Zorica Dražić

A SURVEY ON CONTINUOUS VARIABLE NEIGHBORHOOD SEARCH BASED HEURISTICS

Abstract. Variable neighborhood search is a metaheuristic designed for solving hard optimization problems. This paper gives an overview of methods for solving continuous optimization problems, or more precisely, Continuous nonlinear programming problems.

Mathematics Subject Classification (2020): 90C26, 90C30, 90C59; 90-02

Keywords: global optimization, nonlinear programming, variable neighborhood search, metaheuristics

Faculty of Mathematics, University of Belgrade, Belgrade, Serbia
zorica.drazic@matf.bg.ac.rs
ORCID: 0000-0002-3434-6734

https://doi.org/10.64191/zr2510310106

Contents

1. Introduction 216
2. Metric C-VNS (with neighborhoods induced by metric) 218
2.1. Min-max NLP-polyphase radar code design 219
2.2. GLOB-VNS 220
2.3. Box-constrained NLP-molecular potential energy 222
2.4. General C-VNS 223
2.5. The influence of a neighborhood shape 223
2.6. VNS with Modified Nelder Mead as local search 224
2.7. VNS for NSEs 226
2.8. Some other metric C-VNS implementations 227
3. Non-metric C-VNS 228
3.1. Gauss-VNS 228
3.2. C-VNS with fat tailed distributions 229
4. Constrained NLP by C-VNS 232
5. Conclusion 234
References 234

1. Introduction

The unconstrained Nonlinear programming (NLP) problem can be formulated as follows:

$$\min\{f(x)|x \in S,\ S \subseteq \mathbb{R}^n\}, \tag{1.1}$$

where $f\colon \mathbb{R}^n \to \mathbb{R}$ is a continuous objective function, S is a feasible set, and x is a feasible solution. A point x^* is a global minimum of f if $f(x^*) \leqslant f(x)$ for all $x \in \mathbb{R}^n$. If there exists $\varepsilon > 0$ such that $f(x^*) \leqslant f(x)$ for all x such that $||x-x^*|| \leqslant \varepsilon$, then x^* is a local minimum. The NLP consists of finding a global minimum x^* (or several of them), which is also called the optimal solution.

In general, there are no further assumptions for the objective function f: It does not have to be convex or concave, it may be non-differentiable and there may exist many local minima.

The literature on NLP can be divided into two parts: Deterministic and exact approaches on the one hand and (meta)heuristics on the other. An exact algorithm for the problem (1.1), if one exists, finds an optimal solution x^*, together with the proof of its optimality, or shows that there is no feasible solution i.e., $X = \emptyset$. Moreover, the time required for this should be finite and not too long in practice.

For nonconvex functions, if the dimension of the problem is sufficiently large, the problem (1.1) becomes almost impossible to solve precisely in a reasonable time frame. This explains why so many general (meta)heuristic approaches have been proposed so far.

Unconstrained NLP problems arise in many practical applications, such as financial planning, risk management, data analysis, scientific modelling, facility location, etc. Advanced engineering design problems such as the spread spectrum radar polyphase code design can be modelled as a min-max nonlinear nonconvex optimization problem [44]. In chemistry, NLPs are used in the determination of the three-dimensional structure of a molecule [21, 37], in molecular modeling [40], in phase equilibrium calculations [55], protein structure prediction [34], etc. In econometrics, NLP models are developed to address the complexity of the real phenomena to be analysed [9, 17, 29]. In biology, the models are usually dynamic in nature and highly nonlinear, and therefore rely on optimization methods [6].

In most cases, such real-life problems are very difficult because the objective function contains a large number of local minima, which tends to increase exponentially with the dimension of the problem. In such cases, it seems impossible to design methods that provide a guarantee for finding the global minimum in a reasonable time frame. Therefore, many heuristic methods have been developed for solving (1.1): Genetic Algorithm [14, 13], the Reactive Tabu Search with stochastic minimizer [7], integration of Nelder–Mead simplex search method with Genetic Algorithm and Particle Swarm Optimization [24], Continuous scatter search [32], Ant Colony Optimization approach [56], Tabu Search [2], Simulated Annealing [31], etc.

The main problem in the implementation of (meta)heuristic methods is the estimation of their parameters. Variable Neighborhood Search (VNS), on the other hand, depends basically on one parameter that determines the number of different neighborhoods, which allows an efficient implementation.

Contrary to discrete optimization, in continuous optimization the solution space and the neighborhoods $\mathcal{N}_k$ used in the VNS are infinite sets. Therefore, it is not possible to fully explore any neighborhood of a point in a local search step, which is usually done in the discrete case. Some of the local minimizers that can be used in a local search step of VNS for continuous optimization are Nelder-Mead [46], Hooke-Jeeves [33], Rosenbrock [52], Steepest Descent, Fletcher-Powell [27], Fletcher-Reeves [26], etc. The choice of method depends on the characteristics of the objective function: Some of them do not require gradients and can be used for non-smooth functions, while others use the information about the gradient, which in this case has to be provided by the user in some way, or automatically calculated numerically.

Several methods based on VNS have been proposed in the literature for solving continuous (un)constrained global optimization problems. The first continuous optimization problem solved with VNS was the multi-source Weber problem (the continuous version of the location-allocation problem) [11, 10]. Although this problem belongs to continuous optimization, the main steps to obtain a solution were actually combinatorial. The second continuous problem with combinatorial structure solved by VNS was the general bilinear problem [30] and its special case, the

pooling problem [4]. Although these two problems belong to continuous optimization, the main steps to solve them were actually combinatorial in nature. In the multi-source Weber problem, for example, the neighbourhood of a point (from $\mathbb{R}^n$) can be changed either by moving objects or by reallocating consumers. In this way, the total number of solutions is always finite.

VNS for solving continuous global optimization problems (C-VNS) has been used in the literature in two different ways: With or without neighborhoods induced by an ℓ_p norm. The first method seems to be a natural approach for applying VNS to solve global continuous optimization problems and is therefore the most common in the literature. On the other hand, the second approach showed certain advantages over the use of classical neighborhoods.

This paper is organized as follows. In the next section a metric C-VNS based heuristics are presented, i.e. different VNS implementations with neighborhoods induced by metric. A non-metric C-VNS realizations are presented in Section 3. Applying VNS for constrained NLP is discussed in section 4. Brief conclusions are drawn in Section 5.

2. Metric C-VNS (with neighborhoods induced by metric)

The metric function $\rho(x, y)$ which specifies the distance between two solutions $x, y \in S$ from the solution space S can be defined as ℓ_p distance as follows:

$$\rho(x,y) = ||x-y||_p = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}, \quad 1 \leqslant p < \infty, \tag{2.1}$$

$$\rho(x,y) = ||x-y||_\infty = \max_{1 \leqslant i \leqslant n} |x_i - y_i|, \quad p = \infty. \tag{2.2}$$

These metrics lead to different geometry structures $\mathcal{G}$ of the neighborhoods to be explored. Common choices are ℓ_1, ℓ_2, and ℓ_∞ metric. The neighborhood structures $\mathcal{N}_k(x)$ are defined as a set of solutions in the k-th neighborhood of a solution x and can be written using the metric ℓ_p in the following way:

$$\mathcal{N}_k(x) = \{y \in S | \rho(x,y) \leqslant r_k\}, \tag{2.3}$$

$$\mathcal{N}_k(x) = \{y \in S | r_{k-1} < \rho(x,y) \leqslant r_k\}, \tag{2.4}$$

where S is a solution space and r_k is the radius of a neighborhood $\mathcal{N}_k$ monotonically increasing with k. Thus, the geometry of the neighborhood structures $\mathcal{G}$ is determined by the choice of the metric $\rho(x, y)$, and the neighborhoods $\mathcal{N}_k$ are determined by $\mathcal{G}$ and r_k.

When designing a C-VNS, the following parameters should be defined:

- A total number $k_{\max}$ of neighborhoods $\mathcal{N}_k$.
- A set of neighborhood structures $\mathcal{N}_k$ which are defined by the pair (ρ, r_k), $k = 1, \ldots, k_{max}$.
- A distribution $\mathcal{P}_k$ or a way to generate a random point from $\mathcal{N}_k$ in the shaking step.
- Local optimizer which will be used in the local search step.

The pseudo-code for C-VNS is given in Figure 1.

```
        Algorithm C-VNS
        /* Initialization */
01      Select the set of neighborhood structures N_k, k = 1,...,k_max
        with the corresponding random distributions.
02      Choose an arbitrary initial point x ∈ S
03      Set x* ← x, F* ← F(x)
        /* Main loop */
04      repeat the following steps until the stopping condition is met
05          Set k ← 1
06          repeat the following steps until k > k_max
07              Shake: Generate at random a point y ∈ N_k(x*)
08              Apply some local search method from y to obtain a local
                minimum y'
09              if F(y') < F* then
10                  Set x* ← y', F* ← F(y') and goto 05
11              endif
12              Set k ← k + 1
13          end
14      end
15      Stop. Point x* is an approximate solution of the problem.
```

FIGURE 1. Algorithm C-VNS

2.1. Min-max NLP-polyphase radar code design. The VNS algorithm for solving a "pure" continuous optimization problem was used for the first time in [44]. The authors considered the polyphase radar code design which is a continuous min-max nonlinear nonconvex optimization problem with piecewise smooth objective function. This problem can be expressed as follows:

$$\texttt{global}\min_{x\in X} f(x) = \max\{\varphi_1(x),\dots,\varphi_{2m}(x)\}, \tag{2.5}$$

$$X = \{(x_1,\dots,x_n)\in\mathbb{R}^n | 0 \leqslant x_j \leqslant 2\pi, j = 1,\dots,n\}$$

where $m = 2n-1$, and

$$\varphi_{2i-1}(x) = \sum_{j=i}^{n}\cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k\right), \quad i = 1,\dots,n,$$

$$\varphi_{2i}(x) = 0.5 + \sum_{j=i+1}^{n}\cos\left(\sum_{k=|2i-j|+1}^{j} x_k\right), \quad i = 1,\dots,n-1,$$

$$\varphi_{m+i}(x) = -\varphi_i(x), \quad i = 1,\dots,m.$$

This problem has exponentially growing number of local minima and in [35] it is proved to be NP-hard. The objective function of this problem for the case $n = 2$ is shown in Figure 2.

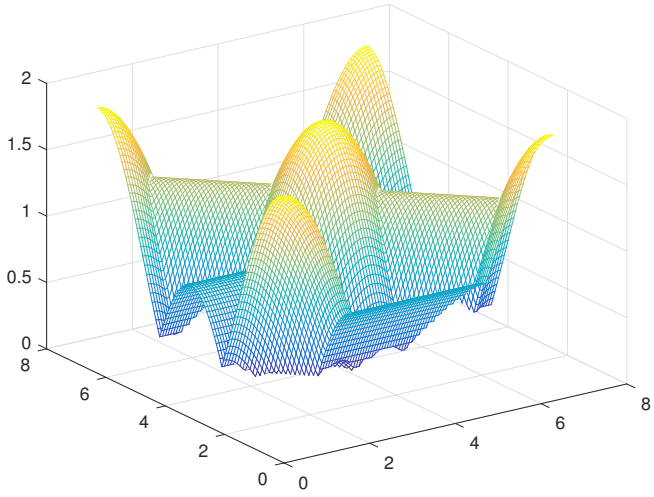


FIGURE 2. Objective function of polyphase radar code design for $n = 2$.

To solve the problem (2.5), the proposed VNS algorithm and its two variants of local search were quite general, so that the method could be applied to any continuous min-max optimization problem. The neighborhood structure is defined by choosing a set of directions around the current solution and a parameter, corresponding to the distance traveled along a direction. The first local search procedure was performed as in [35], while the second one selects the steepest descent directions for the currently active functions. The neighborhoods $\mathcal{N}_k(x), k = 1, \ldots, k_{\max}$ used in both versions are hypercubes around the incumbent solution x of different sizes.

2.2. GLOB-VNS. In [42, 36, 20] the authors consider a special class of continuous global NLP optimization problems with box constraints

$$\min f(x_1, \ldots, x_n), \quad a_i \leqslant x_i \leqslant b_i, \qquad i = 1, \ldots, n, \tag{2.6}$$

and they implemented various VNS-based heuristics to solve these problems within the software package GLOB. Any unconstrained NLP problem can be considered as a box-constrained problem if the left and right values of the variables defining the box are set to some large negative and positive values a_i and b_i. The code of GLOB is written in ANSI C programming language and besides VNS, it implements Multistart Local Search and Random Search (Monte Carlo Method). The package offers various statistical reporting options that are suitable for research purposes.

In GLOB-VNS, the authors have implemented three basic VNS-based heuristics whose pseudo-code is shown in Figure 3.

The different choice of geometric neighborhood shapes and random point distributions led to different VNS heuristics. The random point in the shaking step is generated using the following two steps. In the first step, a random direction d is generated using some distribution defined by the heuristic. In the second step, a random radius r is determined from the interval $[0, r_k]$ for neighborhoods of type

(2.3) or $[r_{k-1}, r_k]$ for neighborhoods of type (2.4). Finally, the direction d is scaled to r in a metric that defines the geometry of $\mathcal{N}_k$.

In the first VNS heuristic, the neighborhood structures $\mathcal{N}_k,\ k = 1, \ldots, k_{\max}$ were defined as (2.3), i.e. the ℓ_∞ balls with uniform distribution used in the shaking step. The second heuristic uses ℓ_1 balls with a uniform distribution, and the third heuristic uses ℓ_1 balls with a specially designed distribution. This specially designed distribution (named hypergeometric random point distribution) for the third heuristic generates a random point $x = (x_1, \ldots, x_n)$ on ℓ_1 unit sphere in the following way:

(i) x_1 is taken uniformly from the interval $[-1, 1]$, x_k is taken uniformly from the interval $[-A_k, A_k]$ where $A_k = 1 - |x_1| - ... - |x_{k-1}|, k = 2, \ldots, n-1$, and x_n takes A_n with random sign.
(ii) coordinates of x are randomly permuted.

	Algorithm Glob-VNS
	/* *Initialization* */
01	Select the pairs $(\mathcal{G}_l, \mathcal{P}_l),\ l = 1, \ldots, m$ of geometry structures and distribution types and a set of radii $\rho_i,\ i = 1, \ldots, k_{\max}$
02	Choose an arbitrary initial point $x \in S$
03	Set $x^* \leftarrow x$, $f^* \leftarrow f(x)$
	/* *Main loop* */
04	**repeat** the following steps **until** the stopping condition is met
05	Set $l \leftarrow 1$
06	**repeat** the following steps **until** $l > m$
07	Form the neighborhoods $\mathcal{N}_k$, $k = 1, \ldots, k_{\max}$ using geometry structure $\mathcal{G}_l$ and radii ρ_k
08	Set $k \leftarrow 1$
09	**repeat** the following steps **until** $k > k_{\max}$
10	*Shake:* Generate at random a point $y \in \mathcal{N}_k(x^*)$ using random distribution $\mathcal{P}_l$
11	Apply some *local search* method from y to obtain a local minimum y'
12	**if** $f(y') < f^*$ **then**
13	Set $x^* \leftarrow y'$, $f^* \leftarrow f(y')$ and **goto** line 05
14	**endif**
15	Set $k \leftarrow k + 1$
16	**end**
17	Set $l \leftarrow l + 1$
18	**end**
19	**end**
20	**Stop**. Point x^* is an approximate solution of the problem.

FIGURE 3. Algorithm Glob-VNS

GLOB-VNS also implements hybrid techniques which are combinations of any two or all three implemented heuristics. The idea of using several neighborhood structures and distributions in the shaking step turned out to be more efficient compared to variants with fixed geometry and distribution.

For all three heuristics, the radii $r_k,\ k = 1, .., k_{\max}$ are monotonically increasing and can be defined by the user or calculated automatically in the minimization process. The user can choose one of six different non-linear programming methods for the local search phase. Three of them are the first-order methods: Steepest Descent, Fletcher-Reeves, and Flecher-Powell, and the other three are: Nelder-Mead, Hooke-Jeeves, and Rosenbrock with no gradient information.

2.3. Box-constrained NLP-molecular potential energy. Using the advanced version of GLOB-VNS in [21], the authors have proposed a VNS heuristic for minimizing the potential energy (MPE) function of a molecule. The determination of the three-dimensional structure of a molecule can be formulated as a continuous optimization NLP with box-constraints. This structure corresponds to the one involving the global minimum of the MPE function for which the number of local minima grows exponentially with molecule size. The problem has a form:

$$\texttt{global}\min_{x\in X} f(x),$$

$$X = \{(x_1, \ldots, x_n) | 0 \leqslant x_i \leqslant 5, i = 1, \ldots, n\},$$

$$f(x) = \sum_{i=1}^{n} \Big(1 + \cos(3x_i) + \frac{(-1)^i}{\sqrt{10.60099896 - 4.141720682\cos(x_i)}}\Big).$$

The objective function of this problem for the case $n = 2$ is illustrated in Figure 4.

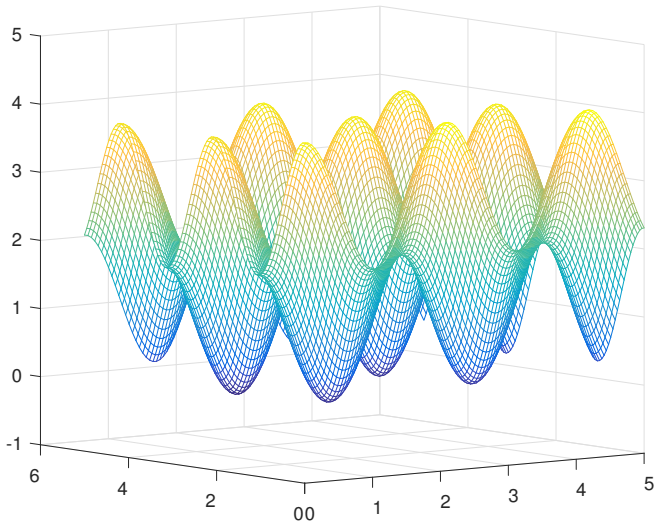


FIGURE 4. Objective function of MPE for n=2.

For this problem, the authors observed that using the uniform distribution to generate the random point in the shaking step was not the best choice. The specially designed hypergeometric distribution on a unit ℓ_1 sphere allowed them to generate more initial points for descents closer to the axial direction and achieve much better results in terms of computational effort.

2.4. General C-VNS. In [43] a general C-VNS (CGVNS) scheme was proposed to solve (2.6). Previous approaches used neighborhoods induced from a single metric function, and VNS was iterated until some stopping condition was met. In CGVNS, the neighborhoods induced from the different metric functions are considered simultaneously within one heuristic. In that way the same "ingredients" were used as in GLOB-VNS, but within one, general VNS scheme.

Neighborhood structures were determined by the pairs $(\ell_1, r_k), (\ell_\infty, r_k), k = 1, \ldots, k_{\max}$ according to (2.3) or (2.4) where r_k are predefined radii such that $r_1 < \ldots < r_{k_{\max}}$.

In addition to the three distributions used within GLOB-VNS, the fourth distribution for generating a random point in the shaking step was added. Using this distribution, a random point is obtained by generating a random direction d using a uniform distribution in ℓ_∞ unit ball. With the current point x and the box constraint vectors a and b, the coordinates of a vector d are scaled in the following way. If the coordinate $d_i \geqslant 0$, the scaling factor is $(b_i - x_i)/r_{k_{\max}}$ and if $d_i < 0$ the scaling factor is $(x_i - a_i)/r_{k_{\max}}$, where $r_{k_{\max}}$ is a predefined maximal radii. In this way, the point obtained is always within the bounds, $\mathcal{N}_{k_{\max}}$ is equal to the entire box region and $\mathcal{N}_k$ is the box region that is contracted towards the point x with the contraction factor $r_k/r_{k_{\max}}$.

With CGVNS, the idea of changing neighborhoods within the search has been fully explored: Neighborhood structures in $\mathbb{R}^n$ are induced from the different metric functions, several points from the same neighborhood are randomly selected according to four different distributions, and several local minimizers are used. This approach turned out to be noticeably more efficient than the variants with fixed geometry and fixed distributions.

2.5. The influence of a neighborhood shape. The efficiency of VNS for continuous global optimization problems strongly depends on the geometric shape of the neighborhood structures used by the algorithm. Since the shaking step represents the diversification part of the VNS, it is crucial for an algorithm to be able to escape from a local minimum to some better solution in the solution space. The influence of the neighborhood shape in the shaking step on the efficiency of a VNS is investigated in [18].

The most simple for software implementation and therefore the most commonly used in the literature are ℓ_∞ balls with uniform random point distribution. If z_i are uniformly distributed points from the interval $[0, 1]$, then

$$y_i = x_i + 2\rho_k(z_i - 0.5), \quad i = 1, \ldots, n\,.$$

Uniformly distributed points in the ℓ_p ball for $1 \leqslant p < \infty$ can be obtained using the acceptance-rejection method by repeatedly generating uniformly generated points

in the ℓ_∞ ball until they also lie within the ℓ_p ball. For a large space dimension n this method is very inefficient due to the large differences in volume between ℓ_∞ and ℓ_p balls. Fortunately, there are efficient algorithms for generating uniformly distributed points within ℓ_2 and ℓ_1 balls. Also, it is not difficult to obtain uniformly distributed points in spherical shells for ℓ_∞, ℓ_2 and ℓ_1 metrics.

All three neighborhood shapes ℓ_∞, ℓ_2 and ℓ_1, both in the ball (2.3) and the spherical shell (2.4) variants, were tested, while all other algorithm parameters were fixed. Additionally, the ℓ_1 ball and spherical shell with the hypergeometric random distribution was considered. The local minimizers were set to Steepest Descent for smooth test problems and Nelder-Mead with restarts for non-smooth problems. Although some other local minimizers might be more effective, they were not used as their role is not significant for testing the effectiveness of the different neighborhood shapes. A set of test functions was chosen to represent challenging problems in high-dimensional solution spaces, with many local minima, which make them hard or impossible to solve by direct methods.

Exhaustive tests led to the following conclusions. There was no significant difference in the efficiency of VNS between ball (2.3) and spherical shell (2.4) neighborhood shapes. On the other hand, the different neighborhood shapes ℓ_∞, ℓ_2 and ℓ_1 showed remarkable differences in performance: The box-shaped ℓ_∞ neighborhood was inefficient and often unsuccessful for problems of higher dimensions, while the ℓ_1 neighborhood shape with the hypergeometric random distribution performed best overall. When the implementation can combine multiple neighborhood shapes, ℓ_1 with the hypergeometric random distribution in combination with ℓ_2 with uniform random distribution were overall more efficient than all other shape combinations.

2.6. VNS with Modified Nelder Mead as local search. In [19] the authors proposed a new VNS-based heuristic that uses a modified Nelder-Mead direct search method as a local search. In GLOB-VNS, the choice of the local search procedure is left to the user. Nelder-Mead (NM) is probably the most popular direct search method for solving unconstrained continuous optimization problem, because it is easy to implement, and the objective function does not need to be smooth. The authors have included the Restarted Modified Nelder-Mead (RMNM)[59, 58] as a local search algorithm in GLOB-VNS and conducted extensive comparative analysis to show that it improves the efficiency of GLOB-VNS. The steps of RMNM used in the local search of GLOB-VNS are shown in Figure 5.

A starting point $x_1 \in \mathbb{R}^n$ is a randomly generated point and for each next iteration of the RMNM algorithm, the `Initial point` is a local optimum obtained by the last NM. The function `InitialSimplex()` creates the simplex $X = \{x_1, \ldots, x_{n+1}\}$ whose one vertex is this initial point x_1 and the remaining n vertices are generated. In the function `Shrink()`, instead of shrinking all vertices except the best one of the current simplex (as it is done in NM), the RMNM shrinks only the worst one, i.e. the one with the worst objective function value. If there is no improvement, the algorithm reconstructs the previous simplex and shrinks two of the worst vertices and so on. The pseudo-code for the `MNMIteration()` function are given in the Figure 6.

```
        Algorithm RMNM
        /* Initialization */
01      Set x_1 ← InitialPoint, j ← 0
        /* Main loop */
02      repeat the following steps until f(x_1) < f(x*)
03          Set X ← InitialSimplex(x_1, size), X* ← X, x* ← x_1;
04          repeat the following steps until the stopping condition is met
05              if not MNMIteration(X) then
06                  Set X ← X*, j ← j + 1;
07                  Set X ← Shrink(X, j);
08                  if j = n then
09                      Set X* ← X, j ← 0;
10                  endif
11              else
12                  Set X* ← X, j ← 0;
13              endif
14          end
15      end
16      return x*.
```

FIGURE 5. Algorithm RMNM

```
        Algorithm MNMIteration()
01      function MNMIteration(X,f);
02      let x_1 be the vertex of the simplex X having the lowest function value ((f(x_1) ⩽ f(x_i));
03      let x_{n+1} and x_n be vertices of X having the highest and the second highest value of f;
04      let x̄ denotes centroid for vertices of X excluding x_{n+1};
05      x_r ← x̄ + α(x̄ − x_{n+1});
06      if f(x_1) ⩽ f(x_r) < f(x_n) then
07          X ← X ∪ {x_r} ∖ {x_{n+1}} /*replace x_{n+1} with x_r*/;
08          return true;
09      endif
10      if f(x_r) < f(x_1) then
11          x_e ← x̄ + β(x_r − x̄);
12          if f(x_e) < f(x_r) then
13              X ← X ∪ {x_e} ∖ {x_{n+1}} /*replace x_{n+1} with x_e*/;
14          else
15              X ← X ∪ {x_r} ∖ {x_{n+1}} /*replace x_{n+1} with x_r*/;
16          endif
17          return true;
18      endif
19      if f(x_r) < f(x_{n+1}) then
20          x_c ← x̄ + γ(x_r − x̄);
21      else
22          x_c ← x̄ − γ(x̄ − x_{n+1});
23      endif
24      if f(x_c) < f(x_{n+1}) then
25          X ← X ∪ {x_c} ∖ {x_{n+1}} /*replace x_{n+1} with x_c*/;
26          return true;
27      else
28          return false;
29      endif
```

FIGURE 6. Algorithm MNMIteration()

Extensive comparative analyses were carried out to test the effectiveness of the proposed method. The VNS with RMNM performed the most reliable among four compared methods. The special attention was paid to the testing of non-differentiable functions.

2.7. VNS for NSEs. In [48] the C-VNS method for finding all solutions of a nonlinear system of equations (NSEs) was presented. The authors considered the NSE

$$\begin{aligned} f_1(x) &= 0 \\ f_2(x) &= 0 \\ &\dots \\ f_n(x) &= 0 \end{aligned} \tag{2.7}$$

where $x \in S = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n] \subset \mathbb{R}^n$ and $f_1, f_2, \dots, f_n$ are non-linear real-valued continuous functions on S. This problem can have a finite or infinite number of solutions, or there can be no solutions. To solve the NSE problem, the authors have used the single-objective-based optimization technique to transform the NSE problem into a single-objective optimization problem

$$\min_x F_0(x) = \sum_{i=1}^{n} |f_i(x)|^p, \quad p > 0 \tag{2.8}$$

where the value for p is usually 1 or 2. In the case that there exist many solutions, the authors have proposed a new objective function for finding all solutions, i.e. (2.8) is modified in the neighborhood of the found solutions $x^1, x^2, ..., x^k$, $k \geqslant 0$ in the following way:

$$F(x) = F_0(x) + F_k(x) = \sum_{i=1}^{n} |f_i(x)|^p + \sum_{j=1}^{k} a\varphi\Big(\frac{\|x - x^j\|}{\rho}\Big), \tag{2.9}$$

where a and ρ are some positive values and $\varphi(x)$ is a continuous real function that is positive for $|x| < 1$ and $\varphi(x) = 0$ for $|x| \geqslant 1$.

In the C-VNS implementation for NSEs, the authors applied the principles of the less-is-more approach (LIMA) [45, 15, 54], i.e. the aim was to create an efficient and effective algorithm with a minimum number of search ingredients to be specified.

The neighborhoods $\mathcal{N}_k(x)$ used in C-VNS for NSEs were taken as follows:

$$\mathcal{N}_1 = B_1 \cap S, \quad \mathcal{N}_k = (B_k \setminus B_{k-1}) \cap S, \qquad k = 2, \dots, k_{\max}$$

where $B_i = \{x| \ ||x - x^*||_\infty \leqslant r_i\}$ denotes a ℓ_∞ ball centered at current best point x^* with the radius r_i. The radii were automatically generated in such a way that they are equally spaced and such that $B_{k_{\max}}$ covers the entire solution space S. A uniform distribution was used to generate the random point from each $\mathcal{N}_k$, and the local minimizer in the local search phase was the Nelder-Mead method. The pseudo code of the proposed C-VNS for finding multiple solutions of (2.7) is shown in Figure 7.

	Algorithm for finding multiple solutions of (2.7) by C-VNS
	/* *Initialization* */
01	Select p, ρ_0, $\rho_{\min}$, a_0, $q_\rho < 1$, $q_a < 1$, tol, K and function $\varphi(x)$
02	Set $k = 0$, $\rho \leftarrow \rho_0$, $a \leftarrow a_0$
	/* *Main loop* */
03	**while** $k < K$ and $\rho \geqslant \rho_{\min}$
04	Perform global optimization for function (2.9) by C-VNS resulting with x^* as the best approximation of optimal point
05	**if** $F(x_*) < tol$ **then**
06	Set $k \leftarrow k + 1$
07	Set $sol(k) = x^*$
08	**else**
09	Set $\rho \leftarrow q_\rho \cdot \rho$
10	Set $a \leftarrow q_a \cdot a$
11	**endif**
12	**endwhile**
13	**Stop**. Points $sol(i)$, $i = 1, \ldots, k$ are approximate solutions of (2.7).

FIGURE 7. Algorithm for finding multiple solutions of (2.7) by C-VNS

One of the stopping conditions for C-VNS was $F(x) < tol$ with $tol = 10^{-7}$. If this condition is satisfied, that means that a new approximate solution has been found, so that the objective function $F(x)$ is modified and C-VNS is restarted with the search for the next solution. Otherwise ($F(x) \geqslant tol$), that means that no additional solutions have been found, so the parameter ρ is reduced and the search continues in the area closer to the solutions already found. The algorithm stops when all K solutions have been found or when ρ becomes smaller then some predefined parameter $\rho_{\min}$. The extensive computational and comparative analysis has shown that this C-VNS-based heuristic can be considered among the best methods for solving NSEs.

2.8. Some other metric C-VNS implementations. A C-VNS combined with a trust-region algorithm as a local search was considered in [8]. This heuristic was proposed for solving unconstrained nonlinear global optimization problems where the objective function is twice differentiable. The information about the function derivatives is used to generate the neighborhoods. The authors analyze the directions given by the eigenvectors of a Hessian matrix of the objective function at the point x and use these directions to generate neighbors. By assigning a higher probability to the directions (eigenvectors) so that the curvature of the function is greater, one can increase the chance of jumping to more promising regions.

A method based on C-VNS to solve the censored quantile regression (CQR) problem was proposed in [51]. This problem belongs to continuous global optimization with Powell estimator [49, 50] as objective function. This problem is difficult because the objective function which has to be minimized is neither convex nor concave. The authors approached the CQR via an unconstrained NLP. The geometry of the neighborhood structures for this implementation was induced by

ℓ_1 (2.1) and ℓ_∞ (2.2), and they used balls as in (2.3). The radii $r_1 \leqslant r_2 \leqslant ... \leqslant r_{k_{\max}}$ were generated automatically in a predefined way. The uniform distribution and the hypergeometric distribution were used. The local minimizer was the Nelder-Mead method. The results presented show that the C-VNS method can improve the quality of censored quantizing regressors estimator.

3. Non-metric C-VNS

Solving the unconstrained NLPs by C-VNS, which does not use metric distances to define neighborhood structures, has been successfully achieved in [57]. Instead of choosing ℓ_p norm, the two different neighborhoods $\mathcal{N}_1(x)$ and $\mathcal{N}_2(x)$ are used. The neighborhood $\mathcal{N}_1(x)$ is defined as follows. For a given parameter r, r random directions are generated starting from the current solution x. A one-dimensional search is performed under each of these directions, resulting in r new points. The best of these r points is selected and compared with the current solution x. If it is better, this point is chosen as the new starting solution for the next iteration. Otherwise, the search is continued within the next neighborhood of x. The neighborhood structure $\mathcal{N}_2(x)$ defines a decreasing jump. The current point $x = (x_1, \ldots, x_n)$ is moved for each coordinate $x_i, i = 1, \ldots, n$ by a value Δ_i randomly taken from the interval $[-\alpha, \alpha]$, where α is a parameter that determines the length of the jump. In this way, new points with coordinates defined as $x_i^{(new)} = x_i + \Delta_i$ or $x_i^{(new)} = x_i - \Delta_i$, define the neighborhood $\mathcal{N}_2(x)$. The plus or minus sign defines a jump direction for each coordinate. The VNS algorithm performs a local search for the best solution in $\mathcal{N}_1(x)$, and only performs a single move in $\mathcal{N}_2(x)$ if the search is trapped in the local optimum found in $\mathcal{N}_1(x)$.

3.1. Gauss-VNS. GLOB-VNS was modified to the Gaussian VNS (Gauss-VNS) in [12]. In the shaking step of a classical approach, one has to specify the distribution to obtain the random point from $\mathcal{N}_k(x)$ and a uniform distribution was the most common choice. However, since the number $k_{\max}$ of different neighborhoods $\mathcal{N}_k(x)$ is finite, it is not possible to reach any point of the solution space from the current point. For unconstrained NLPs, lower and upper bounds for variables are set to arbitrarily large negative and positive values, so that if one wants the neighborhood to be equal to the solution space, the large number $k_{\max}$ should be chosen and thus the obtained radii may be less efficient. Gauss-VNS avoids this limitation by replacing the class of neighborhoods $\{\mathcal{N}_k(x)\}_{1 \leqslant k \leqslant k_{\max}}$ of the point x by a class of probability distributions $\{\mathcal{P}_k(x)\}_{1 \leqslant k \leqslant k_{\max}}$. In Gauss-VNS implementation for each distribution $\{\mathcal{P}_k(x)\}$ a multivariate Gaussian distribution was chosen with mean x and covariance matrix Σ_k. In this way, a random point in the shaking step is generated from an n-dimensional random vector y with a density function of the form

$$\varphi(y) = \frac{1}{(2\pi)^{n/2}|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(y-x)^\top \Sigma_k^{-1}(y-x)}.$$

With this approach, in the shaking step one can jump from a current point x to any point in the solution space, so the region with the global optimum can be reached from any starting point.

Comparing Gauss-VNS which does not use metric distances to define neighborhood structures with GLOB-VNS, one can see that Gauss-VNS has less parameters which have to be specified. In the GLOB-VNS implementation, the user has to specify the number and order of combinations of geometry and distribution $(\mathcal{G}_l, \mathcal{P}_l), l = 1, \ldots, m$, while in Gauss-VNS all neighborhoods correspond to the entire solution space (S or $\mathbb{R}^n$) and only one family of distribution (Gaussian) is used. The covariance matrices Σ_k can be chosen as $\Sigma_k = \sigma_k^2 I, k = 1, 2, \ldots, k_{\max}$ where I is the identity matrix and σ_k are variances. In GLOB-VNS, the user should also specify the radii $r_k, k = 1, \ldots, k_{\max}$, while in Gauss-VNS this parameter is replaced by deviations $\sigma_k, k = 1, 2, \ldots, k_{\max}$. The pseudo-code of Gauss-VNS is given in Figure 8.

The Gauss-VNS approach was compared with various metaheuristic based algorithms (Tabu Search, Genetic Algorithm, Ant Colony, Scatter Search, Swarm Intelligence and their hybrid forms) as well as with four previous VNS-based heuristics. The computational results showed some advantages of this approach and that the obtained results outperformed the results of other heuristics in most standard test instances.

Gauss-VNS approach has also been successfully applied in the literature [22, 47, 16].

3.2. C-VNS with fat tailed distributions. The use of the Gaussian normal distribution on the entire solution space in the Gauss-VNS method has proven to be more successful than the use of traditional bounded neighborhoods, although

```
        Algorithm Gauss-VNS
        /* Initialization */
01      Select the set of covariance matrices Σ_k, k = 1, ..., k_max
02      Choose an arbitrary initial point x ∈ S
03      Set x* ← x, f* ← f(x)
        /* Main loop */
04      repeat the following steps until the stopping condition is met
05          Set k ← 1
06          repeat the following steps until k > k_max
07              Shake: Generate y from a Gaussian distribution with
                mean x* and covariance matrix Σ_k
08              Apply some local search method from y to obtain a local
                minimum y'
09              if f(y') < f* then
10                  Set x* ← y', f* ← f(y') and goto line 05
11              endif
12              Set k ← k + 1
13          end
14      end
15      Stop. Point x* is an approximate solution of the problem.
```

FIGURE 8. Algorithm Gauss-VNS

fewer parameters need to be specified. On the other hand, the normal distribution performs well for moderately close optima, but fails for distant ones.

For unbounded problems with distant optimal solutions, the normal distribution, although not limited by bounded geometric neighborhoods, has been shown to be inefficient due to the exponential decrease of the normal density function. In order to reach distant solutions more efficiently, six additional "fat-tailed" distributions were tested in [23] to generate a random point in the shaking step.

The main focus in selecting the distributions was to increase the probability of achieving larger step size values. If the solution is distant from the incumbent point, the use of such distributions should speed up the VNS algorithm.

The considered "fat-tailed" distributions are presented and described below. For all distributions, $f_X(x)$ denotes the distribution density, $F_X(x)$ the distribution function, $Mean$ the mean, and Var the variance.

(i) Folded normal distribution (FN)

$$f_X(x) = \frac{1}{B\sqrt{2\pi}}\Big[\exp\Big(-\frac{(x-A)^2}{2B^2}\Big) + \exp\Big(-\frac{(x+A)^2}{2B^2}\Big)\Big]$$

If Z is normally distributed, $|Z|$ is folded normally distributed, $A = B = (\rho_k + \rho_{k-1})/2$ is used.

(ii) Log-normal distribution (LN)

$$f_X(x) = \frac{1}{Bx\sqrt{2\pi}}\exp\Big(-\frac{(\log(x)-A)^2}{2B^2}\Big), \quad x > 0$$

$$\text{Mean} = \exp(A + B^2/2), \quad \text{Var} = (\exp(B^2) - 1)\exp(2A + B^2/2)$$

If Z is normally distributed, $\exp(Z)$ is log-normally distributed. B is a free parameter, initially set to $B = 1/2$, and A is determined so that the mean value is $(\rho_k + \rho_{k-1})/2$.

(iii) Pareto distribution (P)

$$f_X(x) = \frac{BA^B}{x^{B+1}}, \quad F_X(x) = 1 - \Big(\frac{A}{x}\Big)^B, \quad x > A$$

$$\text{Mean} = \frac{AB}{B-1}, \quad \text{Var} = \frac{A^2B}{(B-2)(B-1)^2}$$

$A = \rho_{k-1}$, B is determined so that the mean value is $(\rho_k + \rho_{k-1})/2$.

(iv) Shifted Pareto distribution (SP)

$$f_X(x) = \frac{BA^B}{(x+A)^{B+1}}, \quad F_X(x) = 1 - \Big(\frac{A}{x+A}\Big)^B, \quad x > 0$$

$$\text{Mean} = \frac{AB}{B-1} - A, \quad \text{Var} = \frac{A^2B}{(B-2)(B-1)^2}$$

If Z is Pareto distributed, then $Z - A$ is shifted Pareto distributed. B is a free parameter, initially set to $B = 2$, and A is determined so that the mean value is $(\rho_k + \rho_{k-1})/2$.

(v) Weibull distribution (W)

$$f_X(x) = \frac{C}{B}\Big(\frac{x-A}{B}\Big)^{C-1} \exp\Big[-\Big(\frac{x-A}{B}\Big)^{C}\Big],$$

$$F_X(x) = 1 - \exp\Big[-\Big(\frac{x-A}{B}\Big)^{C}\Big], \quad x > A$$

$$\text{Mean} = A + BG(1), \quad \text{Var} = (G(2) - G^2(1))B^2, \quad G(n) = \Gamma(1 + n/C)$$

$A = 0$, C is a free parameter initially set to $C = 1/2$, and B is determined so that the mean value is $(\rho_k + \rho_{k-1})/2$.

(vi) Burr distribution (B)

$$f_X(x) = \frac{BC}{A}\Big(\frac{x}{A}\Big)^{B-1}\Big[1 + \Big(\frac{x}{A}\Big)^{B}\Big]^{-C-1},$$

$$F_X(x) = 1 - \Big[1 + \Big(\frac{x}{A}\Big)^{B}\Big]^{-C}, \quad x > 0$$

$$\text{Mean} = \mu_1, \quad \text{Var} = -\mu_1^2 + \mu_2, \quad \mu_k = A^k C Beta(C - k/B, 1 + k/B)$$

B and C are free parameters, initially set to $B = 1$, $C = 2$, and A is determined so that the mean value is $(\rho_k + \rho_{k-1})/2$.

From an implementation point of view, one of the main factors for choosing these distributions was that the random numbers can be obtained quickly using uniform and normal distribution generators. The first two distributions (FN and LN) can be easily generated using the normal distribution generator, and the other four (P, SP, W and B) with the uniform distribution generator, since the functions $F_X(x)$ have a simple form and it is known that the values of $F_X(x)$ are uniformly distributed over $[0, 1]$. In the literature, there are efficient algorithms for the fast generation of normally distributed variables, such as the Ahrens–Dieter algorithm [1, 25].

In the shaking step, the random displacement from the current point x^* is generated as follows. First, a random point z is generated using a multivariate normal distribution (generated as a vector of n univariate normal distributions). Then the random direction d is generated by normalizing the vector z in the Euclidean norm, $d = z/\|z\|$. Finally, generating a univariate random value r obtained by one of the six distributions (FN, LN, P, SP W or B), a random point y from step 10 in Figure 3 is generated as $y = x^* + rd$.

The experiments were carried out on a set of functions which have a lot of local minima, as well as that the global minimum is far from the starting point. For those test functions the number of local minima may grow exponentially with the dimension of the problem. Additionally, all those test problems were modified so that the feasible set was significantly expanded and the starting point was chosen far from the optimal solution.

The experiments on test functions showed greater efficiency for most new distributions opposite to a normal distribution. Moreover, following the "less is more approach" (LIMA), the algorithm was very efficient for both close and distant optimal solutions. It combines two neighborhood structures: One being efficient for

near solutions, and the other more efficient for distant solutions. This approach, with a reduced number of parameters the user must define in advance, has shown to be robust when the position of the optimal point is unknown.

4. Constrained NLP by C-VNS

The continuous constrained nonlinear global optimization programming problem (NLP) can be formulated as follows:

$$\begin{aligned} &\min\ f(x) \\ &s.t.\ g_i(x) \leqslant 0, \quad \forall i \in \{1,2,\dots,m\} \\ &\qquad h_i(x) = 0, \quad \forall i \in \{1,2,\dots,r\} \\ &\qquad a_j \leqslant x_j \leqslant b_j, \quad \forall j \in \{1,2,\dots,n\} \end{aligned} \tag{4.1}$$

where $x \in \mathbb{R}^n$, $f\colon \mathbb{R}^n \to \mathbb{R}$, $g_i\colon \mathbb{R}^n \to \mathbb{R}$, $i = 1,2,\dots,m$ and $h_i\colon \mathbb{R}^n \to \mathbb{R}$, $i = 1,2,\dots,r$ are possibly non-linear continuous functions, and $a = (a_1,\dots,a_n)^T, b = (b_1,\dots,b_n)^T \in \mathbb{R}^n$ are the variable bounds.

A global optimization solver for general constrained NLP problems based on VNS has been implemented for the first time in [38]. Let $H_k(x)$ be hyperrectangles of growing size ($H_{k-1}(x) \subset H_k(x), k \leqslant k_{\max}$) centered at the current local minimum. Letting $\mathcal{N}_k(x) = H_k(x)$ there is a risk that a randomly selected point lies within $H_{k-1}(x)$ which has already been explored. Therefore, a neighborhood structure was induced by ℓ_∞ norm defined as (2.3), i.e. $\mathcal{N}_k(x) = H_k(x)\backslash H_{k-1}(x)$. The commercial solver SNOPT [28] was used as the local search method. The same VNS solver was used in [39] to solve the Molecular Distance Geometry Problem. The algorithm is called Double VNS with smoothing because the VNS is first used to solve a smoothed version of the problem in order to find the most promising region, and then it is used for the second time to solve the original problem restricted to this promising region.

In [43] the authors have developed a general VNS heuristic (CGVNS) for solving continuous global NLPs, which extends the unconstrained solver GLOB-VNS to the constrained case using the penalty function method. Penalization methods are used to transform a constrained optimization problem into a sequence of unconstrained (box-constrained) problems by introducing an exterior penalty function for the violation of the constraints. There are two different groups of penalization methods: Exterior and interior point penalty methods. In the former, the penalty is imposed for violating a constraint, in the latter for reaching the boundary of an inequality constraint. Since the problem (4.1) contains both inequality and equality constraints, the authors used the exterior penalty method for local minimization and combined it with VNS.

The general formulation of the unconstrained problem that minimizes the penalty function is given by:

$$\min_{a \leqslant x \leqslant b} F_{\mu,q}(x) = f(x) + \frac{1}{\mu} P_q(x) \tag{4.2}$$

with penalty function

$$P_q(x) = \sum_{i=1}^{m} (\max\{0, g_i(x)\})^q + \sum_{i=1}^{r} |h_i(x)|^q, \tag{4.3}$$

where $F_{\mu,q}(x)$ denotes the new objective function to be minimized, μ is a positive penalty parameter and $q \geqslant 0$ is a penalty exponent.

When $q = 1$ the penalty function (4.3) is called a "linear penalty function", and under certain restrictions it has been shown that there exists a sufficiently small value of μ such that the minimum of the objective function $F_{\mu,q}(x)$ in the problem (4.2) is equal to the minimum of the original constrained problem (4.1). In this case, however, the penalty function (4.3) is not differentiable at the points where $g_i(x) = 0$ or $h_i(x) = 0$. Setting $q > 1$ is the most commonly used form in practice because it guarantees the differentiability of the penalty function (4.3). Under

```
     Algorithm CGVNS with exterior point penalty method
     /* Initialization */
01   Select the set of neighborhood structures N_k, k = 1,...,k_max,
     the penalty function parameters (q, μ_0,μ_min)
     and the array of random distributions types;
02   Choose an arbitrary initial point x feasible for the box constrained
     problem (4.1);
03   Set x* ← x, f* ← F_{μ,q}(x), f*_F ← ∞; μ ← μ_0;
     /* Main loop */
04   repeat the following steps until the stopping condition is met
05       Set k ← 1;
06       repeat the following steps until k > k_max
07           for all distributions from the array do
08               Generate at random a point y ∈ N_k(x*)
09               Apply some local search method from y to obtain a local
                 minimum y';
10               if P_{μ,q}(y') ⩽ ε and f(y') < f*_F then set x*_F ← y', f*_F ← f(y');
11               if F_{μ,q}(y') < F_{μ,q}(x*) then
12                   Set x* ← y' and goto 05;
13               else
14                   Set μ ← max{μ_min, αμ};
15               endif
16           next
17           Set k ← k + 1;
18       end
19   end
20   if f*_F = ∞ (i.e., x*_F is not found) then
21       The feasible solution x*_F is obtained by a local minimizer applied
         to F_{μ,q}(x) with q = 1, μ = μ_min, starting from x*;
22   endif
23   Stop. The point x*_F is an approximate feasible solution of the problem.
```

FIGURE 9. Algorithm CGVNS with exterior point penalty method

mild conditions, the solution of a problem (4.2) converges to a local minimum of a problem (4.1) when $\mu \to 0$ [5].

The pseudo-code for CGVNS with exterior point penalty method is summarized in Figure 9.

In the initialization step, the initial solution x is found as a feasible solution of the problem (4.2) for the fixed value of the parameters q and $\mu = \mu_0$, where μ_0 is the given initial value. In the shaking step, a new point y is randomly generated as a feasible solution of the problem (4.2) for the current value of μ. In the local search step, a local minimum y' is generated starting from the point y. The penalty function is calculated for each of these points. If the value $P_{\mu,q}(y')$ is not greater than some small feasible tolerance factor ε the point is considered feasible for the constrained problem (4.1). If y' is better than x the search is continued from y' with the same value of μ. Otherwise, the search is continued from the point x and a decreased value of μ. If μ is greater than some given value $\mu_{\min}$, this parameter is updated by multiplying it by a constant factor $\alpha,\ 0 < \alpha < 1$.

In [3] the authors proposed a way to combine VNS with Mesh Adaptive Direct Search to solve non-smooth constrained optimization problems. The constrained optimization problem of packing identical spheres with unit radii into the smallest sphere (PSS) is solved in [41] using VNS and Non-linear Programming with Non-monotone and Distributed Line Search (NLPQLP) [53] as a local search.

5. Conclusion

This paper deals with a continuous variable neighborhood search (C-VNS) for solving nonlinear global optimization problems (NLP). Problems of interest are NLPs that are very difficult because the objective function contains a large number of local minima, the number of which increases exponentially with the dimension of the problem. VNS-based heuristics turn out to be a state-of-the-art method for many of these NP-hard optimization problems. Special attention is paid to C-VNS realizations with and without neighborhoods induced by metric in the shaking step.

References

[1] J. H. Ahrens, U. Dieter, *Efficient table-free sampling methods for the exponential, Cauchy, and normal distributions*, Commun. ACM **31**(11) (1988), 1330–1337.
[2] K. S. Al-Sultan, M. A. Al-Fawzan, *A tabu search Hooke and Jeeves algorithm for unconstrained optimization*, Eur. J. Oper. Res. **103**(1) (1997), 198–208.
[3] C. Audet, V. Béchard, S. Le Digabel, *Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search*, J. Glob. Optim. **41**(2) (2008),, 299–318.
[4] C. Audet, J. Brimberg, P. Hansen, S. Le Digabel, N. Mladenović, *Pooling problem: Alternate formulations and solution methods*, Manage. Sci. **50**(6) (2004), 761–776.
[5] M. Avriel, *Nonlinear Programming: Analysis and Methods*, Prentice Hall, Mineola, 1976.
[6] S. M. Baker, K. Schallau, B. H. Junker, *Comparison of different algorithms for simultaneous estimation of multiple parameters in kinetic metabolic models*, Journal of integrative bioinformatics **7**(3) (2010), 254–262.
[7] R. Battiti, G. Tecchiolli, *The continuous reactive tabu search: blending combinatorial optimization and stochastic search for global optimization* Ann. Oper. Res. **63** (1996), 151–188.
[8] M. Bierlaire, M. Thémans, N. Zufferey, *A heuristic for nonlinear global optimization* INFORMS J. Comput. **22**(1) (2010), 59–70.

[9] M, Ben-Akiva, M. Bierlaire, *Discrete choice models with applications to departure time and route choice*, In: R. W. Hall (ed.) *Handbook of Transportation Science*, Int. Ser. Oper. Res. Manag. Sci. **56**, 7–37, Springer, New York, 2003.

[10] J. Brimberg, P. Hansen, N. Mladenović, E. D. Taillard, *Improvements and comparison of heuristics for solving the uncapacitated multisource Weber problem*, Oper. Res. **48**(3) (2000), 444–460.

[11] J. Brimberg, N. Mladenović, *A variable neighborhood algorithm for solving the continuous location-allocation problem*, Stud. Locat. Anal. **10** (1996), 1–12.

[12] E. Carrizosa, M. Dražić, Z. Dražić, N. Mladenović, *Gaussian variable neighborhood search for continuous optimization*, Comput. Oper. Res. **39**(9) (2012), 2206–2213.

[13] R. Chelouah, P. Siarry, *A continuous genetic algorithm designed for the global optimization of multimodal functions*, J. Heuristics **6** (2000), 191–213.

[14] R. Chelouah, P. Siarry, *Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions*, Eur. J. Oper. Res. **148**(2) (2003), 335–348.

[15] L. R. Costa, D. Aloise, N. Mladenović, *Less is more: Basic variable neighborhood search heuristic for balanced minimum sum-of-squares clustering*, Inf. Sci. **415** (2017), 247–253.

[16] M. S. Djogatović, M. J. Stanojević, N. Mladenović, *A variable neighborhood search particle filter for bearings-only target tracking*, Comput. Oper. Res. **52** (2014), 192–202.

[17] R. E. Dorsey, W. J. Mayer, *Genetic algorithms for estimation problems with multiple optima, nondifferentiability, and other irregular features* J. Bus. Econ. Stat. **13**(1) (1995), 53–66.

[18] M. Dražić, *Influence of a neighborhood shape on the efficiency of continuous variable neighborhood search*, Yugosl. J. Oper. Res. **30**(1) (2019), 3–17.

[19] M. Dražić, Z. Dražić, N. Mladenović, D. Urošević, Q. H. Zhao, *Continuous variable neighbourhood search with modified Nelder–Mead for non-differentiable optimization*, IMA J. Manag. Math. **27**(1) (2014), 75–88.

[20] M. Dražić, V. Kovačević-Vujčić, M. Čangalović, N. Mladenović, *GLOB — a new VNS-based software for global optimization*, In: L. Liberti, N. Maculan (Eds.), *Global Optimization: From Theory to Implementation*, Nonconvex Optimization and Its Applications **84**, 135–154, Springer, New York, 2006.

[21] M. Dražić, C. Lavor, N. Maculan, N. Mladenović, *A continuous variable neighborhood search heuristic for finding the three-dimensional structure of a molecule*, Eur. J. Oper. Res. **185**(3) (2008), 1265–1273.

[22] Z. Dražić, *Gaussian variable neighborhood search for the file transfer scheduling problem*, Yugosl. J. Oper. Res. **26**(2) (2016), 173–188.

[23] Z. Dražić, *Fat-tailed distributions for continuous variable neighborhood search*, Optim. Lett. **17**(9) (2023), 2299–2320.

[24] S.-K. S Fan, Y.-C. Liang, E. Zahara, *A genetic algorithm and a particle swarm optimizer hybridized with Nelder–Mead simplex search*, Computers & industrial engineering **50**(4) (2006), 401–425.

[25] G. Fishman, *Monte Carlo: Concepts, Algorithms, and Applications*, Springer Science & Business Media, 2013.

[26] R. Fletcher, C. M. Reeves, *Function minimization by conjugate gradients*, The computer journal **7**(2)(1964), 149–154.

[27] R. Fletcher, M. J. D. Powell. *A rapidly convergent descent method for minimization*, The computer journal **6**(2) (1963), 163–168.

[28] P. E Gill, W. Murray, M. A. Saunders, *SNOPT: An SQP algorithm for large-scale constrained optimization*, SIAM Rev. **47**(1) (2005), 99–131.

[29] W. L. Goffe, G. D. Ferrier, J. Rogers, *Global optimization of statistical functions with simulated annealing* J. Econom. **60**(1–2) (1994), 65–99.

[30] P. Hansen, N. Mladenović, *Variable neighborhood search: Principles and applications*, Eur. J. Oper. Res. **130**(3) (2001), 449–467.

[31] A.-R. Hedar, M. Fukushima, *Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization*, Optimization Methods and Software **17**(5) (2002), 891–912.

[32] F. Herrera, M. Lozano, D. Molina, *Continuous scatter search: an analysis of the integration of some combination methods and improvement strategies*, Eur. J. Oper. Res. **169**(2)(2006), 450–476.

[33] R. Hooke, T. A. Jeeves, *"Direct search" solution of numerical and statistical problems*, J. ACM **8**(2) (1961), 212–229.

[34] J. L. Klepeis, M.J. Pieja, C.A. Floudas, *Hybrid global optimization algorithms for protein structure prediction: Alternating hybrids*, Biophysical journal **84**(2) (2003), 869–882.

[35] V. Kovačević-Vujčić, M. Čangalović, M. D. Ašić, L. Ivanović, M. Dražić. *Tabu search methodology in global optimization*, Comput. Math. Appl. **37**(4–5) (1999), 125–133.

[36] V. Kovačević-Vujčić, M. Čangalović, M. Dražić, N. Mladenović, *VNS-based heuristics for continuous global optimization*, L. T. H. An, P. D. Thao (Eds.), *Modelling, Computation and Optimization in Information Systems and Management Sciences*, 215–222, Hermes Science Publishing, 2004.

[37] C. Lavor, N. Maculan, *A function to test methods applied to global minimization of potential energy of molecules* Numer. Algorithms **35** (2004), 287–300.

[38] L. Liberti, M. Dražić, *Variable neighbourhood search for the global optimization of constrained NLPs*, In: I. García, L. G. Casado, E. M. T. Hendrix, B. Tóth (Eds.), *Proceedings of GO Workshop*, VII Global Optimization Workshop, June 13–17, 2005, San José, Almería, Spain, 147–151, 2005.

[39] L. Liberti, C. Lavor, N. Maculan, F. Marinelli, *Double variable neighbourhood search with smoothing for the molecular distance geometry problem*, J. Glob. Optim. **43**(2–3) (2009), 207–218.

[40] Y. Lin M. A. Stadtherr, *Advances in interval methods for deterministic global optimization in chemical engineering*, J. Glob. Optim. **29** (2004), 281–296.

[41] R. M'Hallah, A. Alkandari, N. Mladenović, *Packing unit spheres into the smallest sphere using VNS and NLP*, Comput. Oper. Res. **40**(2) (2013), 603–615.

[42] N. Mladenović, M. Dražić, M. Čangalović, V. Kovačević-Vujčić. Variable neighborhood search in global optimization. In: *Proceedings of XXX Symposium on Operations Research*, 30.09–03.10.2003, Herceg Novi, 327–330, 2003.

[43] N. Mladenović, M. Dražić, V. Kovačevic-Vujčić, M. Čangalović, *General variable neighborhood search for the continuous optimization*, Eur. J. Oper. Res. **191**(3)(2008), 753–770.

[44] N. Mladenović, J. Petrović, V. Kovačević-Vujčić, M. Čangalović, *Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search*, Eur. J. Oper. Res. **151**(2) (2003), 389–399.

[45] N. Mladenović, R. Todosijević, D. Urošević. *Less is more: Basic variable neighborhood search for minimum differential dispersion problem*, Inf. Sci. **326** (2016), 160–171.

[46] J. A. Nelder, R. Mead, *A simplex method for function minimization*, The computer journal **7**(4) (1965), 308–313.

[47] I. Nesić, A. Rakićević, A. Poledica, B. Petrović. *Gaussian variable neighborhood search and enhanced genetic algorithm for continuous optimization*, Electron. Notes Discrete Math. **39** (2012), 273–280.

[48] J. Pei, Z. Dražić, M. Dražić, N. Mladenović, P. Pardalos, *Continuous variable neighborhood search (C-VNS) for solving systems of nonlinear equations*, INFORMS J. Comput. **31**(2) (2018), 235–250.

[49] J. L. Powell, *Least absolute deviations estimation for the censored regression model*, J. Econom. **25**(3) (1984), 303–325.

[50] J. L. Powell, *Censored regression quantiles*, J. Econom. **32**(1) (1986), 143–155.

[51] R. S. Rajab, M. Dražić, N. Mladenović, P. Mladenović, K. Yu, *Fitting censored quantile regression by variable neighborhood search*, J. Glob. Optim. 63(3):481–500, 2015.

[52] H H. Rosenbrock, *An automatic method for finding the greatest or least value of a function*, The computer journal **3**(3) (1960), 175–184.
[53] K. Schittkowski, *NLPQLP: A Fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search-user's guide*, 2006.
[54] K. G. Silva, D. Aloise, S. Xavier-de-Souza, N. Mladenović, *Less is more: Simplified Nelder-Mead method for large unconstrained optimization*, Yugosl. J. Oper. Res. **28**(2) (2018), 153–169.
[55] Y. S. Teh, G. P. Rangaiah, *Tabu search for global optimization of continuous functions with application to phase equilibrium calculations*, Computers & Chemical Engineering **27**(11) (2003), 1665–1679.
[56] M. D. Toksarı, *Minimizing the multimodal functions with ant colony optimization approach*, Expert Systems with Applications **36**(3) (2009), 6030–6035.
[57] M. Duran Toksarı, E. Güner, *Solving the unconstrained optimization problem by a variable neighborhood search*, J. Math. Anal. Appl. **328**(2) (2007), 1178–1187.
[58] Q. H. Zhao, N. Mladenović, D. Urošević, *A parametric simplex search for unconstrained optimization problem*, Trans. Adv. Res. **8** (2012), 22–27.
[59] Q. H. Zhao, D. Urošević, N. Mladenović, P. Hansen, *A restarted and modified simplex search for unconstrained optimization*, Comput. Oper. Res. **36**(12) (2009), 3263–3271.