathbf{x}_0 + \epsilon_i \mathbf{e}_i$. $\mathbf{e}_i$ is the ith component unit vector, and ϵ_i is given by

$$\epsilon_i = \begin{cases} 0.05\ \mathbf{x}_0\mathbf{e}_i & \mathbf{x}_0\mathbf{e}_i \neq 0 \\ 0.00025 & \mathbf{x}_0\mathbf{e}_i = 0 \end{cases}, \qquad i = 1, 2 \ldots n_p \ . \tag{3}$$

The starting point $\mathbf{x}_0$ is $[1, 1 \ldots 1]^{\mathrm{T}}$ for GH benchmarks [15], and as given in [22] for MGH benchmarks.

A Nelder–Mead run terminates when the simplex becomes too flat or shrinks below a certain size. In this paper, we use tolerances Tol_f for simplex flatness and Tol_X for simplex size. A Nelder–Mead run stops when both criteria (4) are met:

$$\max_{i=1}^{n_p} |f(\mathbf{P}_i) - f(\mathbf{P}_0)| < Tol_f \qquad \max_{j=0}^{n_p-1} \max_{i=1}^{n_p} |P_{ij} - P_{0j}| < Tol_X, \tag{4}$$

where $\mathbf{P}_i = [P_{i0}, P_{i1}, \ldots P_{i(n_p-1)}]^T$ is the ith vertex of the simplex.

3. Optimization of the Adaptive Parameter Schema

The default (1) and adaptive parameter schema functions (2) shown in Figure 1 have in general similar behavior. By choosing appropriate values c_{0p} and c_{1p}, an individual parameter c from a particular schema could be closely fitted with function $c = c_{0p} + \frac{c_{1p}}{n}$.

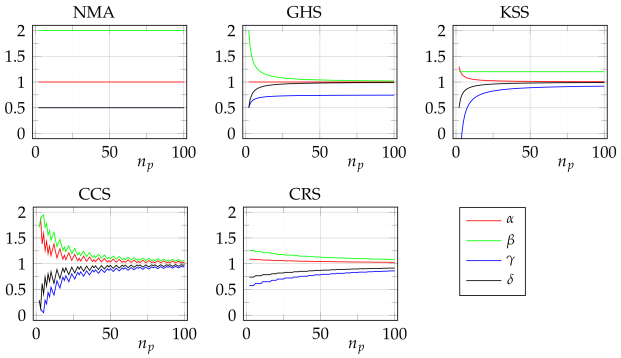

Figure 1. Parameter schema functions for the original Nelder–Mead Algorithm (NMA), Gao–Han Schema (GHS), Kumar–Suri Schema (KSS), Chebyshev Crude Schema (CCS), and Chebyshev Refined Schema (CRS).

At this point, a question arises: can a better adaptive parameter schema of the formulation (5) be obtained by choosing the right values for $c_{0\alpha}$, $c_{1\alpha}$, $c_{0\beta}$, $c_{1\beta}$, etc.? Do such values exist, and what are they? A meta-optimization procedure could provide some answers.

$$\alpha = c_{0\alpha} + \frac{c_{1\alpha}}{n_p}, \quad \beta = c_{0\beta} + \frac{c_{1\beta}}{n_p}, \quad \gamma = c_{0\gamma} + \frac{c_{1\gamma}}{n_p}, \quad \delta = c_{0\delta} + \frac{c_{1\delta}}{n_p} \tag{5}$$

Let us first define the meta-optimization procedure. As in any optimization, we need optimization parameters, an objective function, and an optimization method. Optimization parameters ($c_{0\alpha} \ldots c_{1\delta}$) follow from the mathematical formulation describing an SC (5). Therefore, we have an eight-dimensional meta-optimization parameter space.

The meta-optimization objective function measures the weighted difference in data profiles [24] between the best reference schema and an SC. For a mathematical formulation of the objective function, some definitions are needed. A data profile function of a schema s over a set of benchmark problems $\mathcal{P}$

$$d_{s\mathcal{P}}(\kappa) = \frac{|p \in \mathcal{P} : \frac{t_{ps}}{n_p+1} \leq \kappa|}{|\mathcal{P}|} \tag{6}$$

defines the fraction of problems in a set $\mathcal{P}$ that schema s solves in κ simplex gradient estimates. s is a schema from the set of schemas $\mathcal{S}$ ($s \in \mathcal{S} = \{\text{NMA, GHS, KSS, CCS, CRS, SC}\}$). $|\cdot|$ denotes cardinality of a set. t_{ps} is the number of objective function evaluations needed by schema s to achieve convergence on problem p, and n_p is the problem dimension. Since one simplex gradient estimate corresponds to $n_p + 1$ objective function evaluations, fraction

$\frac{t_{ps}}{n_p+1}$ is the number of simplex gradient estimates required for convergence. Convergence is achieved when

$$f(\mathbf{x}) \leq f_{\mathrm{L}} + \tau(f(\mathbf{x_0}) - f_{\mathrm{L}}) \,, \tag{7}$$

where $\mathbf{x}$ is an evaluated point in n_p-dimensional parameter space, and f_{L} is the lowest objective function value reached by any of the schemas $s \in \mathcal{S}$ in $\kappa_{\max}$ simplex gradient estimates. Tolerance τ specifies accuracy level. Moré and Wild [24] use tolerance values 10^{-1}, 10^{-3}, 10^{-5}, and 10^{-7}. We set convergence condition tolerance τ to 10^{-7} as Mehta did in his work [20]. If a particular schema s fails to satisfy condition (7) for problem p, then t_{ps} is set to infinity.

The final meta-optimization objective function $h(\mathrm{SC})$ evaluates a particular SC defined by eight meta-optimization parameters $c_{0\alpha} \ldots c_{1\delta}$ with

$$h(\mathrm{SC}) = \sum_{\mathcal{P}\in\mathcal{X}} \sum_{\kappa=1}^{\kappa_{\max}} \left((\max_{r\in\mathcal{R}} d_{r\mathcal{P}}(\kappa) - d_{\mathrm{SC}\mathcal{P}}(\kappa)) \times \begin{cases} w_{\mathcal{P}_+} & \max\limits_{r\in\mathcal{R}} d_{r\mathcal{P}}(\kappa) > d_{\mathrm{SC}\mathcal{P}}(\kappa) \\ w_{\mathcal{P}_-} & \max\limits_{r\in\mathcal{R}} d_{r\mathcal{P}}(\kappa) \leq d_{\mathrm{SC}\mathcal{P}}(\kappa) \end{cases} \right) . \tag{8}$$

The GH and MGH benchmark problems are treated separately, $\mathcal{X} \in \{\mathrm{GH}, \mathrm{MGH}\}$. For every number of simplex gradient estimates κ, the data profile of the SC is compared with the best of the reference profiles, $r \in \mathcal{R} = \mathcal{S} - \{\mathrm{SC}\}$. The difference is weighted with $w_{\mathcal{P}_+}$ when the best reference is better, and $w_{\mathcal{P}_-}$ otherwise. Weights $w_{\mathcal{P}_+}$ and $w_{\mathcal{P}_-}$ define a trade-off between under- and over-achieving the optimization goal (which in turn is the best schema's performance). Usually, under-achieving is penalized more than over-achieving is rewarded. In our case, the weight values were set to $w_{\mathrm{GH}_+} = w_{\mathrm{MGH}_+} = 10$, and $w_{\mathrm{GH}_-} = w_{\mathrm{MGH}_-} = 1$.

Finally, we have to choose an optimization method to perform our meta-optimization procedure. We are searching for a global minimum in eight-dimensional parameter space. Therefore, a global optimization method with proven convergence can do the job. We chose PSADE [21], a parallel version of [25] since it is available in the PyOPUS Python package [26]. Among others, the PyOPUS package includes optimization algorithms (original NMA included), parallel processing support, and benchmark problems (GH, MGH, and CUTEr problems included), all the ingredients needed in our meta-optimization procedure. It can be found in the Python Package Index (PyPI) software repository. The PSADE method exhibited good performance on global benchmark functions as well as on real optimization problems [27–29]. Further, PSADE is an asynchronous global method achieving speedups up to the number of slave computational cores when run in parallel. We ran PSADE on a cluster of 25 personal computers. Instead of PSADE, one of a plethora of newer global methods, e.g., [30–33], could be used. However, besides faster convergence, we do not expect significantly better results.

A meta-optimization search for a better NMA parameter schema requires significant computational power. An individual SC, represented by eight meta-optimization parameter values $c_{0\alpha} \ldots c_{1\delta}$, has to be evaluated against reference parameter schemas in every meta-optimization iteration. One SC evaluation requires as many Nelder–Mead runs as there are problems included in the objective function evaluation. In general, a single Nelder–Mead run stops when the termination criteria are achieved, e.g., when the simplex becomes flat or shrinks below the tolerance. However, additional improvement is possible if the procedure runs further. When driven beyond tolerances, a non-convergent SC may become convergent, although rather slow. Gao and Han [15], Kumar and Suri [19], and Mehta [20] all set an absolute limit to the number of objective function evaluations to 10^6, i.e., 9900–90,909 simplex gradient estimates. However, in their results, 1000–2000 estimates are needed on average for the NMA using an adaptive parameter schema to converge. They set termination tolerances Tol_{f}, Tol_{X} in range 10^{-10}–10^{-4}. Therefore, after some preliminary tests, we set $\kappa_{\max}$ to 5000. More would be better. And—after some preliminary experimental optimization runs—we established that around 10^6 meta-optimization iterations are required to achieve convergence in an unconstrained eight-dimensional parameter space

using the robust PSADE global optimization method [21]. Again, the more, the better. Thus, the total number of required objective function evaluations can be estimated as the number of simplex gradient estimates per benchmark problem times the number of simplex vertices summed over all benchmark problems times the number of meta-optimization iterations. For GH and MGH benchmarks altogether, this is over 10^{13} evaluations.

Therefore, brute force is not very promising. Instead, we tuned the meta-optimization parameters by conducting a series of shorter meta-optimization runs. We varied parameter space constraints, NMA termination tolerances Tol_f and Tol_X, the gradient estimates limit $\kappa_{\max}$ per Nelder–Mead run, meta-optimization iteration limit, and we also experimented with the objective function definition at the beginning. By observing the results, we gradually eliminated parts that were not essential, e.g., setting NMA termination tolerances Tol_f and Tol_X too low can significantly extend the meta-optimization procedure without producing a significantly better result. The final meta-optimization parameter values are as follows: the parameter space is a discrete eight-dimensional box with 0.01 grid and constraints set to $[c_{0\alpha}, c_{1\alpha}, c_{0\beta}, c_{1\beta}, c_{0\gamma}, c_{1\gamma}, c_{0\delta}, c_{1\delta}]^T \in [[0.80, 1.20], [0.20, 0.60], [0.85, 1.25], [0.35, 0.75], [0.65, 1.05], [-0.50, -0.10], [0.05, 0.45], [-0.40, 0.00]]^T$. NMA termination tolerances are set to $Tol_f = Tol_X = 10^{-4}$. The simplex gradient estimates limit was set to $\kappa_{\max} = 5000$, enough to catch degenerated SCs. The final objective function formulation is given in (8). Meta-optimization iteration limit was set to 10^6.

The final meta-optimized parameter values, i.e., the global minimum of (8), are

$$\begin{array}{llll} c_{0\alpha} = 1.02, & c_{1\alpha} = 0.31, & c_{0\beta} = 1.06, & c_{1\beta} = 0.53, \\ c_{0\gamma} = 0.82, & c_{1\gamma} = -0.27, & c_{0\delta} = 0.28, & c_{1\delta} = -0.19\,. \end{array} \tag{9}$$

This is the global minimum when (5) is used as the adaptive parameter schema. We chose (5) as the adaptive parameter schema because of its similarity to the existing schemas.

4. Evaluation of the Optimized Schema with Discussion

In this section, we present and discuss the properties of the optimized schema (9). We analyze its performance and compare it with the other schemas, (1) and (2). Since the schema is a result of the meta-optimization procedure, we do not deal with the issue of why the schema performs well. A mathematical evaluation of an arbitrary schema is given with the objective function definition (8). However, an analytical solution to the defined meta-optimization problem exceeds the scope of this paper.

The optimized parameter schema (9) is compared with the original NMA (1) and the existing adaptive parameter schemas (2), GHS, KSS, CCS, and CRS, in Figure 2. We can observe that the optimized schema follows the same pattern as the other schemas. However, the optimized schema approaches its high-dimensional value faster which is the consequence of smaller $|c_{1p}/c_{0p}|$ ratios. Otherwise, the optimized schema curves do not significantly deviate from the others. Shrink parameter δ is an exception. While δ in all the other adaptive schemas approaches 1, the optimized schema's high-dimensional δ value is $c_{0\delta} = 0.28$ which is far lower, even than 0.5 of the original NMA. On the other hand, the shrink parameter turns out to be insignificant for GH and MGH benchmarks. The fraction of shrink steps is at most 0.5% for Penalty I and Penalty II problems. However, the fraction of shrink steps in the optimized schema is 0.0% for all GH and MGH benchmarks.

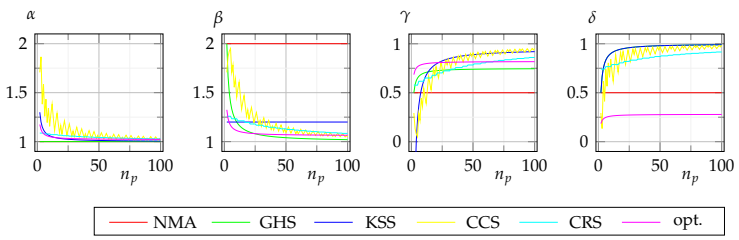

Figure 2. Comparison of NMA adaptive schema parameters.

To evaluate the obtained schema further, we compare it in terms of accuracy and speed. Note that the meta-optimization objective function (8) in combination with NMA termination tolerances rewards speed. Therefore, accuracy was not a subject of meta-optimization.

Table 1 shows accuracy results for a GH modified quadratic function (10) up to problem dimension $n_p = 100$. Parameter $\epsilon \geq 0$ defines condition number of matrix $\mathbf{D}$, and $\sigma \geq 0$ specifies deviation from quadratic form. The minimum value of (10) is zero ($\min_{\mathbf{x}\in\mathbb{R}^{n_p}} f(\mathbf{x}) = 0$) for any ϵ, σ, or n_p. An individual Nelder–Mead run was stopped after $\kappa_{\max}$ = 25,000 simplex gradient estimates, that is, after $N_{\text{eval}} = 25,000(n_p + 1)$ objective function evaluations. We applied no tolerance-based termination criteria, i.e., $Tol_{\text{f}} = Tol_{\text{X}} = 0$. The table shows the lowest achieved objective function values. A schema is considered to be accurate if the achieved minimum value is correct to at least six decimal places, i.e., when $f(\mathbf{x}) < 5 \times 10^{-7}$. The accurate schema values are shown in bold. If a schema does not converge according to condition (7) for $\tau = 10^{-7}$, the value is shown in italics.

$$f(\mathbf{x}) = \mathbf{x}^{\text{T}}\mathbf{D}\mathbf{x} + \sigma(\mathbf{x}^{\text{T}}\mathbf{B}\mathbf{x})^2$$
$$\mathbf{D} = \text{diag}[(1+\epsilon), (1+\epsilon)^2, \ldots (1+\epsilon)^{n_p}], \quad \mathbf{B} = \mathbf{U}^{\text{T}}\mathbf{U}, \quad \mathbf{U} = \begin{bmatrix} 1 & \cdots & 1 \\ & \ddots & \vdots \\ & & 1 \end{bmatrix} \tag{10}$$

Given 25,000 simplex gradient estimates, the optimized schema is accurate and convergent for all 40 GH benchmark problems, as all the existing schemas also are. As expected, the original NMA encounters accuracy and convergence problems in high dimensions.

Accuracy results for MGH benchmark problems [22] are shown in Table 2. The same rules (25,000 simplex gradient estimates, $Tol_{\text{f}} = Tol_{\text{X}} = 0$, etc.) apply. Minima of the used MGH functions are all zero with the exception of Penalty I and Penalty II functions. For $n_p = 10$, the corresponding minima are 7.0876515... $\times 10^{-5}$ and 0.00029366054..., respectively. Thus, the six digit accuracy criteria are given by $f_{\text{Penalty I}}(\mathbf{x}) < 7.087655 \times 10^{-5}$, and $f_{\text{Penalty II}}(\mathbf{x}) < 0.0002936615$.

Table 1. Accuracy of the original NMA, the existing adaptive schemas (GHS, KSS, CCS, CRS), and the optimized schema on Gao–Han (GH) modified quadratic benchmark problems.

	n_p	$f(\mathbf{x})_{\text{NMA}}$	$f(\mathbf{x})_{\text{GHS}}$	$f(\mathbf{x})_{\text{KSS}}$	$f(\mathbf{x})_{\text{CCS}}$	$f(\mathbf{x})_{\text{CRS}}$	$f(\mathbf{x})_{\text{opt.}}$
$\epsilon = 0.0$	10	$\mathbf{3.5 \times 10^{-323}}$	**0.0**	**0.0**	$\mathbf{3.5 \times 10^{-323}}$	**0.0**	**0.0**
$\sigma = 0.0$	20	$\mathbf{2 \times 10^{-322}}$	**0.0**	**0.0**	**0.0**	**0.0**	$\mathbf{10^{-323}}$
	30	$\mathbf{1.14 \times 10^{-11}}$	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	40	2.03×10^{-4}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	50	5.54×10^{-4}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	60	1.38×10^{-5}	**0.0**	**0.0**	$\mathbf{5 \times 10^{-324}}$	**0.0**	$\mathbf{5 \times 10^{-324}}$
	70	5.76×10^{-5}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	80	4.87×10^{-6}	$\mathbf{5 \times 10^{-323}}$	**0.0**	**0.0**	**0.0**	**0.0**
	90	2.75×10^{-6}	$\mathbf{1.4 \times 10^{-322}}$	**0.0**	$\mathbf{5 \times 10^{-324}}$	**0.0**	**0.0**
	100	3.19×10^{-6}	$\mathbf{6 \times 10^{-323}}$	**0.0**	$\mathbf{2 \times 10^{-323}}$	**0.0**	**0.0**
$\epsilon = 0.05$	10	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
$\sigma = 0.0$	20	$\mathbf{6.23 \times 10^{-322}}$	**0.0**	**0.0**	**0.0**	$\mathbf{5 \times 10^{-324}}$	**0.0**
	30	5.31×10^{-3}	**0.0**	**0.0**	**0.0**	**0.0**	$\mathbf{5 \times 10^{-324}}$
	40	1.32×10^{-2}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	50	1.62×10^{-1}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	60	12.7	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	70	8.24	$\mathbf{2 \times 10^{-323}}$	**0.0**	**0.0**	**0.0**	**0.0**
	80	32.2	$\mathbf{1.24 \times 10^{-322}}$	**0.0**	$\mathbf{5 \times 10^{-324}}$	**0.0**	**0.0**
	90	3.77	$\mathbf{5.4 \times 10^{-323}}$	$\mathbf{5 \times 10^{-324}}$	$\mathbf{5 \times 10^{-324}}$	**0.0**	**0.0**
	100	278	$\mathbf{6 \times 10^{-323}}$	**0.0**	$\mathbf{10^{-323}}$	**0.0**	**0.0**

Table 1. *Cont.*

	n_p	$f(\mathbf{x})_{\mathrm{NMA}}$	$f(\mathbf{x})_{\mathrm{GHS}}$	$f(\mathbf{x})_{\mathrm{KSS}}$	$f(\mathbf{x})_{\mathrm{CCS}}$	$f(\mathbf{x})_{\mathrm{CRS}}$	$f(\mathbf{x})_{\mathrm{opt.}}$
$\epsilon = 0.0$	10	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
$\sigma = 0.0001$	20	2.05×10^{-3}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	30	1.91×10^{-5}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	40	*16.7*	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	50	*2.63*	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	60	*10.9*	**0.0**	**0.0**	$\mathbf{10^{-323}}$	**0.0**	**0.0**
	70	*276*	$\mathbf{5 \times 10^{-324}}$	**0.0**	**0.0**	**0.0**	**0.0**
	80	*292*	$\mathbf{8 \times 10^{-323}}$	**0.0**	**0.0**	**0.0**	**0.0**
	90	*11.7*	$\mathbf{6 \times 10^{-323}}$	$\mathbf{5 \times 10^{-324}}$	$\mathbf{5 \times 10^{-324}}$	**0.0**	**0.0**
	100	*48.7*	$\mathbf{7.4 \times 10^{-323}}$	**0.0**	$\mathbf{10^{-323}}$	**0.0**	**0.0**
$\epsilon = 0.05$	10	$\mathbf{1.5 \times 10^{-323}}$	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
$\sigma = 0.0001$	20	1.93×10^{-4}	**0.0**	**0.0**	**0.0**	$\mathbf{5 \times 10^{-324}}$	$\mathbf{5 \times 10^{-324}}$
	30	1.12×10^{-2}	**0.0**	**0.0**	**0.0**	**0.0**	$\mathbf{5 \times 10^{-324}}$
	40	7.31×10^{-1}	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	50	*37.2*	**0.0**	**0.0**	**0.0**	**0.0**	$\mathbf{5 \times 10^{-324}}$
	60	*179*	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	70	*18.7*	$\mathbf{3 \times 10^{-323}}$	**0.0**	**0.0**	**0.0**	**0.0**
	80	*16.4*	$\mathbf{7.4 \times 10^{-323}}$	**0.0**	$\mathbf{5 \times 10^{-324}}$	**0.0**	**0.0**
	90	*1480*	$\mathbf{1.3 \times 10^{-322}}$	$\mathbf{1.5 \times 10^{-323}}$	**0.0**	**0.0**	**0.0**
	100	*3802*	$\mathbf{5 \times 10^{-323}}$	**0.0**	$\mathbf{10^{-323}}$	**0.0**	**0.0**
accurate		7/40	40/40	40/40	40/40	40/40	40/40

Again, with 25,000 simplex gradient estimates available, the optimized schema is accurate and convergent for all MGH benchmarks, except for four trigonometric functions, where the result is approaching the minimum, although not reaching it. We can observe similar behavior for the other schemas as well. Furthermore, for trigonometric function benchmarks, the schemas achieved relatively small objective function value reduction, which is reflected in relation $\tau\frac{f(\mathbf{x}_0)}{f_{\mathrm{L}}} \ll 1$. Consequently, $f_{\mathrm{L}} \gg \tau(f(\mathbf{x_0}) - f_{\mathrm{L}})$, and convergence condition (7) degenerates into $f(\mathbf{x}) \lesssim f_{\mathrm{L}}$. Strictly following the convergence condition, only schemas reaching the lowest value f_{L} are considered convergent. Nevertheless, the optimized schema managed to produce the lowest objective function value in all trigonometric function benchmarks, except one.

Convergence speed of the obtained schema (9) is compared to the other schemas (1) and (2) using data profiles (6) and (7) [24]. Figure 3 shows data profiles for GH and MGH benchmark sets separately and combined. The profiles are calculated at $\tau = 10^{-7}$ with $\kappa_{\max}$ = 25,000 simplex gradient estimates and without tolerance-based algorithm termination ($Tol_{\mathrm{f}} = Tol_{\mathrm{X}} = 0$). The graphs are shown for up to 15,000 simplex gradient estimates.

Graphs in Figure 3 reveal a slight advantage in terms of speed for the optimized schema over the existing adaptive parameter schemas. The schemas perform similarly when only GH benchmarks are considered. Although the optimized schema can be declared as the fastest (solves the highest percentage of problems at almost any given κ), the remaining schemas quickly follow. All the adaptive schemas solve 100% of GH problems after approximately 2500 simplex gradient estimates. The CCS is the first to achieve this goal at ~2100 simplex gradient estimates. The original NMA, on the other hand, manages to be competitive with the adaptive schemas for the first 10% of problems. With the lower-dimensional problems solved, it starts to lag, finally solving less than 30% of GH problems.

Table 2. Accuracy of the original NMA, the existing adaptive schemas (GHS, KSS, CCS, CRS), and the optimized schema on Moré–Garbow–Hilstrom (MGH) benchmark problems.

Function	n_p	$f(\mathbf{x})_{\mathbf{NMA}}$	$f(\mathbf{x})_{\mathbf{GHS}}$	$f(\mathbf{x})_{\mathbf{KSS}}$	$f(\mathbf{x})_{\mathbf{CCS}}$	$f(\mathbf{x})_{\mathbf{CRS}}$	$f(\mathbf{x})_{\mathbf{opt.}}$
Extended Rosenbrock	12	$\mathbf{2.91 \times 10^{-28}}$	$\mathbf{2.35 \times 10^{-29}}$	$\mathbf{1.73 \times 10^{-28}}$	$\mathbf{4.64 \times 10^{-29}}$	$\mathbf{5.18 \times 10^{-29}}$	$\mathbf{6.88 \times 10^{-29}}$
	18	*20.0*	$\mathbf{6.97 \times 10^{-29}}$	$\mathbf{6.51 \times 10^{-29}}$	$\mathbf{1.44 \times 10^{-28}}$	$\mathbf{5.66 \times 10^{-28}}$	$\mathbf{1.35 \times 10^{-28}}$
	24	*12.5*	$\mathbf{1.72 \times 10^{-28}}$	$\mathbf{2.58 \times 10^{-28}}$	$\mathbf{1.86 \times 10^{-28}}$	$\mathbf{3.72 \times 10^{-28}}$	$\mathbf{6.21 \times 10^{-28}}$
	30	*34.5*	$\mathbf{4.09 \times 10^{-28}}$	$\mathbf{6.70 \times 10^{-28}}$	$\mathbf{7.28 \times 10^{-28}}$	$\mathbf{3.70 \times 10^{-27}}$	$\mathbf{2.76 \times 10^{-28}}$
	36	*49.1*	$\mathbf{8.72 \times 10^{-28}}$	$\mathbf{6.64 \times 10^{-28}}$	$\mathbf{6.81 \times 10^{-28}}$	$\mathbf{4.77 \times 10^{-28}}$	$\mathbf{1.11 \times 10^{-27}}$
Extended Powell singular	12	$\mathbf{8.34 \times 10^{-55}}$	$\mathbf{3.33 \times 10^{-57}}$	$\mathbf{7.09 \times 10^{-55}}$	$\mathbf{3.09 \times 10^{-57}}$	$\mathbf{1.07 \times 10^{-59}}$	$\mathbf{1.18 \times 10^{-58}}$
	24	$\mathbf{1.33 \times 10^{-9}}$	$\mathbf{1.83 \times 10^{-54}}$	$\mathbf{3.45 \times 10^{-56}}$	$\mathbf{5.37 \times 10^{-56}}$	$\mathbf{1.67 \times 10^{-53}}$	$\mathbf{3.39 \times 10^{-53}}$
	40	1.69×10^{-6}	$\mathbf{1.06 \times 10^{-50}}$	$\mathbf{2.34 \times 10^{-52}}$	$\mathbf{1.46 \times 10^{-52}}$	$\mathbf{5.22 \times 10^{-52}}$	$\mathbf{2.33 \times 10^{-53}}$
	60	*4.16 × 10⁻⁴*	9.71×10^{-6}	$\mathbf{3.43 \times 10^{-50}}$	$\mathbf{2.88 \times 10^{-52}}$	$\mathbf{1.28 \times 10^{-37}}$	$\mathbf{1.16 \times 10^{-46}}$
Penalty I	10	7.57×10^{-5}	$\mathbf{7.09 \times 10^{-5}}$	$\mathbf{7.09 \times 10^{-5}}$	7.60×10^{-5}	$\mathbf{7.09 \times 10^{-5}}$	$\mathbf{7.09 \times 10^{-5}}$
Penalty II	10	2.98×10^{-4}	$\mathbf{2.94 \times 10^{-4}}$	$\mathbf{2.94 \times 10^{-4}}$	2.98×10^{-4}	2.95×10^{-4}	$\mathbf{2.94 \times 10^{-4}}$
Variably dimensioned	12	*4.77*	$\mathbf{3.72 \times 10^{-30}}$	$\mathbf{1.47 \times 10^{-29}}$	$\mathbf{3.64 \times 10^{-29}}$	$\mathbf{2.30 \times 10^{-29}}$	$\mathbf{1.78 \times 10^{-29}}$
	18	4.22	$\mathbf{8.96 \times 10^{-30}}$	$\mathbf{2.06 \times 10^{-29}}$	$\mathbf{1.52 \times 10^{-29}}$	$\mathbf{4.74 \times 10^{-29}}$	$\mathbf{4.25 \times 10^{-29}}$
	24	11.5	$\mathbf{8.22 \times 10^{-29}}$	$\mathbf{8.37 \times 10^{-29}}$	$\mathbf{7.52 \times 10^{-29}}$	$\mathbf{9.23 \times 10^{-29}}$	$\mathbf{2.27 \times 10^{-28}}$
	30	40.5	$\mathbf{8.08 \times 10^{-29}}$	$\mathbf{1.08 \times 10^{-28}}$	$\mathbf{1.06 \times 10^{-28}}$	$\mathbf{3.38 \times 10^{-28}}$	$\mathbf{4.49 \times 10^{-28}}$
	36	60.1	$\mathbf{4.21 \times 10^{-28}}$	$\mathbf{1.46 \times 10^{-28}}$	$\mathbf{8.82 \times 10^{-29}}$	$\mathbf{8.35 \times 10^{-28}}$	$\mathbf{7.60 \times 10^{-28}}$
Trigonometric	10	2.80×10^{-5}	2.80×10^{-5}	2.80×10^{-5}	2.80×10^{-5}	2.80×10^{-5}	2.80×10^{-5}
	20	1.35×10^{-6}	1.35×10^{-6}	*6.03 × 10⁻⁶*	*6.86 × 10⁻⁶*	1.35×10^{-6}	1.35×10^{-6}
	30	*2.20 × 10⁻⁵*	*9.90 × 10⁻⁷*	*9.90 × 10⁻⁷*	*5.65 × 10⁻⁶*	*9.90 × 10⁻⁷*	5.98×10^{-7}
	40	*1.41 × 10⁻⁵*	*1.55 × 10⁻⁶*	*3.95 × 10⁻⁶*	$\mathbf{1.68 \times 10^{-7}}$	*5.58 × 10⁻⁷*	*1.55 × 10⁻⁶*
	50	*2.52 × 10⁻⁵*	$\mathbf{2.24 \times 10^{-7}}$	*3.41 × 10⁻⁶*	*9.23 × 10⁻⁷*	*1.11 × 10⁻⁶*	$\mathbf{2.24 \times 10^{-7}}$
	60	*3.87 × 10⁻⁵*	*8.68 × 10⁻⁷*	*7.57 × 10⁻⁷*	*7.57 × 10⁻⁷*	*1.27 × 10⁻⁶*	$\mathbf{4.62 \times 10^{-7}}$
Discrete boundary value	10	$\mathbf{6.85 \times 10^{-32}}$	$\mathbf{3.03 \times 10^{-33}}$	$\mathbf{8.36 \times 10^{-32}}$	$\mathbf{2.20 \times 10^{-31}}$	$\mathbf{3.07 \times 10^{-32}}$	$\mathbf{1.59 \times 10^{-32}}$
	20	$\mathbf{4.69 \times 10^{-30}}$	$\mathbf{7.24 \times 10^{-32}}$	$\mathbf{2.51 \times 10^{-32}}$	$\mathbf{2.39 \times 10^{-32}}$	$\mathbf{1.05 \times 10^{-31}}$	$\mathbf{3.92 \times 10^{-32}}$
	30	*9.87 × 10⁻⁶*	$\mathbf{1.10 \times 10^{-31}}$	$\mathbf{1.43 \times 10^{-31}}$	$\mathbf{1.19 \times 10^{-31}}$	$\mathbf{2.02 \times 10^{-31}}$	$\mathbf{9.29 \times 10^{-32}}$
	40	*6.46 × 10⁻⁶*	$\mathbf{4.58 \times 10^{-31}}$	$\mathbf{3.55 \times 10^{-31}}$	$\mathbf{1.37 \times 10^{-31}}$	$\mathbf{4.76 \times 10^{-31}}$	$\mathbf{3.45 \times 10^{-31}}$
	50	*5.72 × 10⁻⁶*	$\mathbf{6.02 \times 10^{-31}}$	$\mathbf{5.70 \times 10^{-31}}$	$\mathbf{2.84 \times 10^{-31}}$	$\mathbf{5.35 \times 10^{-31}}$	$\mathbf{4.35 \times 10^{-31}}$
	60	*3.19 × 10⁻⁶*	$\mathbf{2.46 \times 10^{-30}}$	$\mathbf{8.09 \times 10^{-31}}$	$\mathbf{6.64 \times 10^{-31}}$	$\mathbf{1.11 \times 10^{-30}}$	$\mathbf{7.39 \times 10^{-31}}$
Discrete integral equation	10	$\mathbf{1.91 \times 10^{-31}}$	$\mathbf{4.24 \times 10^{-33}}$	$\mathbf{1.44 \times 10^{-32}}$	$\mathbf{2.27 \times 10^{-31}}$	$\mathbf{2.56 \times 10^{-32}}$	$\mathbf{3.08 \times 10^{-33}}$
	20	$\mathbf{7.69 \times 10^{-30}}$	$\mathbf{4.62 \times 10^{-32}}$	$\mathbf{2.90 \times 10^{-32}}$	$\mathbf{6.27 \times 10^{-32}}$	$\mathbf{3.40 \times 10^{-32}}$	$\mathbf{2.37 \times 10^{-32}}$
	30	*7.11 × 10⁻⁴*	$\mathbf{2.22 \times 10^{-31}}$	$\mathbf{2.50 \times 10^{-31}}$	$\mathbf{3.45 \times 10^{-31}}$	$\mathbf{8.55 \times 10^{-32}}$	$\mathbf{2.50 \times 10^{-31}}$
	40	*3.63 × 10⁻⁴*	$\mathbf{3.82 \times 10^{-31}}$	$\mathbf{3.07 \times 10^{-31}}$	$\mathbf{3.21 \times 10^{-31}}$	$\mathbf{4.25 \times 10^{-31}}$	$\mathbf{3.04 \times 10^{-31}}$
	50	*3.05 × 10⁻³*	$\mathbf{8.51 \times 10^{-31}}$	$\mathbf{1.34 \times 10^{-30}}$	$\mathbf{5.95 \times 10^{-31}}$	$\mathbf{1.47 \times 10^{-30}}$	$\mathbf{7.34 \times 10^{-31}}$
	60	*4.46 × 10⁻⁴*	$\mathbf{2.24 \times 10^{-30}}$	$\mathbf{1.30 \times 10^{-30}}$	$\mathbf{1.59 \times 10^{-30}}$	$\mathbf{9.74 \times 10^{-31}}$	$\mathbf{6.12 \times 10^{-31}}$
Broyden tridiagonal	10	$\mathbf{3.99 \times 10^{-30}}$	$\mathbf{3.12 \times 10^{-30}}$	$\mathbf{7.31 \times 10^{-30}}$	$\mathbf{3.28 \times 10^{-29}}$	$\mathbf{2.92 \times 10^{-30}}$	$\mathbf{2.92 \times 10^{-30}}$
	20	$\mathbf{3.20 \times 10^{-26}}$	$\mathbf{1.63 \times 10^{-29}}$	$\mathbf{3.34 \times 10^{-29}}$	$\mathbf{6.15 \times 10^{-29}}$	$\mathbf{2.45 \times 10^{-29}}$	$\mathbf{3.15 \times 10^{-29}}$
	30	$\mathbf{4.70 \times 10^{-26}}$	$\mathbf{1.68 \times 10^{-28}}$	$\mathbf{1.19 \times 10^{-28}}$	$\mathbf{9.66 \times 10^{-29}}$	$\mathbf{6.50 \times 10^{-29}}$	$\mathbf{8.18 \times 10^{-29}}$
	40	$\mathbf{9.11 \times 10^{-14}}$	$\mathbf{2.24 \times 10^{-28}}$	$\mathbf{6.73 \times 10^{-28}}$	$\mathbf{3.70 \times 10^{-28}}$	$\mathbf{2.45 \times 10^{-28}}$	$\mathbf{2.24 \times 10^{-28}}$
	50	$\mathbf{2.67 \times 10^{-13}}$	$\mathbf{5.82 \times 10^{-28}}$	$\mathbf{6.98 \times 10^{-28}}$	$\mathbf{4.61 \times 10^{-28}}$	$\mathbf{6.86 \times 10^{-28}}$	$\mathbf{4.54 \times 10^{-28}}$
	60	$\mathbf{3.78 \times 10^{-11}}$	$\mathbf{9.12 \times 10^{-28}}$	$\mathbf{1.69 \times 10^{-27}}$	$\mathbf{1.01 \times 10^{-27}}$	$\mathbf{1.34 \times 10^{-27}}$	$\mathbf{1.10 \times 10^{-27}}$
Broyden banded	10	$\mathbf{4.18 \times 10^{-28}}$	$\mathbf{4.61 \times 10^{-30}}$	$\mathbf{4.81 \times 10^{-30}}$	$\mathbf{6.82 \times 10^{-29}}$	$\mathbf{7.43 \times 10^{-30}}$	$\mathbf{2.36 \times 10^{-30}}$
	20	$\mathbf{1.85 \times 10^{-26}}$	$\mathbf{2.63 \times 10^{-29}}$	$\mathbf{6.04 \times 10^{-29}}$	$\mathbf{7.47 \times 10^{-29}}$	$\mathbf{1.60 \times 10^{-28}}$	$\mathbf{4.77 \times 10^{-29}}$
	30	*12.2*	$\mathbf{2.25 \times 10^{-28}}$	$\mathbf{1.34 \times 10^{-28}}$	$\mathbf{2.08 \times 10^{-28}}$	$\mathbf{3.15 \times 10^{-28}}$	$\mathbf{2.89 \times 10^{-28}}$
	40	2.02×10^{-6}	$\mathbf{3.48 \times 10^{-28}}$	$\mathbf{7.41 \times 10^{-28}}$	$\mathbf{9.32 \times 10^{-28}}$	$\mathbf{1.34 \times 10^{-28}}$	$\mathbf{3.25 \times 10^{-28}}$
	50	9.33×10^{-5}	$\mathbf{6.08 \times 10^{-28}}$	$\mathbf{1.38 \times 10^{-27}}$	$\mathbf{6.78 \times 10^{-28}}$	$\mathbf{5.98 \times 10^{-28}}$	$\mathbf{1.04 \times 10^{-27}}$
	60	5.13×10^{-6}	$\mathbf{2.93 \times 10^{-27}}$	$\mathbf{3.44 \times 10^{-27}}$	$\mathbf{4.09 \times 10^{-27}}$	$\mathbf{7.98 \times 10^{-28}}$	$\mathbf{7.59 \times 10^{-28}}$
accurate		15/46	40/46	40/46	39/46	39/46	42/46

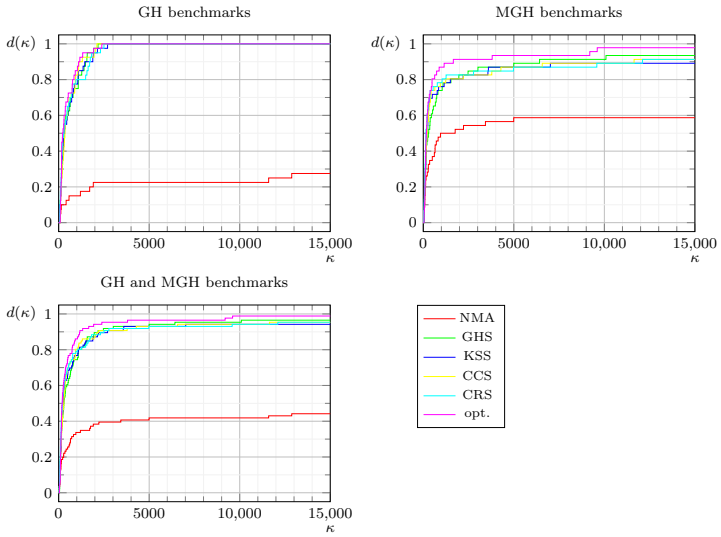

Figure 3. Data profiles for GH and MGH benchmark sets, separated and combined. No tolerance-based algorithm termination was applied ($Tol_f = Tol_X = 0$).

The advantage of the optimized schema becomes notable when observing the MGH benchmark set. At ~1600 simplex gradient estimates, the optimized schema solves more than 90% of the problems while the remaining adaptive schemas solve up to 83%, and the original NMA merely 50% of the problems. It is clearly the fastest schema and has the highest final percentage of solved problems (98%). Other adaptive schemas solve up to 93%, and the original NMA solves 59% of the problems.

The optimized schema remains the fastest when all GH and MGH benchmark problems are considered. Its advantage over the existing adaptive schemas is 6% at 1000, 5% at 2000, and 3% at 3000 simplex gradient estimates. The original NMA manages to keep up for less than 20% of the problems at ~150 simplex gradient estimates. The final percentage of solved problems is 99% for the optimized schema, up to 97% for the remaining adaptive schemas, and 44% for the original NMA.

In Figure 4, the same measurement of convergence speed is repeated with tolerance based algorithm termination set to $Tol_f = Tol_X = 10^{-4}$. The profiles are once more calculated for $\tau = 10^{-7}$. The maximum number of simplex gradient estimates κ_{max} = 25,000 does not play any role in this experiment because none of the individual Nelder–Mead runs ever uses the entire budget of available simplex gradient estimates. The algorithm is always terminated earlier by the tolerance-based criterion. The graphs are shown for up to 8000 simplex gradient estimates. No further progress is made beyond that point.

With tolerance-based termination enabled, the optimized schema performs even better than the other schemas. When we consider only GH benchmarks, the schemas again perform similarly, although, in general, the optimized schema is still slightly faster. The CCS is the first that solves 100% of GH problems in ~2100 estimates. Other adaptive schemas quickly follow. The only notable alteration can be observed for the original NMA which now performs worse and solves less than 20% of GH problems. The original NMA manages to converge in an additional ~10% of GH problems when it is allowed to run beyond the tolerance-based stopping criterion.

The advantage of the optimized schema becomes apparent in MGH data profiles. The KSS, CCS, and CRS keep pace for up to ~370 simplex gradient estimates where ~70% of the MGH problems are solved. The GHS starts to lag at ~240 estimates with ~40% of the problems solved. It catches up at 3000 estimates, finally solving 70% of the MGH problems. KSS, the best of the existing adaptive schemas, ends at 71%. The optimized schema is clearly better with 80% of the problems solved in 730 estimates, ending with 82% at 1660 estimates. The original NMA again performs worse compared to the run without tolerance-based termination, ending with 23% of solved problems.

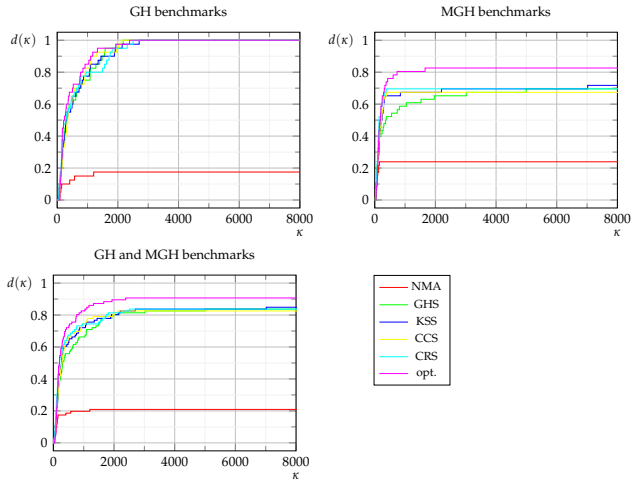

Figure 4. Data profiles for GH and MGH benchmark sets, separated and combined. Tolerance-based algorithm termination was applied ($Tol_f = Tol_X = 10^{-4}$).

For all-inclusive GH and MGH benchmark data profiles, the optimized schema starts to stand out from the rest of the existing adaptive schemas at ~60% of solved problems, achieved within ~300 simplex gradient estimates. The optimized schema reaches its final result, i.e., 90% of solved problems, in 2400 estimates. Its advantage over the best existing adaptive schema, i.e., KSS, is 6%. The KSS comes to 84% in 7020 estimates. As expected, the original NMA ends at a modest 21% of solved problems within 1200 estimates.

Figure 5 shows our last convergence speed measurement on GH, MGH, and CUTEr benchmark problems, 169 problems in total. The data profiles are shown for the CUTEr benchmark set, and GH, MGH, and CUTEr benchmark sets combined. They are calculated at convergence condition tolerance $\tau = 10^{-7}$ with κ_{max} = 25,000 simplex gradient estimates limit. Cases without the tolerance-based algorithm termination ($Tol_f = Tol_X = 0$), and with it ($Tol_f = Tol_X = 10^{-4}$), are shown. When the tolerance-based criterion is applied, the Nelder–Mead runs are always terminated before the limit of κ_{max} simplex gradient estimates is reached.

Data profiles in Figure 5 confirm our previous observations. Although not meta-optimized on CUTEr benchmark problems, the optimized schema solves the highest percentage of the problems in all shown cases at any given κ. It makes a difference of 4 to 6% compared to the first follower at ~400 simplex gradient estimates when tolerance-based algorithm termination is applied, and at ~1100 estimates when it is not. The optimized schema reaches or comes close to its final result in ~4000 estimates. It solves 91 to 97% of problems, GHS 84 to 93%, CRS 84 to 93%, KSS 86 to 91%, CCS 81 to 88%, and the original NMA manages 13 to 36%.

Besides the 40 GH and 46 MGH benchmarks, the following problems are included in data profiles in Figure 5: BrownAlmostLinear with dimensions $n_p = \{20, 30, 40, 50, 70, 100\}$ from MGH set, HilbertQuadratic with dimensions $n_p = \{10, 30, 60, 90\}$, OrenPower [34] with dimensions $n_p = \{30, 50, 60, 70, 80, 90, 100\}$, and ARWHEAD_100, DQDRTIC_50, DQDRTIC_100, SPARSINE_50, SPARSINE_100, CHNROSNB_25, CHNROSNB_50, SCOSINE_10, LIARWHD_100, FLETCHBV_100, DIXON3DQ_100, OSCIGRAD_25, OSCIGRAD_100, NONCVXUN_10, NONCVXUN_100, PENALTY1_50, PENALTY1_100, SINQUAD_50, SINQUAD_100, FLETCBV3_100, PENALTY2_100, TOINTGSS_50, TOINTGSS_100, ARGLINC_50, EXTROSNB_100, COSINE_100, TRIDIA_50, TRIDIA_100, NONDQUAR_100, QUARTC_25, QUARTC_100, FREUROTH_100, WATSON_31, ERRINROS_25, ERRINROS_50, NONDIA_20, NONDIA_30, NONDIA_50, NONDIA_90, NONDIA_100, MANCINO_20, MANCINO_30, DQRTIC_100, ENGVAL1_50, ENGVAL1_100, HILBERTA_10, FLETCBV2_100, TQUARTIC_10, EDENSCH_36, ARGLINA_50, ARGLINA_100, BOX_100, POWELLSG_36, POWELLSG_40, POWELLSG_60, POWELLSG_80, POWELLSG_100, POWER_75, POWER_100, HILBERTB_50, ARGLINB_50,

MOREBV_50, BDQRTIC_100, SCURLY10_100, VAREIGVL_49, VAREIGVL_99 from CUTEr benchmark problem set.

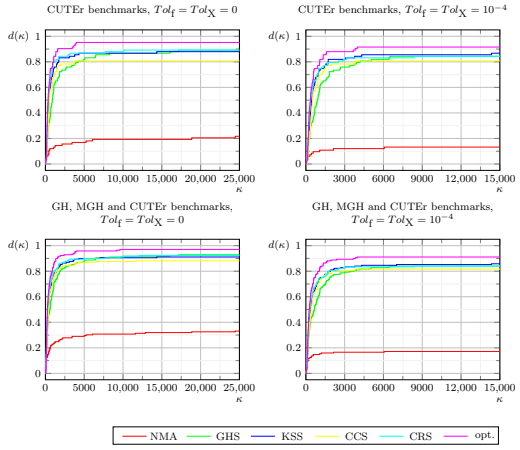

Figure 5. Data profiles for Constrained and Unconstrained Testing Environment, revisited (CUTEr) benchmark set and GH, MGH, and CUTEr benchmark sets combined, without ($Tol_f = Tol_X = 0$) and with tolerance-based algorithm termination applied ($Tol_f = Tol_X = 10^{-4}$).

Speed of convergence of a particular adaptive parameter schema is mirrored in the simplex's best value descent during an individual Nelder–Mead run. The descent rate can be expressed by $\cos\theta$, where θ is an angle between the search direction $\mathbf{d}$ and the gradient of the objective function $\nabla f(\mathbf{x})$:

$$\cos\theta = \frac{\mathbf{d}^{\mathrm{T}} \nabla f(\mathbf{x})}{|\mathbf{d}|\ |\nabla f(\mathbf{x})|} \quad . \tag{11}$$

Search direction $\mathbf{d}$ is locally descending when $\cos\theta < 0$. The fastest descent is achieved at $\cos\theta = -1$. According to the NMA definition, the search direction is $\mathbf{d} = c(\mathbf{P}_{n_p} - \overline{\mathbf{P}})$, where c is the reflection (α), expansion (β), or contraction (γ) NMA parameter. The descent rate in a non-shrinking NMA iteration is therefore calculated as

$$\cos\theta = \frac{(\mathbf{P}_{n_p} - \overline{\mathbf{P}})^{\mathrm{T}} \nabla f(\mathbf{P}_{n_p})}{|\mathbf{P}_{n_p} - \overline{\mathbf{P}}|\ |\nabla f(\mathbf{P}_{n_p})|} \quad . \tag{12}$$

The simplex's best value descents and corresponding descent rates for $n_p = 100$-dimensional GH benchmark problems are shown in Figure 6. The figure shows that all the existing schemas as well as the optimized adaptive schema manage to maintain some level of descent during the entire Nelder–Mead run. A higher descent rate, i.e., more negative $\cos\theta$, ensures faster objective function descent and fulfillment of the termination criteria. The optimized schema is the fastest or near fastest in all shown cases except for the $\epsilon = 0.0$, $\sigma = 0.0001$ case. This is partly reflected in the poorer descent rate of the optimized schema for that particular case.

The tolerance boundary intersections shown in Figure 6 are t_{ps} values from (6). Data profiles in Figures 3–5 summarize tolerance boundary intersections over the entire benchmark problem set.

The absence of a sufficient descent rate is fatal for the original NMA. The original NMA manages some slow descent only in the $\epsilon = 0.0$, $\sigma = 0.0$ case. In all remaining cases, $\cos\theta$ approaches 0°. Search direction $\mathbf{d}$ becomes orthogonal to the negative gradient which was first observed by Torczon [12]. As a consequence, the original NMA stops descending and does not achieve the convergence boundary.

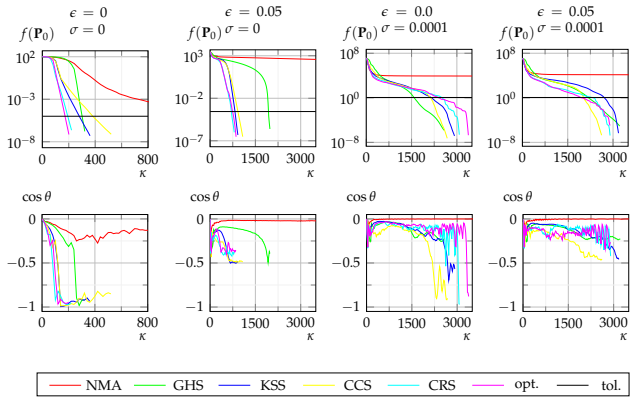

Figure 6. Best objective function values ($f(\mathbf{P}_0)$) and corresponding descent rates ($\cos\theta$) during Nelder–Mead run for $n_p = 100$-dimensional GH benchmarks. Tolerance-based algorithm termination applied ($Tol_f = Tol_X = 10^{-4}$). Black line represents convergence boundary (7) for $\tau = 10^{-7}$.

In [4], the authors prove that NMA does not perform shrink iteration when the objective function is strictly convex. Furthermore, for a uniformly convex objective function, the descent rate is provided by expansion and contraction iterations [15], although the effect diminishes with problem dimension n_p. In other words, to maintain a sufficient descent rate, an adequate share of expansion and contraction iterations is needed. Note that a uniformly convex function is also strictly convex. Since the modified quadratic function (10) is uniformly convex, the above applies to the GH benchmark set. The share of non-reflection iterations, i.e., expansion and contraction iterations combined, is shown in Figure 7. In general, it declines with the problem dimension for all schemas and (ϵ,σ) pairs. Nevertheless, all the adaptive parameter schemas manage to keep the share above 5%, which provides a sufficient descent rate. The CCS stands out with its lowest non-reflection share above 35%, yet, such a high share is not reflected in better performance. The non-reflection share alone, therefore, does not guarantee high convergence speed. The lowest non-reflection share of the optimized schema is 12% at $n_p = 100$ in the $\epsilon = 0.0$, $\sigma = 0.0001$ case. The original NMA's share is, on the other hand, never greater than 26% (which in turn is achieved for lower-dimensional problems). With problem dimension increase, it quickly drops as low as to 0.56% in the worst case, which confirms the poor performance and convergence problems of the original NMA schema (1).

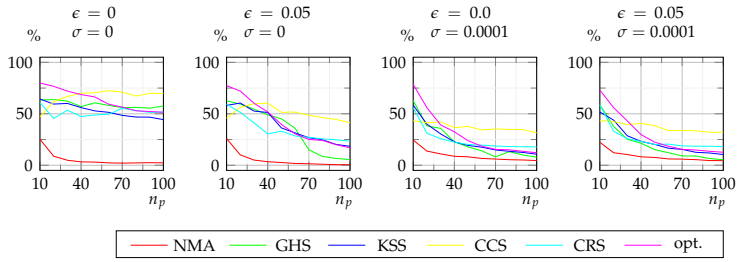

Figure 7. Share of expansion and contraction iterations for GH benchmark problems.

5. Conclusions

Adaptive parameter schemas address poor NMA performance on high-dimensional problems. We used a meta-optimization procedure to find a novel adaptive parameter schema presented in this paper. Although the meta-optimization problem seems simple, brute force optimization is not feasible due to the immense computing power required. To set up the problem, we defined a mathematical formulation of the adaptive parameter schema and an objective function evaluating a schema's performance. We tuned the meta-optimization parameters in a series of shorter meta-optimization runs. The final settings constrain the meta-optimization parameter space, define a single NMA run termination

criteria to evaluate an SC's performance, limit the number of NMA iterations to catch non-convergent SCs, and limit the number of meta-optimization iterations. We used PSADE, a robust global parallel asynchronous method.

The performance of the proposed adaptive parameter schema is discussed and compared with the existing schemas. The share of non-reflection iterations and the descent rate do not show any significant deviation of the proposed schema from the existing ones. However, data profiles on GH modified quadratic, MGH, and CUTEr benchmark problems show that the proposed schema outperforms the existing ones in both accuracy and convergence speed. We performed the evaluation with and without tolerance-based termination of the NMA.

The proposed schema is a result of a meta-optimization procedure. We evaluate its performance but, on the other hand, provide no mathematical explanation for why the schema performs so well. The proposed schema is the global minimum determined by the schema's mathematical formulation and meta-optimization objective function definition.

Author Contributions: Conceptualization, Á.B. and J.P.; methodology, Ž.R. and J.P.; software, J.O. and J.P.; validation, Á.B.; formal analysis, J.P.; investigation, Ž.R., J.O., and J.P.; data curation, J.P.; writing—original draft preparation, J.P.; writing—review and editing, Á.B. and J.P.; supervision, Á.B. and T.T.; project administration, T.T.; funding acquisition, T.T. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge the financial support from the Slovenian Research Agency (research core funding No. P2-0246 ICT4QoL—Information and Communications Technologies for Quality of Life).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NMA	Nelder–Mead Algorithm
PSADE	Parallel Simulated Annealing with Differential Evolution
SC	Schema Candidate
GH	Gao–Han
MGH	Moré–Garbow–Hilstrom
CUTEr	Constrained and Unconstrained Testing Environment, revisited
GHS	Gao–Han Schema
KSS	Kumar–Suri Schema
CCS	Chebyshev Crude Schema
CRS	Chebyshev Refined Schema
PyPI	Python Package Index
f	objective function
n_p	number of optimized variables or problem dimension
$\mathbf{P}_i$, $\overline{\mathbf{P}}$	ith simplex vertex and centroid of simplex vertices
$\mathbf{P}_r$, $\mathbf{P}_e$	reflected point and expanded point
$\mathbf{P}_{rc}$, $\mathbf{P}_{nc}$	reflected point and worst point contracted towards centroid
α, β, γ, δ	NMA reflection, expansion, contraction, and shrink parameters
n_c	CRS constant calculated from n_p
$\mathbf{x}_0$, $\mathbf{x}$	starting point, an arbitrary point in n_p-dimensional parameter space
ϵ_i, $\mathbf{e}_i$	ith component Pfeffer's constant and unit vector
Tol_f, Tol_X	simplex flatness and size tolerances

P_{ij}	jth component of ith simplex vertex
$c_{0\alpha}$, $c_{1\alpha}$, $c_{0\beta}$, $c_{1\beta}$, etc.	meta-optimization variables defining an SC
p, $\mathcal{P}$	single benchmark problem and set of benchmark problems
s, $\mathcal{S}$	single parameter schema and set of all parameter schemas
r, $\mathcal{R}$	reference parameter schema and set of all reference parameter schemas
$\mathcal{X}$	set of sets of GH and MGH benchmark problems
κ	number of simplex gradient estimates
$\kappa_{\max}$	κ available for schema evaluation per single p
t_{ps}	number of objective function evaluations needed on problem p by schema s to satisfy (7)
$d_{s\mathcal{P}}(\kappa)$	share of problems from set $\mathcal{P}$ solved by schema s in κ simplex gradient estimates
f_{L}	lowest objective function value reached in $\kappa_{\max}$ simplex gradient estimates by any of the schemas $s \in \mathcal{S}$ on a particular problem
τ	convergence condition tolerance
$w_{\mathcal{P}_+}$	weight used in meta-optimization objective function when at least one of the reference schemas outperforms the evaluated SC on set of benchmark problems $\mathcal{P}$
$w_{\mathcal{P}_-}$	weight used in meta-optimization objective function when the evaluated SC outperforms all the reference schemas on set of benchmark problems $\mathcal{P}$

References

1. Nelder, J.A.; Mead R. A simplex method for function minimization. *Comput. J.* **1965**, *7*, 308–313. [CrossRef]
2. McKinnon, K.I.M. Convergence of the Nelder-Mead simplex method to a non-stationary point. *J. Optim.* **1998**, *9*, 148–158.
3. Galántai, A. A convergence analysis of the Nelder-Mead simplex method. *Acta Polytech. Hungarica* **2021**, *18*, 93–105. [CrossRef]
4. Lagarias, J.C.; Reeds, J.A.; Wright, M.H.; Wright, P.E. Convergence properties of the Nelder-Mead simplex method in low dimensions. *J. Optim.* **1998**, *9*, 112–147. [CrossRef]
5. Lagarias, J.C.; Poonen, B.; Wright, M.H. Convergence of the restricted Nelder-Mead algorithm in two dimensions. *J. Optim.* **2012**, *22*, 501–532. [CrossRef]
6. Kelley, C.T. Detection and remediation of stagnation in the Nelder-Mead algorithm using a sufficient decrease condition. *J. Optim.* **1999**, *10*, 43–55. [CrossRef]
7. Tseng, P. Fortified-descent simplicial search method: A general approach. *J. Optim.* **1999**, *10*, 269–288. [CrossRef]
8. Nazareth, L.; Tseng, P. Gilding the lily: A variant of the Nelder-Mead algorithm based on golden-section search. *Comput. Optim. Appl.* **2002**, *22*, 133–144. [CrossRef]
9. Price, C.J.; Coope, I.D.; Byatt, D. A convergent variant of the Nelder-Mead algorithm. *J. Optim. Theory. Appl.* **2002**, *113*, 5–19. [CrossRef]
10. Bűrmen, Á.; Puhan, J.; Tuma, T. Grid restrained Nelder-Mead algorithm. *Comput. Optim. Appl.* **2006**, *34*, 359–375. [CrossRef]
11. Bűrmen, Á.; Tuma, T. Unconstrained derivative-free optimization by successive approximation. *Comput. Appl. Math.* **2009**, *223*, 62–74. [CrossRef]
12. Torczon, V.J.; Multi-Directional Search: A Direct Search Algorithm for Parallel Machines. Ph.D. Thesis, Rice University, Houston, TX, USA, 1989.
13. Wright, M.; Direct search methods: Once scorned, now respectable. In Proceedings of the 16th Dundee Biennial Conference in Numerical Analysis, Dundee, Scotland, 27–30 June 1996; pp. 191–208.
14. Han, L.; Neumann M. Effect of dimensionality on the Nelder-Mead simplex method. *Optim. Methods Softw.* **2006**, *21*, 1–16. [CrossRef]
15. Gao, F.; Han, L. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. *Comput. Optim. Appl.* **2012**, *51*, 259–277. [CrossRef]
16. Fajfar, I.; Puhan, J.; Bűrmen, Á. Evolving a Nelder-Mead algorithm for optimization with genetic programming. *Evol. Comput.* **2017**, *25*, 351–373. [CrossRef] [PubMed]
17. Musafer, H. A.; Mahmood, A. Dynamic Hassan–Nelder-Mead with simplex free selectivity for unconstrained optimization. *IEEE Access* **2018**, *6*, 39015–39026. [CrossRef]
18. Fajfar, I.; Bűrmen, Á.; Puhan, J. The Nelder-Mead simplex algorithm with perturbed centroid for high-dimensional function optimization. *Optim. Lett.* **2019**, *13*, 1011–1025. [CrossRef]
19. Kumar, G.N.S.; Suri, V.K. Multilevel Nelder-Mead's simplex method. In Proceedings of the 2014 9th International Conference on Industrial and Information Systems (ICIIS), Gwalior, India, 15–17 December 2014; Volume 9; pp. 1–6.
20. Mehta, V.K. Improved Nelder-Mead algorithm in high dimensions with adaptive parameters based on Chebyshev spacing points. *Eng. Optim.* **2020**, *52*, 1814–1828. [CrossRef]
21. Olenšek, J.;Tuma, T.; Puhan, J.; Bűrmen, Á. A new asynchronous parallel global optimization method based on simulated annealing and differential evolution. *Appl. Soft Comput.* **2011**, *11*, 1481–1489. [CrossRef]

22. Moré, J.J.; Garbow, B.S.; Hilstrom, K.E. Testing unconstrained optimization software. *ACM Trans. Math. Softw.* **1981**, *7*, 17–41. [CrossRef]
23. Gould, N.I.M.; Orban, D.; Toint, P.L. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Softw.* **2003**, *29*, 373–394. [CrossRef]
24. Moré, J.J.; Wild, S.M. Benchmarking derivative-free optimization algorithms. *SIAM J. Optim.* **2009**, *20*, 172—191. [CrossRef]
25. Olenšek, J.; Bűrmen, Á.; Puhan, J.; Tuma, T. DESA: A new hybrid global optimization method and its application to analog integrated circuit sizing. *J. Glob. Optim.* **2009**, *44*, 53–77. [CrossRef]
26. Bűrmen, Á. PyOPUS-Simulation, Optimization, and Design. Available online: http://fides.fe.uni-lj.si/pyopus (accessed on 15 May 2022).
27. Bűrmen, B.; Locatelli, I.; Bűrmen, Á.; Bogataj, M.; Mrhar, A. Mathematical modeling of individual gastric emptying of pellets in the fed state. *J. Drug. Deliv. Sci. Technol* **2014**, *24*, 418–424. [CrossRef]
28. Rojec, Ž.; Olenšek, J.; Fajfar, I. Analog circuit topology representation for automated synthesis and optimization. *Inf. MIDEM* **2018**, *48*, 29–40.
29. Rojec, Ž.; Bűrmen, Á.; Fajfar, I. Analog circuit topology synthesis by means of evolutionary computation. *Eng. Appl. Artif. Intell.* **2019**, *80*, 48–65. [CrossRef]
30. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. CCSA: Conscious Neighborhood-based Crow Search Algorithm for Solving Global Optimization Problems. *Appl. Soft Comput.* **2019**, *85*, 105583. [CrossRef]
31. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. QANA: Quantum-based avian navigation optimizer algorithm. *Eng. Appl. Artif. Intell.* **2021**, *104*, 104314. [CrossRef]
32. Nadimi-Shahraki, M.H.; Zamani, H. DMDE: Diversity-maintained multi-trial vector differential evolution algorithm for non-decomposition large-scale global optimization. *Expert Syst. Appl.* **2022**, *98*, 116895. [CrossRef]
33. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. Starling murmuration optimizer: A novel bio-inspired algorithm for global and engineering optimization. *Comput. Methods Appl. Mech. Engrg.* **2022**, *392*, 114616. [CrossRef]
34. Shanno, D.F.; Phua, K. Matrix conditioning and nonlinear optimization. *Math. Program.* **1978**, *14*, 149–160. [CrossRef]