Applied Soft Computing 142 (2023) 110331

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

# Single-objective and multi-objective optimization for variance counterbalancing in stochastic learning

Dimitra G. Triantali *, Konstantinos E. Parsopoulos, Isaac E. Lagaris

*Department of Computer Science and Engineering, University of Ioannina, GR-45110 Ioannina, Greece*

ARTICLE INFO

*Article history:*
Received 19 September 2022
Received in revised form 30 March 2023
Accepted 11 April 2023
Available online 22 April 2023

*MSC:*
68T07
68T20
68W50
90C59
93E35

*Keywords:*
Variance counterbalancing
Stochastic learning
Neural networks
Single-objective optimization
Multi-objective optimization

ABSTRACT

Artificial neural networks have proved to be useful in a host of demanding applications, therefore becoming increasingly important in science and engineering. Large-scale problems constitute a challenging task for training neural networks using the stochastic gradient descent method and variations, which are based on the random selection of mini-batches of training points at every iteration. The challenge lies on the mandatory use of diminishing search step sizes in order to retain mild error fluctuations throughout the training set preserving so the quality of the network's generalization capability. Variance counterbalancing was recently proposed as a remedy for addressing the diminishing step sizes in neural network training using stochastic gradient methods. It is based on the concurrent minimization of the average mean squared error of the network along with the error variance over random sets of mini-batches. Also, it promotes the use of advanced optimization algorithms instead of the slowly convergent gradient descent. The present work aims at enriching our understanding of the original variance counterbalancing approach, as well as reformulating it as a multi-objective problem by taking advantage of its bi-objective nature. Experimental analysis reveals the performances of the studied approaches and their competitive edge over the established Adam method.



## 1. Introduction

In the era of artificial intelligence and machine learning, creating an effective neural network is particularly important in numerous scientific problems. The challenge lies in the selection of an appropriate network architecture and an efficient training method for the problem at hand. Neural architecture search (NAS), a growing field of machine learning, has offered a significant amount of research concerning the automation of neural network design [1], object detection [2], image classification [3], meta-learning [4], and hyperparameter optimization [5]. However, NAS methods are computationally expensive as they typically involve huge search spaces of diverse network architectures to test, train, and evaluate. Efficient training algorithms constitute an essential part of such demanding tasks.

Even in the case that a well suited network architecture is available for a specific problem, big data that is often met in demanding applications underline the necessity for efficient training algorithms. For instance, such problems met in Physics include the development of interatomic potentials using data from electronic structure calculations [6], the construction of high-dimensional potentials for multicomponent systems [7], the representation at several stages of silicon atoms in a neural network [8], and the modeling of potential energy surfaces based on density functional theory [9].

Typical gradient descent optimization algorithms are usually inefficient for network training in such problems within strict running time constraints. For this reason, stochastic variants such as the stochastic gradient descent (SGD) are widely used [10]. The foundation of SGD lies in the work of Robbins and Monro [11]. In essence, it constitutes a stochastic approximation of the gradient descent optimizer, where the gradient is estimated on a randomly selected subset of the data instead of the whole training set at each iteration. This can be beneficial, especially in cases of large datasets. SGD counts a number of variants such as Adam [12], AdaGrad [13], and AMSGrad [14], among others, which are all based on gradient information. As such, they are expected to exhibit inferior performance compared to quasi-Newton optimizers that maintain an approximation of the Hessian matrix. A comprehensive review of optimization methods for large-scale learning can be found in [15].

A common characteristic of SGD algorithms is the use of mini-batches [16]. A mini-batch is a typically small subset of the training set. At each iteration, the algorithm selects at random a mini-batch over which, it calculates the gradient of the mean

* Corresponding author.
*E-mail addresses:* d.triantali@uoi.gr (D.G. Triantali), kostasp@uoi.gr (K.E. Parsopoulos), lagaris@uoi.gr (I.E. Lagaris).

https://doi.org/10.1016/j.asoc.2023.110331
1568-4946/

squared error (MSE) of the network. Based on this information, a rule that depends on the specific SGD variant is used to update the network's parameters. Low MSE variations among the mini-batches are related to nice generalization properties of the network. However, this requirement typically implies small step size updates that delay the algorithm's convergence.

Recently, a number of research works have been focused specifically on the step size in SGD algorithms that employ mini-batches. De et al. [17] proposed a big batch SGD scheme that adaptively grows the batch size over time, enabling an automated learning rate selection that does not require stepsize decay. Baydin et al. [18] proposed a hypergradient descent to compute an online step size. They showed the effectiveness of their methods in a range of optimization problems. However, the proposed methods heavily depend on the choice of two parameters: the initial step size and the hypergradient learning rate.

Lagari et al. [19] proposed the *Variance CounterBalancing* (VCB) method. Instead of individual mini-batches, the method employs randomly selected sets of mini-batches. Thus, the produced objective function consists of the average MSE over the set of mini-batches along with a term that penalizes large MSE variance. The VCB approach is a remedy for the slow convergence in stochastic learning, eliminating the need to reduce the step size, while it also promotes the use of efficient optimizers such as quasi-Newton methods. Thus, it can yield significant performance enhancement as it was experimentally shown in [19].

By construction, the underlying optimization problem is bi-objective, as it is composed of two distinct objectives, i.e., the average MSE and its variance. In the standard VCB approach, the optimization problem is solved as a constrained single-objective problem using a penalty function technique [19]. Nevertheless, its inherent bi-objective form allows a multi-objective treatment that can offer solutions of comparable quality, while allowing the use of the rich multi-objective optimization algorithmic artillery.

The contribution of the present paper is twofold:

(a) It offers a comprehensive presentation of the single-objective VCB approach equipped with the BFGS algorithm with strong Wolfe–Powell line search conditions.
(b) It introduces its extension to the multi-objective framework, where the MSE and the variance over the sets of mini-batches stand for the two objective functions. For this purpose, the state-of-the-art multi-objective particle swarm optimization (MOPSO) optimizer [20] is used.

The two VCB approaches are demonstrated and compared in both noiseless and noisy function approximation problems using Radial Basis Function (RBF) networks. Moreover, they successfully compete against the established Adam method, adding merit to the general VCB framework.

The structure of the paper is as follows: Section 2 contains detailed descriptions of the two VCB variants, namely the single-objective formulation with the BFGS optimizer, and the proposed multi-objective formulation with the MOPSO optimizer. It also offers brief descriptions of the corresponding optimizers and the neural network model considered in the experimental part. Section 3 is devoted to the experimental assessment of the VCB approaches over diverse datasets. Finally, the paper concludes in Section 4.

## 2. Methods and algorithms

As baseline of our presentation, we henceforth consider the problem of approximating a continuous function, $f : X \subset \mathbb{R}^n \to \mathbb{R}$, using a neural network. The trainable parameters of the network are assumed to be collectively included in a vector $w \in W$, where $W$ is an appropriate domain related to the specific application. The network type is irrelevant for our presentation.

### 2.1. Variance counterbalancing method

Let $T$ be the given training set consisting of $\tau > 0$ training vectors,

$$T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_\tau, y_\tau)\}, \tag{1}$$

with $x_i \in X \subset \mathbb{R}^n$, and $y_i = f(x_i) \in \mathbb{R}$, for all $i = 1, 2, \ldots, \tau$. Also, let $N(x, w)$ denote the output of the neural network with parameter vector $w$ for the training vector $x$. The training procedure comprises the minimization of the MSE of the network with respect to $w$ over the whole training set, i.e.,

$$\min_{w \in W} \; E(w) = \frac{1}{\tau} \sum_{i=1}^{\tau} (N(x_i, w) - y_i)^2 .$$

This is usually achieved using gradient-based optimizers. SGD defines a family of algorithms that are commonly used for this purpose. A well-known drawback of SGD is its slow convergence, especially in cases of large training sets [19].

In order to accelerate training, the notion of *mini-batch* is used. A mini-batch is defined as a small, randomly selected subset of the training set $T$. The optimization algorithm is used to consecutively minimize the MSE of the neural network over a number of mini-batches instead of the whole training set. For a mini-batch $T_i$ consisting of $\zeta > 0$ randomly selected training points,

$$T_i = \left\{(x_{i_1}, y_{i_1}), (x_{i_2}, y_{i_2}), \ldots, (x_{i_\zeta}, y_{i_\zeta})\right\} \subset T,$$

with $\zeta \ll \tau$, the associated minimization problem is defined as

$$\min_{w \in W} \; E_i(w) = \frac{1}{\zeta} \sum_{j=1}^{\zeta} \left(N(x_{i_j}, w) - y_{i_j}\right)^2 . \tag{2}$$

A desirable minimizer $w^*$ should inherit the neural network enhanced generalization performance. This implies that the specific minimizer retains small fluctuations of the MSE among different mini-batches. According to [19], a plausible hypothesis is that

$$E(w^*) \approx E_i(w^*), \quad i = 1, 2, \ldots$$

Based on this assumption, a suitable objective function can be constructed in order to ameliorate the MSE fluctuations among the mini-batches. Considering a fixed number of mini-batches, $\eta > 1$, the error variance that is defined as

$$\sigma^2(w) = \frac{1}{\eta} \sum_{i=1}^{\eta} (E_i(w) - E(w))^2 ,$$

can be used to counterbalance $E(w)$ by introducing a penalty term. Thus, a new objective (penalty) function is formed,

$$F(w, \lambda) = E(w) + \lambda\, \sigma^2(w),$$

where $\lambda > 0$ is the penalty coefficient [19].

In order to avoid the time-consuming computation of the full MSE over the whole training set, $E(w)$ can be approximated by the average MSE over the considered mini-batches [19], i.e.,

$$\bar{E}(w) = \frac{1}{\eta} \sum_{i=1}^{\eta} E_i(w). \tag{3}$$

This implies the approximate objective function

$$\bar{F}(w, \lambda) = \bar{E}(w) + \lambda\, \bar{\sigma}^2(w),$$

where

$$\bar{\sigma}^2(w) = \frac{1}{\eta} \sum_{i=1}^{\eta} \left(E_i(w) - \bar{E}(w)\right)^2 , \tag{4}$$

is the approximated variance.

**Algorithm 1** The standard VCB method [19]

1: **procedure** VCB($T, W, \eta, \zeta, [\lambda_{\min}, \lambda_{\max}]$)
2: $c \leftarrow 0$, $w_c \leftarrow$ `initialize`$(W)$, $w^* \leftarrow w_c$ ▷ *Initial parameter vector*
3: $E(w_c) \leftarrow$ `fullMSE`$(T)$, $E^* \leftarrow E(w_c)$ ▷ *Full MSE computation*
4: **while** (Not Stopping) **do**
5: $c \leftarrow c + 1$ ▷ *Start new VCB cycle*
6: $S_c \leftarrow$ `pickMiniBatches`$(T, \eta, \zeta)$ ▷ *New set of mini-batches*
7: $(\bar{E}_c, \bar{\sigma}_c^2) \leftarrow$ `setMSE`$(S_c, w_{c-1})$ ▷ *Average MSE and variance*
8: $\lambda_c \leftarrow$ `updatePenalty`$(E^*, \bar{E}_c, \bar{\sigma}_c^2, [\lambda_{\min}, \lambda_{\max}])$ ▷ *Penalty coefficient*
9: $w_c^* \leftarrow$ `solve`$(\bar{F}(w, \lambda_c), W, S_c)$ ▷ *Solve problem of Eq. (5)*
10: $E(w_c^*) \leftarrow$ `fullMSE`$(T)$ ▷ *Full MSE calculation*
11: **if** $(E(w_c^*) < E^*)$ **then**
12: $w^* \leftarrow w_c^*$, $E^* \leftarrow E(w_c^*)$ ▷ *Update best solution*
13: **end if**
14: **end while**
15: **return** $w^*$, $E^*$
16: **end procedure**

According to the VCB method [19], the training of the network proceeds in *cycles* (epochs). At the $c$th cycle, a set $S_c$ consisting of $\eta > 1$ mini-batches is taken,

$$S_c = \left\{T_{c1}, T_{c2}, \ldots, T_{c\eta}\right\},$$

where each mini-batch consists of $\zeta$ randomly selected training points from the training set $T$. Then, the minimization problem

$$\min_{w \in W} \ \bar{F}(w, \lambda) = \bar{E}(w) + \lambda\, \bar{\sigma}^2(w), \tag{5}$$

is solved using a gradient-based optimizer. The minimization procedure provides a solution vector (network parameters) $w_c^*$ along with $\bar{E}\left(w_c^*\right)$, i.e., its average MSE for the specific set of mini-batches $S_c$. The solution vector $w_c^*$ corresponds to a specific parameter setting of the neural network. The full MSE, $E\left(w_c^*\right)$, of the network is also evaluated once at the end of each cycle, using the whole training set $T$. The solution vector that achieves the overall best full MSE value $E^*$, i.e., the lowest value of $E\left(w_c^*\right)$ over all cycles, is tracked and updated as it is the final solution eventually returned by the VCB method [19].

The penalty coefficient $\lambda$ of Eq. (5) is suggested to follow a *retarded penalty* approach [19]. More specifically, in the $c$th cycle it is updated right before the minimization of Eq. (5) as follows:

$$\lambda_c = \frac{E^* - \bar{E}\left(w_{c-1}^*\right)}{\bar{\sigma}^2(w_{c-1}^*)}, \tag{6}$$

where $E^*$ is the overall best full MSE up to the $(c-1)$-th cycle, and $\bar{E}\left(w_{c-1}^*\right)$, $\bar{\sigma}^2(w_{c-1}^*)$, are the average MSE and the corresponding variance, respectively, for the solution $w_{c-1}^*$ of the previous cycle. In order to ensure that $\lambda_c$ takes reasonable values, it is restricted within a user-defined interval $[\lambda_{\min}, \lambda_{\max}]$.

The VCB method is outlined in Algorithm 1. The input of the algorithm comprises the complete training set, $T$, the parameter domain of the network, $W$, the number $\eta > 1$ of mini-batches per mini-batch set, the number $\zeta > 0$ of training points per mini-batch, and the acceptable range $[\lambda_{\min}, \lambda_{\max}]$ for the penalty coefficient. In Step 2, the algorithm initializes the cycle counter $c$, as well as the parameter vector $w_c \in W$ and the overall best solution $w^*$. In Step 3, it evaluates the full MSE, $E(w_c)$, using the complete training set $T$, and sets the overall best solution value $E^*$.

The while-loop of Steps 4–14 defines a complete cycle of the VCB method. First, the cycle counter is updated in Step 5 and a new set $S_c$ of $\eta$ randomly selected mini-batches, each one consisting of $\zeta$ points picked from the training set $T$ is determined in Step 6. The average MSE, $\bar{E}_c$, and the corresponding variance, $\bar{\sigma}_c^2$, for $w_{c-1}$ over $S_c$ are calculated in Step 7. This allows the calculation of the new penalty coefficient value $\lambda_c$ according to Eq. (6) in Step 8. In Step 9, the optimizer is evoked for solving the minimization problem of Eq. (5), producing a new solution $w_c^* \in W$. Note that, during the minimization each candidate solution $w$ is assessed according to its MSE on the set of mini-batches only, instead of the whole training set. The resulting solution $w_c^*$ of the $c$th cycle is then evaluated over the complete training set $T$ in Step 10 and, subsequently, the overall best solution is updated in Steps 11–13. At this point, the current VCB cycle is complete and the algorithm decides to either continue its run or stop.

Various conditions can serve as the termination criteria of the algorithm. Most frequently the following two conditions are used either individually or combined:

(i) A prescribed maximum number of function evaluations, $t_{\max}$, is reached.
(ii) A prescribed maximum number of VCB cycles, $c_{\max}$, is reached.

In our approach, one function evaluation corresponds to the evaluation of the neural network or its gradient on a single training point.

### 2.2. Single-objective VCB method

The standard VCB method is based on the iterative solution of the single-objective optimization problem of Eq. (5). Since the objective function $\bar{F}(w, \lambda)$ is differentiable, gradient-based optimizers are directly applicable. Assuming a fixed value of the penalty coefficient $\lambda$, the objective function $\bar{F}(w, \lambda)$ can be written as a function in $w$, with partial derivatives

$$\frac{\partial \bar{F}(w)}{\partial w_l} = \frac{\partial \bar{E}(w)}{\partial w_l} + \lambda\, \frac{\partial \bar{\sigma}^2(w)}{\partial w_l}, \quad l = 1, 2, \ldots, \kappa, \tag{7}$$

where $\kappa > 0$ is the number of tunable parameters of the network, i.e., the dimension of the parameter vector $w$. From Eq. (3) it follows that

$$\frac{\partial \bar{E}(w)}{\partial w_l} = \frac{1}{\eta} \sum_{i=1}^{\eta} \frac{\partial E_i(w)}{\partial w_l},$$

where, taking into account Eq. (2), we have

$$\frac{\partial E_i(w)}{\partial w_l} = \frac{2}{\zeta} \sum_{j=1}^{\zeta} \left[ \left(N(x_{i_j}, w) - y_{i_j}\right) \frac{\partial N(x_{i_j}, w)}{\partial w_l} \right]. \tag{8}$$

Also, from Eq. (4), it follows that

$$\frac{\partial \bar{\sigma}^2(w)}{\partial w_l} = \frac{2}{\eta} \sum_{i=1}^{\eta} \left[ \left(E_i(w) - \bar{E}(w)\right) \left( \frac{\partial E_i(w)}{\partial w_l} - \frac{\partial \bar{E}(w)}{\partial w_l} \right) \right].$$

Thus, all the partial derivatives involved in Eq. (7) are completely defined. Their specific form for the RBF network considered later in our experimental assessment is given in Section 2.4.

The application of gradient-based algorithms such as SGD for the minimization in Eq. (5) is straightforward. The gradient of the objective function $\bar{F}(w)$ is defined as

$$\nabla_w \bar{F}(w) = \left( \frac{\partial \bar{F}(w)}{\partial w_1}, \frac{\partial \bar{F}(w)}{\partial w_2}, \ldots, \frac{\partial \bar{F}(w)}{\partial w_\kappa} \right)^T,$$

consisting of the partial derivatives defined above. Alternatively, quasi-Newton methods can be used to increase efficiency. In the present study, we consider the established BFGS algorithm combined with the strong Wolfe–Powell line search conditions. BFGS is a state-of-the-art optimization algorithm that requires

only first-order derivative information [21]. Starting from a randomly selected initial point $w^{(0)} \in W$, the algorithm produces a sequence of iterates,

$$w^{(k+1)} = w^{(k)} + \alpha_k p^{(k)},$$

where $p^{(k)}$ is the search direction and $\alpha_k > 0$ is the step size. The search direction is given as

$$p^{(k)} = -H_k \nabla_w \bar{F}\left(w^{(k)}\right),$$

where $\bar{F}$ is our objective function and $H_k$ is a symmetric, positive-definite approximation of the inverse Hessian matrix of $\bar{F}$ evaluated at the current point $w^{(k)}$. This matrix undergoes rank-2 updates at each iteration as follows:

$$H_{k+1} = H_k + \frac{\left(s_k^T y_k + y_k^T H_k y_k\right)\left(s_k s_k^T\right)}{\left(s_k^T y_k\right)^2} - \frac{H_k y_k s_k^T + s_k y_k^T H_k}{s_k^T y_k},$$

where

$$y_k = \nabla_w \bar{F}\left(w^{(k+1)}\right) - \nabla_w \bar{F}\left(w^{(k)}\right), \qquad s_k = w^{(k+1)} - w^{(k)}.$$

The initial approximation $H_0$ can be simply taken as the identity matrix.

The step size $\alpha_k > 0$ is determined through line search, accepting only values that satisfy the strong Wolfe–Powell conditions:

(a) *Armijo condition (sufficient reduction)*:

$$\bar{F}\left(w^{(k+1)}\right) \leqslant \bar{F}\left(w^{(k)}\right) + \rho_1 \alpha_k \nabla_w \bar{F}\left(w^{(k)}\right) p^{(k)}$$

(b) *Curvature condition*:

$$\left|\nabla_w \bar{F}\left(w^{(k+1)}\right)^T p^{(k)}\right| \leqslant \rho_2 \left|\nabla_w \bar{F}\left(w^{(k)}\right)^T p^{(k)}\right|$$

The parameters $0 < \rho_1 < \rho_2 < 1$ are algorithm-dependent. Thus, starting from an initial step size $\alpha_0$, the line search procedure iterates by either reducing or expanding the step until a suitable value is found or a maximum number of line search iterations, $k_{\max}^{[ls]}$, is reached.

Starting with a symmetric, positive-definite approximation of the inverse Hessian matrix, BFGS equipped with the Wolfe–Powell conditions retains these properties for all the subsequent updates. However, in some cases, the line search may exceed its allowed number of iterations before both the Wolfe–Powell conditions are satisfied. Along with possible numerical inaccuracies (e.g., due to ill-conditioned Hessians), such cases may prevent the algorithm from retaining positive definite inverse Hessians. In such cases, the Hessian can be reset to the identity matrix.

In our approach, the optimization algorithm stops its run and returns the best-detected solution back to the VCB method if the available computational budget (i.e., network evaluations) $t_{\max}$ is exceeded. Additional termination conditions may apply, depending on whether the user has specified a fixed number of VCB cycles or not. In the first case, the optimizer is assigned a predetermined computational budget for each cycle. If a local minimum is reached or convergence stagnates before exceeding the cycle's budget, the optimizer is restarted from a new initial point. For this purpose, two restarting conditions are checked after producing each new parameter vector $w^{(k+1)}$:

(a) *Restarting condition 1*
This is the relative function improvement condition defined as

$$\frac{\left|\bar{F}\left(w^{(k+1)}\right) - \bar{F}\left(w^{(k)}\right)\right|}{\left|\bar{F}\left(w^{(k+1)}\right)\right|} \leqslant \varepsilon_f, \tag{9}$$

where $\varepsilon_f > 0$ is a user-defined improvement tolerance.

(b) *Restarting condition 2*
This is the gradient-norm tolerance condition defined as

$$\left\|\nabla_w \bar{F}\left(w^{(k+1)}\right)\right\| \leqslant \varepsilon_g, \tag{10}$$

where $\varepsilon_g > 0$ is a user-defined gradient-norm tolerance.

If either condition holds, the algorithm proceeds to the next iteration with a new initial point selected at random; otherwise it continues its current trajectory.

In case the user has not specified the exact number of VCB cycles, the optimizer stops either if the maximum computational budget is reached or either of the conditions of Eqs. (9) and (10) is verified. As soon as the optimizer stops, the obtained solution is returned to the VCB method, which further progresses with its current cycle as it was previously described in Algorithm 1.

### 2.3. Multi-objective VCB method

The minimization problem of Eq. (5) consists of a penalty function that involves two distinct objectives, namely the average MSE, $\bar{E}(w)$, and the corresponding variance, $\bar{\sigma}^2(w)$, which are evaluated over the current cycle's set of mini-batches. An ideal solution would concurrently minimize both functions. However, this may be impossible due to conflicting objectives. Thus, a novel multi-objective formulation is appealing as an alternative approach.

More specifically, instead of the penalty function of Eq. (5), we consider the vectorial objective function

$$\bar{F}(w) = \left[\bar{E}(w) \;\; \bar{\sigma}^2(w)\right]^T,$$

where $w \in W$ is the vector of tunable parameters of the network. The main goal is to find a *Pareto optimal* solution, i.e., a parameter vector $w^*$ that has one of the following properties for all $w_i \in W \setminus \{w^*\}$:

(1) $\bar{E}(w_i) = \bar{E}(w^*)$ and $\bar{\sigma}^2(w_i) = \bar{\sigma}^2(w^*)$
(2) $\bar{E}(w_i) > \bar{E}(w^*)$ or $\bar{\sigma}^2(w_i) > \bar{\sigma}^2(w^*)$ (or both)

In other words, Pareto optimality dictates that moving away from $w^*$ would increase at least one of the objectives. Moreover, given two objective vectors,

$$\bar{F}(w_1) = \left[\bar{E}(w_1) \;\; \bar{\sigma}^2(w_1)\right]^T, \quad \bar{F}(w_2) = \left[\bar{E}(w_2) \;\; \bar{\sigma}^2(w_2)\right]^T,$$

we say that $\bar{F}(w_1)$ *dominates* $\bar{F}(w_2)$, and denote $\bar{F}(w_1) \preceq \bar{F}(w_2)$, if and only if it holds that:

(a) $\bar{E}(w_1) \leqslant \bar{E}(w_2)$ and $\bar{\sigma}^2(w_1) \leqslant \bar{\sigma}^2(w_2)$
(b) $\bar{E}(w_1) < \bar{E}(w_2)$ or $\bar{\sigma}^2(w_1) < \bar{\sigma}^2(w_2)$ (or both)

The set of all parameter vectors $w \in W$ with non-dominated objective vectors $\bar{F}(w)$ is called the *Pareto optimal set*,

$$\mathcal{P}^* \triangleq \left\{w \in W : \nexists\, w' \in W \text{ such that } \bar{F}(w') \preceq \bar{F}(w)\right\},$$

while their corresponding set of objective vectors is called the *Pareto front*,

$$\mathcal{PF}^* \triangleq \left\{\bar{F}(w) = \left[\bar{E}(w) \;\; \bar{\sigma}^2(w)\right]^T : \; w \in \mathcal{P}^*\right\}.$$

Thus, in the multi-objective framework, there is a whole set $\mathcal{P}^*$ of potentially interesting solutions. These solutions correspond to incomparable (non-dominated) objective vectors, granting different performance properties to the associated neural network settings. This is the main motivation for considering the multi-objective formulation in the VCB method.

Naturally, the multi-objective formulation requires suitable optimizers. Evolutionary computation has offered a variety of efficient multi-objective optimization algorithms [22]. Vector evaluated evolutionary algorithms employ a population of search

points that can concurrently approximate multiple solutions. Therefore, more than one solution of the Pareto optimal set can be detected in a single run. Among the available algorithms, NSGA-II (non-dominated sorting genetic algorithm) [23] and MOPSO (multi-objective particle swarm optimization) [20] stand in a prominent position. Regarding its representation and vector operations, MOPSO is inherently designed for continuous problems. Hence, it can be reasonably considered as a promising optimizer for the studied problem.

MOPSO adopts the standard PSO concept. It assumes $M > 2$ search agents, called the *particles*, which collectively form a population $P$, called the *swarm*,

$$P = \{w_1, w_2, \ldots, w_M\},$$

to probe the network's parameter space $W$. Each particle $w_i$ corresponds to a candidate solution, which is interpreted as an alternative parameter vector of the network, similarly to the single-objective case presented in the previous section. Thus, each particle is a $\kappa$-dimensional vector $w_i = (w_{i1}, w_{i2}, \ldots, w_{i\kappa})^T \in W$, which iteratively updates its position in $W$, in order to detect better candidate solutions. For this purpose, each particle $w_i$ assumes an adaptable position shift, $v_i$, also called its *velocity*. The velocity is updated at each iteration according to the best findings of the particle itself as well as those of the whole swarm.

While probing the search space $W$, each particle retains in a separate memory the best position it has ever visited. These positions are stored in a set of *best positions*,

$$B = \{b_1, b_2, \ldots, b_M\}.$$

Furthermore, aiming at visiting the most promising regions of $W$, the non-dominated solutions detected from the beginning of the current run of the algorithm are stored in a *repository* set,

$$R = \left\{r_1, r_2, \ldots, r_\xi\right\},$$

which is updated at each iteration.

At the $k$th iteration of the algorithm, the particles and velocities are updated as follows [20]:

$$v_{ij}^{(k+1)} = \omega\, v_{ij}^{(k)} + \mathtt{rand}(\phi_1)\left(b_{ij}^{(k)} - w_{ij}^{(k)}\right) + \mathtt{rand}(\phi_2)\left(r_{lj} - w_{ij}^{(k)}\right), \tag{11}$$

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + v_{ij}^{(k+1)}, \tag{12}$$

for all $i = 1, 2, \ldots, M$, and $j = 1, 2, \ldots, \kappa$, where $\omega$ is an *inertia coefficient* that restricts the magnitude of the velocities; $\mathtt{rand}(\phi)$ is a random number generator that returns a uniformly distributed decimal random number in the interval $[0, \phi]$ at each call; $b_i^{(k)} \in B$ is the corresponding best position of the particle; and $r_l$ is a vector selected from the repository $R$ (we explain the selection mechanism later). The parameters are usually set to their default values, namely $\omega = 0.729$, $\phi_1 = \phi_2 = 1.49$, as suggested in the theoretical analysis of PSO in [24]. Further details on PSO can be found in [25,26].

In MOPSO, the initial swarm is randomly initialized in the search space $W$, while the initial velocities are all set to zero. Despite the use of the inertia coefficient, the velocities are also clamped by a maximum absolute value $v_{\max}$, i.e.,

$$-v_{\max} \leqslant v_{ij} \leqslant v_{\max}, \quad \forall\, i, j.$$

The value of $v_{\max}$ may be determined as a fraction of the range of the search space for each dimension component. After updating all particles and velocities and evaluating the new swarm, the best position of each particle is updated according to the Pareto domination property,

$$b_i^{(k+1)} = \begin{cases} w_i^{(k)}, & \text{if } \bar{F}\left(w_i^{(k)}\right) \preceq \bar{F}\left(b_i^{(k)}\right), \\ b_i^{(k)}, & \text{if } \bar{F}\left(b_i^{(k)}\right) \preceq \bar{F}\left(w_i^{(k)}\right), \\ \mathtt{rand}\left\{w_i^{(k)}, b_i^{(k)}\right\}, & \text{if } w_i^{(k)}, b_i^{(k)}, \text{ are incomparable,} \end{cases}$$

where $\mathtt{rand}\left\{w_i^{(k)}, b_i^{(k)}\right\}$ returns either $w_i^{(k)}$ or $b_i^{(k)}$ at random.

The repository set $R$ is a special feature of the MOPSO algorithm. Initially, $R$ admits all the particles with non-dominated objective vectors (henceforth called the *non-dominated particles*) from the initial swarm. At each iteration, the non-dominated particles of the updated swarm are merged with $R$, and the new repository is formed by all non-dominated vectors among them. In order to prevent the uncontrollable growth of the repository, a maximum repository size shall be defined.

In the case where the number of the available non-dominated vectors exceeds the size of the repository, an *adaptive grid* procedure is applied. This procedure aims at retaining non-dominated vectors that reside in less crowded areas of the objective space. More specifically, the adaptive grid is defined in the objective space as a set of hypercubes (orthogonals in our bi-objective case) formed by partitioning the range of each objective function in a user-defined number of intervals $\delta > 0$. Then, non-dominated particles with objective vectors residing in less populated hypercubes are prioritized in $R$. The exact mechanism of the adaptive grid procedure is thoroughly described in [20].

The selection of a specific vector from $R$ for use in Eq. (11) follows a fitness-based roulette-wheel selection. First, the fitness of each non-empty hypercube of the adaptive grid is set to be inversely proportional to the number of non-dominated vectors it contains. Then, each hypercube is assigned a selection probability that is proportional to its fitness, and probabilistic selection is applied among the hypercubes. Eventually, a non-dominated vector is taken at random from the selected hypercube, and the corresponding vector of $R$ is used for the velocity update.

The MOPSO algorithm in [20] has an additional special feature, namely a *mutation* procedure that is used to stochastically perturb the particles' positions produced from Eqs. (11) and (12). The specific mutation aims at inducing diversity in the swarm in order to alleviate premature convergence. The specific operator calculates the mutation probability at each iteration as follows [20]:

$$\psi^{(k)} = \left(1 - \frac{k}{k_{\max}}\right)^{\frac{5}{\gamma}}, \tag{13}$$

where $k$ is the current iteration of MOPSO; $k_{\max}$ is the prescribed maximum number of iterations; and $\gamma$ is a user-defined control parameter also called the *mutation rate*. Small values of $\gamma$ produce rapidly diminishing probability values. The probability is used for deciding whether a particle will undergo mutation, as well as for scaling the magnitude of the mutation. In the latter case, the mutation is performed within a subspace, which is the fraction of the original search space that is defined by $\gamma$ [20].

In our approach, the algorithm terminates its run if the available computational budget $t_{\max}$ (network evaluations) is exceeded. Similarly to the single-objective case, additional termination conditions apply if the user specifies a fixed number of VCB cycles. In this case, the optimizer runs until it exceeds the assigned computational budget for each cycle. Also, a restarting condition is triggered in order to avoid spending function evaluations without gaining significant progress. Thus, the swarm reinitializes its positions and velocities every time it exceeds a number of iterations, $k_{\max}^{[R]}$, with no change in the repository set $R$. On the other hand, if the user has not specified the exact number of VCB

cycles, the run is terminated if either a user-defined maximum number of MOPSO iterations $k_{\max}$ is reached or the repository set $R$ has not changed for $k_{\max}^{[R]}$ consecutive iterations.

The MOPSO algorithm returns the detected Pareto optimal set $\mathcal{P}^*$ to the main VCB algorithm. It is up to the user whether the Pareto set will be fully or partially evaluated over the whole training set for the full MSE calculation step. Since the detected non-dominated solutions are incomparable with respect to their objective vectors, the ideal choice would be to evaluate the whole set. However, this requires a considerable number of network evaluations. Thus, a different strategy of assessing only one or a few Pareto solutions of interest may be adopted.

Our MOPSO implementation closely follows the original MOPSO algorithm in [20]. The reader is referred to that work for a thorough presentation and analysis of the algorithm.

### 2.4. Radial basis function neural networks

In principle, the VCB method can be applied to any neural network type. Our experimental investigation adopts the widely used RBF networks and demonstrates the workings of the proposed approach on regression problems. For completeness purpose, we offer a brief description of the RBF network and its properties (network output, derivatives, etc.), along with our implementation details.

The RBF network [27] is a type of feedforward neural network that uses radial basis activation functions. The output of the network is a linear combination of neuron weights and the output of the radial basis functions for the corresponding input pattern vector. RBF neural networks have proved to be very effective in function approximation tasks [28]. They usually consist of three layers, namely the *input layer*, the *hidden layer*, and a linear *output layer*. Assuming a training set $T$ defined as in Eq. (1), the input layer simply transfers an $n$-dimensional input pattern vector $x \in T$ to the neurons of the hidden layer. Each neuron implements a Gaussian radial basis function,

$$G(x, \mu_l, \sigma_l) = \exp\left(-\frac{(x-\mu_l)^T(x-\mu_l)}{2\,\sigma_l^2}\right),$$

where $\mu_l$ is the mean (center) vector of the $l$th neuron and $\sigma_l$ is the corresponding scalar value of the standard deviation. The output produced at the output layer of the network is a linear combination of the neurons' responses:

$$N(x, w) = \theta_0 + \sum_{l=1}^{L} \theta_l\, G(x, \mu_l, \sigma_l), \tag{14}$$

where $L$ is the number of neurons; $\theta_l$, $l = 0, 1, \ldots, L$, are the scalar weights of the neurons, with $\theta_0$ being a bias weight; and $w$ is the vector of network parameters.

In our implementation, the relevant order of the network's parameters in $w$ is as follows:

$$w = (\underbrace{\theta_0}_{\text{bias}}, \underbrace{\theta_1, \sigma_1, \mu_{11}, \mu_{12}, \ldots, \mu_{1n}}_{\text{neuron 1}}, \ldots, \underbrace{\theta_L, \sigma_L, \mu_{L1}, \mu_{L2}, \ldots, \mu_{Ln}}_{\text{neuron L}})^T. \tag{15}$$

Thus, the dimension of the network's parameter space $W$ is $\kappa = 1 + nL$. All the parameters are restricted in user-defined ranges,

$$\mu_{li} \in [\mu_{\min}, \mu_{\max}], \qquad \sigma_l \in [\sigma_{\min}, \sigma_{\max}], \qquad \theta_l \in [\theta_{\min}, \theta_{\max}],$$

for all $l = 0, 1, \ldots, L$, and $i = 1, 2, \ldots, n$.

The partial derivatives of $N(x, w)$ in Eq. (8) of the single-objective VCB approach are defined as follows:

$$\frac{\partial N(x, w)}{\partial w_k} = \begin{cases} 1, & \text{if } w_k \text{ is the } \theta_0 \text{ component,} \\ G(x, \mu_l, \sigma_l), & \text{if } w_k \text{ is a } \theta_l \text{ component,} \\ \theta_l \frac{\|x-\mu_l\|^2}{\sigma_l^3}\, G(x, \mu_l, \sigma_l), & \text{if } w_k \text{ is a } \sigma_l \text{ component,} \\ \theta_l \frac{(x_i-\mu_{li})}{\sigma_l^2}\, G(x, \mu_l, \sigma_l), & \text{if } w_k \text{ is a } \mu_{li} \text{ component,} \end{cases}$$

for all $l = 1, 2, \ldots, L$, and $i = 1, 2, \ldots, n$. Using these partial derivatives, we can fully define the gradient vector required for the application of gradient-based methods as described in Section 2.2. On the other hand, the multi-objective approach of Section 2.3 needs only the objective function value and, thus, the network output of Eq. (14) is sufficient.

## 3. Experimental analysis

Our experimental analysis aimed at assessing the VCB approaches for training RBF networks on different datasets. Taking into consideration the stochasticity of VCB as well as the selected solvers, multiple independent experiments and relevant statistical analysis of the obtained solutions were conducted. All datasets were split in training and validation sets, and the quality of the obtained networks was assessed in terms of their generalization properties on the validation set.

### 3.1. Algorithm setting and notation

All algorithmic and experimental settings are summarized in Table 1. More specifically, for each dataset, three different training–validation ratios (henceforth denoted as *T–V ratios*) were considered, namely 60% − 40%, 70% − 30%, and 80% − 20%. Regarding the number of VCB cycles, we considered two alternatives, namely a fixed number of 10 cycles, as well as the case of unspecified number of cycles, which allows the algorithm to run as long as there is available computational budget. Also, for the multi-objective VCB case, we considered 7 different strategies for the selection of Pareto optimal solutions that undergo full MSE evaluation. The different strategies are denoted as follows:

(1) *Strategy 0*: Select all vectors in the Pareto optimal set.
(2) *Strategy 1a*: Select the Pareto optimal vector that has the smallest value, $f_1^*$, for the first objective function, $\bar{E}(w)$.
(3) *Strategy 1b*: Select the Pareto optimal vector that has the smallest value, $f_2^*$, for the second objective function, $\bar{\sigma}^2(w)$.
(4) *Strategy 1c*: Select the intermediate vector of the Pareto set that has the smallest distance from the ideal point $(f_1{}^*, f_2{}^*)$.
(5) *Strategy 1r*: Select a vector from the Pareto optimal set at random.
(6) *Strategy 3a*: Select 3 vectors from the Pareto optimal set at random.
(7) *Strategy 3b*: Select 3 representative vectors, namely the three vectors of Strategies 1a, 1b, and 1c.

If the Pareto set has less than 3 vectors, Strategy 1c selects a vector at random, while Strategies 3a and 3b select the entire Pareto set. We will henceforth use the following notation for the studied VCB variants:

so/$c$ (single-objective VCB), mo/$c$/$s$ (multi-objective VCB)

where $c \in \{0, 10\}$ stands for the number of cycles, with 0 denoting the unspecified number case and 10 denoting the fixed number; $s \in \{0, 1a, 1b, 1c, 1r, 3a, 3b\}$ stands for the selected strategy.

Each mini-batch set consisted of 10 mini-batches. It was formed by selecting 10% of the training set at random, and equally allocating the patterns to the mini-batches.

In all test cases, we considered RBF networks of 5 neurons with their center vector components lying in the interval $[-5.0, 5.0]$, their standard deviations in $[0.1, 2.0]$, and their weights in

**Table 1**
Summary of the experimental settings.

| | |
|---|---|
| Single-objective VCB | |
| VCB cycles | Undefined (so/0) or 10 (so/10) |
| Penalty parameter | $\lambda \in [0.1, 70]$ |
| Maximum BFGS iterations | 100 |
| Maximum line search iterations | 100 |
| Wolfe–Powell conditions parameters | $\rho_1 = 10^{-4}$ and $\rho_2 = 0.1$ |
| Minimum relative improvement | $10^{-4}$ |
| Gradient-norm tolerance | $10^{-4}$ |
| Multi-objective VCB | |
| VCB cycles | Undefined (mo/0/∗) or 10 (mo/10/∗) |
| Pareto solutions for full MSE | 1 (mo/∗/1∗) or many (mo/∗/0, mo/∗/3∗) |
| Number of objectives | 2 |
| Swarm/repository size | 10 |
| Maximum MOPSO iterations | 100 |
| Swarm restart iterations | 20 (if repository not improved) |
| Grid intervals per dimension | 20 |
| Velocity clamping percentage | 0.2 (20%) |
| Mutation parameter | 0.25 |
| Adam | |
| Step size | 0.001 |
| Exponential decay rates | 0.9 and 0.999 |
| Error tolerance | $10^{-8}$ |
| Common settings | |
| Experiments per algorithm | 25 |
| Computational budget | $10^6$ (network evaluations) |
| Training–Validation ratio | 60%–40%, 70%–30%, 80%–20% |
| Mini-batches per set | 10 |
| Pattern vectors per set | 10% of total number/mini-batches per set |
| RBF neural networks | |
| Number of hidden neurons | 5 |
| Center components range | $[-5.0, 5.0]$ |
| Standard deviations range | $[0.1, 2.0]$ |
| Weights range | $[-20.0, 20.0]$ |

$[-20.0, 20.0]$. The available computational budget was set to $10^6$ function evaluations (i.e., single-pattern network evaluations), and 25 independent experiments were conducted for each solver. All the selected parameter values are commonly used in the relevant literature.

In the single-objective VCB approach, the BFGS optimizer described in Section 2.2 was used. Its maximum number of iterations was set to 100, which is used also as the termination condition in case of unspecified number of cycles. Moreover, line search was given a maximum of 100 iterations, while the domain of the penalty parameter $\lambda$ was set to $[0.1, 70.0]$ according to the suggestions in [19]. The minimum relative improvement and the gradient-norm tolerance were both set to $10^{-4}$. Finally, the parameters of the Wolfe–Powell conditions were set to $10^{-4}$ and 0.1, respectively, as suggested in the relevant literature [29].

In the multi-objective case, there were two objectives as explained in Section 2.3. The swarm size and repository size in MOPSO were both equal to 10. MOPSO was allowed to run for 100 iterations in the case of unspecified number of VCB cycles. In order to tackle search stagnation, failing to update the repository for 10 consecutive iterations triggered the swarm-restarting procedure. The velocity update parameters were set to their default values given in Section 2.3, while the adaptive grid assumed a partitioning of 20 intervals per dimension. The velocity clamping percentage was 20%, and the mutation parameter was equal to 0.25. These values were set according to the discussion in [20] and our relevant experience, and they were verified in a trial-and-error preprocessing phase.

The above settings defined 2 variants of the single-objective and 14 variants of the multi-objective VCB approach that were applied on the studied problem and compared among them. Further comparisons were conducted between the VCB approaches and the established Adam algorithm [12]. The default parameters of Adam were used, i.e., the step size was set to 0.001; the exponential decay rates of the moment estimates were set to 0.9 and 0.999, respectively; and the error was equal to $10^{-8}$. For fairness purpose, Adam was allowed to perform either the same number of cycles (each one with a different mini-batch) with the VCB approach or an unspecified number using the limit of 100 iterations as the termination condition of each cycle.

A flowchart of the proposed methodology is illustrated in Fig. 1.

### 3.2. Employed statistical tests

We examined four test cases (datasets). The analysis for each test case and T–V ratio included the following:

(1) Calculation of statistical MSE information, namely mean, standard deviation, median, minimum, and maximum MSE value on the validation set.
(2) Boxplots of the obtained MSEs on the validation set.
(3) Two-sample Wilcoxon rank-sum tests at 0.05 significance level for each pair of algorithms to verify the statistical significance of the observed median performance differences.

All the software for the experiments was implemented in MathWorks Matlab®.

### 3.3. Test Case 1: Function approximation dataset without noise

In the first test case, we considered the 2-dimensional Mexican Hat function, which is defined as

$$f(x) = \frac{\sin\left(x_1^2 + x_2^2\right)}{\sqrt{x_1^2 + x_2^2}},$$

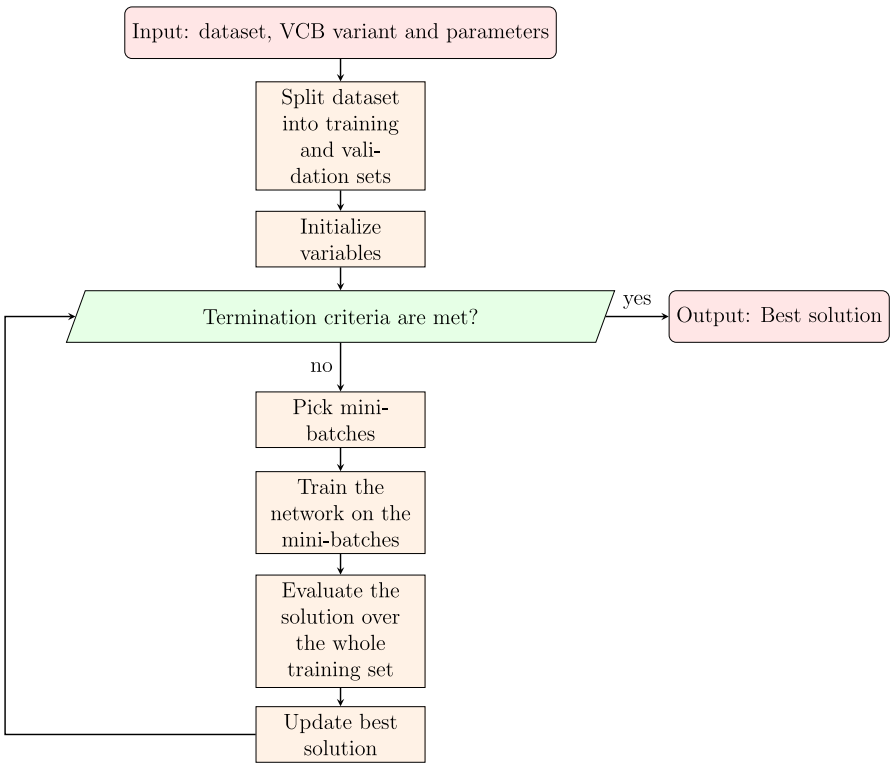


**Fig. 1.** Flowchart of the proposed methodology.

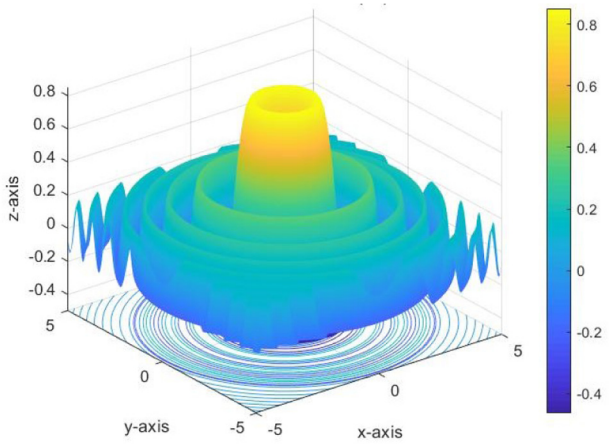


**Fig. 2.** The 2-dimensional Mexican Hat function.

with $x \in X \triangleq [-5.0, 5.0] \times [-5.0, 5.0]$. The function is illustrated in Fig. 2, and, evidently, it possesses a plethora of local minima. The dataset was generated using a 200 $\times$ 200 grid in $X$, which defines 40,000 data points that were all evaluated with $f(x)$. Among them, a percentage dictated by the corresponding T–V ratio was selected at random to form the training set, while the rest were used as the validation set.

The obtained MSE statistics are summarized in Table 2. More specifically, it reports the mean, standard deviation, median, minimum, and maximum MSE over the 25 independent experiments, for each algorithm and T–V ratio. The minimum value per column for both the single-objective and the multi-objective variants is boldfaced, while the overall best value per column is also underlined. In order to gain further insight, the obtained MSE values are also illustrated in the boxplots of Fig. 3. Note that the results of the Adam algorithm are depicted separately to avoid distortion due to the significantly larger MSE values.

In addition, statistical significance tests were conducted for each pair of algorithms to verify that the observed median performance differences are not the outcome of random fluctuations. For this purpose, the Wilcoxon rank-sum test (a.k.a. Mann Whitney U test) at 0.05 significance level was used and the results are reported in Table 3. More specifically, for each comparison of a row algorithm vs a column algorithm in the table, "=" implies statistically insignificant difference between the compared algorithms. In case of significant difference, right and left tailed tests were conducted for the pair of algorithms, with "+" denoting that the row algorithm was superior than the column algorithm, and "–" denoting the opposite result.

Table 2 and Fig. 3 reveal interesting information regarding the algorithms' performance. Firstly, we verify that the two single-objective VCB approaches exhibited different performance between them. The so/0 variant, i.e., the one with unspecified number of cycles, achieved the lowest MSE mean and standard deviation for all T–V ratios, while the so/10 variant, i.e., the one with the fixed number of cycles, was evidently inferior. The explanation behind this observation lies in the effort spent at each cycle. While so/10 had a predefined computational budget to spend at each cycle, so/0 was capable of stopping the current cycle only after achieving a satisfactory MSE value for the given mini-batch. We observed that, in some cases, this could be achieved earlier than in so/10, while in other cases multiple restarts of the optimizer were required. Thus, so/0 had the opportunity of self-tuning driven by the termination conditions of the optimizer, which were beneficial in alleviating under- and over-training. It is worth noting that so/0 frequently performed less cycles than so/10.

Interestingly, the benefits of using unspecified number of cycles were verified also for the multi-objective VCB variants. Fig. 3 clearly reveals that the mo/10/$*$ variants had inferior

**Table 2**
MSE statistics for the Mexican Hat function approximation without noise.

| T–V Ratio | Algorithm | Mean | St.D. | Median | Min | Max |
|---|---|---|---|---|---|---|
| 60% − 40% | so/0 | **7.43e−02** | **1.75e−02** | **6.92e−02** | **6.33e−02** | **1.54e−01** |
| | so/10 | 2.43e−01 | 2.25e−01 | 1.31e−01 | 6.82e−02 | 9.04e−01 |
| | mo/0/0 | 1.71e−01 | 9.28e−02 | 1.44e−01 | **5.81e−02** | 4.92e−01 |
| | mo/0/1a | 1.76e−01 | 1.36e−01 | 1.35e−01 | 6.89e−02 | 7.09e−01 |
| | mo/0/1b | 1.74e−01 | 9.99e−02 | 1.51e−01 | 7.76e−02 | 4.82e−01 |
| | mo/0/1c | **1.43e−01** | **7.62e−02** | 1.15e−01 | 6.60e−02 | **4.04e−01** |
| | mo/0/1r | 2.07e−01 | 1.92e−01 | 1.38e−01 | 6.87e−02 | 9.74e−01 |
| | mo/0/3a | 1.56e−01 | 1.19e−01 | **1.07e−01** | 7.53e−02 | 5.45e−01 |
| | mo/0/3b | 1.59e−01 | 8.94e−02 | 1.20e−01 | 6.56e−02 | 4.30e−01 |
| | mo/10/0 | 2.95e+00 | 1.95e+00 | 2.51e+00 | 8.72e−01 | 8.92e+00 |
| | mo/10/1a | 2.13e+00 | 1.15e+00 | 2.21e+00 | 4.84e−01 | 4.17e+00 |
| | mo/10/1b | 3.03e+00 | 1.93e+00 | 2.73e+00 | 1.05e+00 | 7.59e+00 |
| | mo/10/1c | 2.58e+00 | 1.39e+00 | 2.39e+00 | 4.22e−01 | 5.92e+00 |
| | mo/10/1r | 2.13e+00 | 1.26e+00 | 1.81e+00 | 4.95e−01 | 4.66e+00 |
| | mo/10/3a | 3.02e+00 | 1.71e+00 | 2.97e+00 | 3.76e−01 | 7.20e+00 |
| | mo/10/3b | 3.59e+00 | 2.47e+00 | 3.03e+00 | 3.41e−01 | 9.72e+00 |
| | Adam | 1.06e+02 | 8.83e+01 | 7.39e+01 | 1.32e+01 | 3.09e+02 |
| 70% − 30% | so/0 | **7.11e−02** | **9.63e−03** | **6.94e−02** | **5.26e−02** | **9.90e−02** |
| | so/10 | 1.20e+00 | 2.98e+00 | 2.52e−01 | 7.12e−02 | 1.14e+01 |
| | mo/0/0 | **1.81e−01** | **9.56e−02** | 1.46e−01 | 7.44e−02 | **4.37e−01** |
| | mo/0/1a | 2.77e−01 | 2.70e−01 | 1.79e−01 | 6.63e−02 | 1.15e+00 |
| | mo/0/1b | 3.58e−01 | 5.22e−01 | 2.02e−01 | **5.78e−02** | 2.27e+00 |
| | mo/0/1c | 2.30e−01 | 2.52e−01 | **1.37e−01** | 7.37e−02 | 1.15e+00 |
| | mo/0/1r | 7.77e−01 | 2.55e+00 | 1.38e−01 | 6.42e−02 | 1.30e+01 |
| | mo/0/3a | 2.55e−01 | 1.84e−01 | 1.75e−01 | 7.47e−02 | 7.21e−01 |
| | mo/0/3b | 2.20e−01 | 1.49e−01 | 1.81e−01 | 8.15e−02 | 7.71e−01 |
| | mo/10/0 | 3.55e+00 | 1.82e+00 | 3.36e+00 | 9.22e−01 | 7.83e+00 |
| | mo/10/1a | 2.85e+00 | 1.45e+00 | 2.91e+00 | 6.33e−01 | 6.78e+00 |
| | mo/10/1b | 3.52e+00 | 2.05e+00 | 3.27e+00 | 1.45e+00 | 1.03e+01 |
| | mo/10/1c | 2.74e+00 | 1.85e+00 | 2.32e+00 | 5.06e−01 | 6.35e+00 |
| | mo/10/1r | 2.82e+00 | 1.87e+00 | 2.26e+00 | 3.69e−01 | 8.69e+00 |
| | mo/10/3a | 3.67e+00 | 2.07e+00 | 3.20e+00 | 1.38e+00 | 8.65e+00 |
| | mo/10/3b | 3.55e+00 | 1.80e+00 | 3.43e+00 | 9.94e−01 | 6.47e+00 |
| | Adam | 1.29e+02 | 1.21e+02 | 7.84e+01 | 3.42e+00 | 3.46e+02 |
| 80% − 20% | so/0 | **7.49e−02** | **1.58e−02** | **7.10e−02** | **5.78e−02** | **1.40e−01** |
| | so/10 | 1.51e+00 | 2.49e+00 | 7.42e−01 | 9.85e−02 | 1.24e+01 |
| | mo/0/0 | **1.99e−01** | **8.99e−02** | 1.68e−01 | 9.47e−02 | **4.57e−01** |
| | mo/0/1a | 3.30e−01 | 2.98e−01 | 2.58e−01 | 8.73e−02 | 1.44e+00 |
| | mo/0/1b | 2.97e−01 | 2.04e−01 | 2.83e−01 | 7.71e−02 | 8.30e−01 |
| | mo/0/1c | 2.38e−01 | 1.77e−01 | 2.04e−01 | 7.32e−02 | 8.63e−01 |
| | mo/0/1r | 4.88e−01 | 8.62e−01 | **1.68e−01** | 7.37e−02 | 4.38e+00 |
| | mo/0/3a | 2.67e−01 | 2.57e−01 | 1.68e−01 | **5.99e−02** | 1.25e+00 |
| | mo/0/3b | 2.17e−01 | 1.19e−01 | 1.85e−01 | 8.04e−02 | 5.51e−01 |
| | mo/10/0 | 3.73e+00 | 2.72e+00 | 3.47e+00 | 5.50e−01 | 1.14e+01 |
| | mo/10/1a | 3.47e+00 | 1.50e+00 | 3.20e+00 | 1.22e+00 | 6.93e+00 |
| | mo/10/1b | 4.44e+00 | 2.70e+00 | 3.44e+00 | 4.89e−01 | 1.20e+01 |
| | mo/10/1c | 3.39e+00 | 1.86e+00 | 3.17e+00 | 5.73e−01 | 7.16e+00 |
| | mo/10/1r | 4.07e+00 | 2.35e+00 | 3.40e+00 | 2.12e−01 | 1.12e+01 |
| | mo/10/3a | 4.61e+00 | 2.58e+00 | 4.49e+00 | 5.86e−01 | 9.56e+00 |
| | mo/10/3b | 3.79e+00 | 2.06e+00 | 3.45e+00 | 1.46e+00 | 1.06e+01 |
| | Adam | 1.90e+02 | 1.64e+02 | 1.14e+02 | 1.48e+01 | 5.05e+02 |

performance compared to the mo/0/∗ variants. In this case, we observed a clear tendency of the MOPSO algorithm to spend larger fractions of the computational budget on the current mini-batch of each cycle, rather than promoting a higher number of cycles. Thus, our first test case clearly suggested that this approach shall be promoted instead of selecting a fixed arbitrary number of cycles.

Focusing on the multi-objective case, an interesting observation is the lack of significant effect of the full MSE evaluation. Despite the apparent small deviations in the MSE statistics between the multi-objective variants in Table 2, the statistical tests of Table 3 show that the median performance is statistically equivalent among the variants with same number of cycles (either 10 or undefined) in almost all cases. This equivalence is a consequence of the small sizes of the resulting Pareto sets of each cycle, which do not induce significant additional computational burden in the variants that employ more than one Pareto vector.

Finally, we shall underline that all VCB approaches, even the less effective mo/10/∗ variants, outperformed the established Adam approach. Moreover, a closer look at Table 2 shows that the T–V ratio interacts with performance, with higher training ratios resulting in slightly inferior solutions for so/10 (10 cycles variant). On the other hand, so/0 offers mixed findings by inconsistently improving or worsening its performance slightly when increasing the training percentage from 60% up to 80%, while the same holds for the multi-objective variants. This behavior was expected as higher training ratios are frequently associated with over-training, which is counterbalanced by the algorithms' inherent design. Nevertheless, the gradient-based single-objective approaches appear to be more susceptible to this effect than the stochastic multi-objective ones.

### 3.4. Test Case 2: Function approximation dataset with noise

Our second test case consisted of the same problem as in the previous section, although using a training set that was contaminated with Gaussian noise. More specifically, at each function value of the training set, Gaussian noise of maximum magnitude

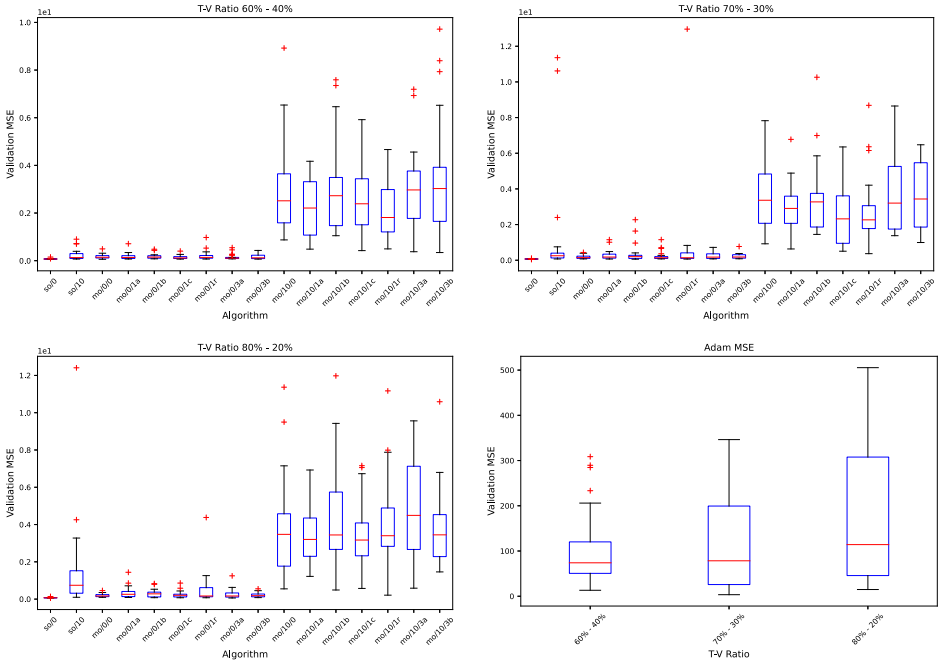


**Fig. 3.** MSE boxplots for the Mexican Hat function approximation without noise.

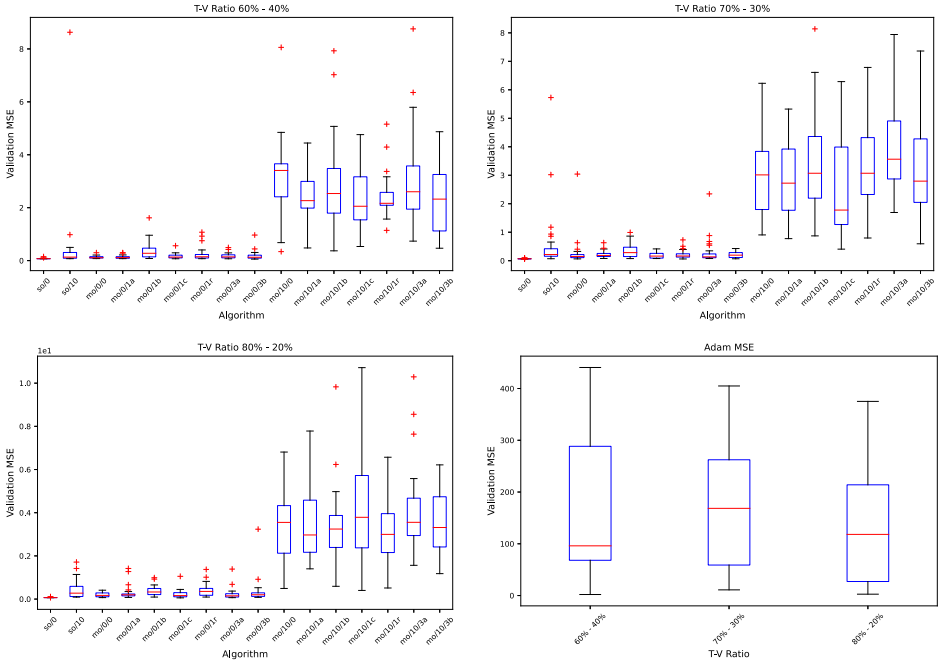


**Fig. 4.** MSE boxplots for the Mexican Hat function approximation with noise.

equal to 10% of the actual value was added. All the rest of our experimental settings were retained as in Test Case 1. This experiment was motivated by our intention to assess the effect of noise, which is habitually met in real-world applications.

The results are summarized in Tables 4 and 5, and in Fig. 4, following the same presentation style as in Test Case 1. The boxplots in Fig. 4 suggest the existence of performance differences among the algorithms compared to the noiseless case of the previous section. Although the general performance trends of the most efficient algorithms are still observed, we can see some alterations in their pairwise comparisons. For instance, the mo/10/1∗ approaches are not systematically equivalent to mo/10/0, and Adam does not produce declining performance as the training ratio increases.

Table 5 verifies that noise becomes also a tie-breaker between algorithms in cases of lower training ratio, e.g., between mo/10/1r and mo/10/0 (the former becomes better) for the 60% — 40% case, and between so/10 and mo/0/0 (the latter becomes superior) for the 70% — 30% case. However, the noise appears to induce the inverse effect for the 80% — 20% case, where overtraining is typically an issue. More specifically, for that case, so/10 achieved statistically equivalent performance with mo/0/0

**Table 3**
Wilcoxon rank-sum tests (row algorithm vs column algorithm) for the Mexican Hat function approximation without noise.

| | so/0 | so/10 | mo/0/0 | mo/0/1a | mo/0/1b | mo/0/1c | mo/0/1r | mo/0/3a | mo/0/3b | mo/10/0 | mo/10/1a | mo/10/1b | mo/10/1c | mo/10/1r | mo/10/3a | mo/10/3b | Adam |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T–V Ratio 60% – 40% | | | | | | | | | | | | | | | | | |
| so/0 | | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| so/10 | − | | = | = | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/0 | − | = | | = | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1a | − | = | = | | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1b | − | = | = | = | | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1c | − | = | = | = | = | | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1r | − | = | = | = | = | = | | = | = | + | + | + | + | + | + | + | + |
| mo/0/3a | − | = | = | = | = | = | = | | = | + | + | + | + | + | + | + | + |
| mo/0/3b | − | = | = | = | = | = | = | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | − | − | − | − | − | − | − | − | − | | = | = | = | = | = | = | + |
| mo/10/1a | − | − | − | − | − | − | − | − | − | = | | = | = | = | = | + | + |
| mo/10/1b | − | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | − | − | − | − | − | − | − | − | − | = | = | = | | = | = | = | + |
| mo/10/1r | − | − | − | − | − | − | − | − | − | = | = | = | = | | = | + | + |
| mo/10/3a | − | − | − | − | − | − | − | − | − | = | = | = | = | = | | = | + |
| mo/10/3b | − | − | − | − | − | − | − | − | − | = | − | = | = | − | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 70% – 30% | | | | | | | | | | | | | | | | | |
| so/0 | | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| so/10 | − | | = | = | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/0 | − | = | | = | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1a | − | = | = | | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1b | − | = | = | = | | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1c | − | = | = | = | = | | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1r | − | = | = | = | = | = | | = | = | + | + | + | + | + | + | + | + |
| mo/0/3a | − | = | = | = | = | = | = | | = | + | + | + | + | + | + | + | + |
| mo/0/3b | − | = | = | = | = | = | = | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | − | − | − | − | − | − | − | − | − | | = | = | = | = | = | = | + |
| mo/10/1a | − | − | − | − | − | − | − | − | − | = | | = | = | = | = | = | + |
| mo/10/1b | − | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | − | − | − | − | − | − | − | − | − | = | = | = | | = | = | = | + |
| mo/10/1r | − | − | − | − | − | − | − | − | − | = | = | = | = | | = | = | + |
| mo/10/3a | − | − | − | − | − | − | − | − | − | = | = | = | = | = | | = | + |
| mo/10/3b | − | − | − | − | − | − | − | − | − | = | = | = | = | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 80% – 20% | | | | | | | | | | | | | | | | | |
| so/0 | | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| so/10 | − | | − | − | − | − | − | − | − | + | + | + | + | + | + | + | + |
| mo/0/0 | − | + | | = | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1a | − | + | = | | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1b | − | + | = | = | | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1c | − | + | = | = | = | | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1r | − | + | = | = | = | = | | = | = | + | + | + | + | + | + | + | + |
| mo/0/3a | − | + | = | = | = | = | = | | = | + | + | + | + | + | + | + | + |
| mo/0/3b | − | + | = | = | = | = | = | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | − | − | − | − | − | − | − | − | − | | = | = | = | = | = | = | + |
| mo/10/1a | − | − | − | − | − | − | − | − | − | = | | = | = | = | = | = | + |
| mo/10/1b | − | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | − | − | − | − | − | − | − | − | − | = | = | = | | = | = | = | + |
| mo/10/1r | − | − | − | − | − | − | − | − | − | = | = | = | = | | = | = | + |
| mo/10/3a | − | − | − | − | − | − | − | − | − | = | = | = | = | = | | = | + |
| mo/10/3b | − | − | − | − | − | − | − | − | − | = | = | = | = | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |

and mo/0/1r, while the mo/0/0 variant surpassed its (formerly equivalent) mo/0/1r counterpart. Note that, in all cases, Adam exhibited inferior performance than the tested VCB approaches.

### 3.5. Test Case 3: Real estate dataset without noise

The third test case consisted of the Real Estate dataset that is available at [30]. It consists of 414 observations and 6 features, namely transaction date, house age, distance to the nearest mass rapid transit station, number of convenience stores, latitude, and longitude. The predicted value is the house price of unit area. Our experimental setting was aligned with the previous test cases.

The results are summarized in Tables 6 and 7, and in Fig. 5. An interesting observation in Fig. 5 that differs from the previous test cases is the performance profile of the so/0 variant. More specifically, although it was capable of attaining satisfactory medians (depicted as red lines in the boxplots), the range of the obtained MSE values was significantly wider than the rest of the VCB variants. This behavior suggests declining robustness of the specific variant, which can be interpreted as an effect of the small dataset size.

Contrary to so/0, the rest of the VCB variants retained consistent performance profiles, with the mo/10/$*$ variants being even more competitive against their mo/0/$*$ counterparts. Even the Adam approach was capable of achieving competitive results,

**Table 4**
MSE statistics for the Mexican Hat function approximation with noise.

| T–V Ratio | Algorithm | Mean | St.D. | Median | Min | Max |
|---|---|---|---|---|---|---|
| 60% – 40% | so/0 | **7.76e−02** | **1.68e−02** | **7.29e−02** | **6.75e−02** | **1.49e−01** |
| | so/10 | 5.46e−01 | 1.70e+00 | 1.32e−01 | 7.29e−02 | 8.63e+00 |
| | mo/0/0 | **1.29e−01** | **5.30e−02** | **1.08e−01** | 7.62e−02 | **2.97e−01** |
| | mo/0/1a | 1.32e−01 | 6.09e−02 | 1.13e−01 | 7.42e−02 | 2.98e−01 |
| | mo/0/1b | 3.80e−01 | 3.51e−01 | 2.77e−01 | 7.93e−02 | 1.62e+00 |
| | mo/0/1c | 1.65e−01 | 1.04e−01 | 1.34e−01 | 7.00e−02 | 5.60e−01 |
| | mo/0/1r | 2.51e−01 | 2.68e−01 | 1.45e−01 | 7.29e−02 | 1.07e+00 |
| | mo/0/3a | 1.74e−01 | 1.07e−01 | 1.41e−01 | 6.92e−02 | 4.93e−01 |
| | mo/0/3b | 1.84e−01 | 1.86e−01 | 1.27e−01 | **5.62e−02** | 9.68e−01 |
| | mo/10/0 | 3.21e+00 | 1.50e+00 | 3.41e+00 | 3.39e−01 | 8.06e+00 |
| | mo/10/1a | 2.43e+00 | 1.02e+00 | 2.27e+00 | 4.78e−01 | 4.44e+00 |
| | mo/10/1b | 2.97e+00 | 1.79e+00 | 2.54e+00 | 3.70e−01 | 7.93e+00 |
| | mo/10/1c | 2.36e+00 | 1.24e+00 | 2.05e+00 | 5.33e−01 | 4.76e+00 |
| | mo/10/1r | 2.42e+00 | 8.50e−01 | 2.17e+00 | 1.14e+00 | 5.16e+00 |
| | mo/10/3a | 2.99e+00 | 1.82e+00 | 2.60e+00 | 7.36e−01 | 8.76e+00 |
| | mo/10/3b | 2.27e+00 | 1.32e+00 | 2.32e+00 | 4.68e−01 | 4.87e+00 |
| | Adam | 1.64e+02 | 1.30e+02 | 9.60e+01 | 2.00e+00 | 4.41e+02 |
| 70% – 30% | so/0 | **7.00e−02** | **9.19e−03** | **6.72e−02** | **5.56e−02** | **1.00e−01** |
| | so/10 | 6.48e−01 | 1.22e+00 | 2.17e−01 | 7.01e−02 | 5.73e+00 |
| | mo/0/0 | 2.94e−01 | 5.86e−01 | 1.49e−01 | 6.37e−02 | 3.04e+00 |
| | mo/0/1a | 2.21e−01 | 1.25e−01 | 1.85e−01 | 7.90e−02 | 6.30e−01 |
| | mo/0/1b | 3.59e−01 | 2.66e−01 | 2.84e−01 | 7.48e−02 | 9.92e−01 |
| | mo/0/1c | **1.96e−01** | 1.08e−01 | 1.62e−01 | 7.59e−02 | **4.15e−01** |
| | mo/0/1r | 2.24e−01 | 1.57e−01 | 1.66e−01 | **6.17e−02** | 7.29e−01 |
| | mo/0/3a | 3.17e−01 | 4.73e−01 | **1.35e−01** | 7.65e−02 | 2.34e+00 |
| | mo/0/3b | 2.02e−01 | **1.04e−01** | 1.95e−01 | 6.82e−02 | 4.28e−01 |
| | mo/10/0 | 3.09e+00 | 1.64e+00 | 3.01e+00 | 9.06e−01 | 6.23e+00 |
| | mo/10/1a | 2.89e+00 | 1.28e+00 | 2.72e+00 | 7.75e−01 | 5.32e+00 |
| | mo/10/1b | 3.48e+00 | 1.84e+00 | 3.07e+00 | 8.72e−01 | 8.14e+00 |
| | mo/10/1c | 2.57e+00 | 1.73e+00 | 1.78e+00 | 4.06e−01 | 6.29e+00 |
| | mo/10/1r | 3.48e+00 | 1.80e+00 | 3.07e+00 | 7.97e−01 | 6.79e+00 |
| | mo/10/3a | 4.05e+00 | 1.57e+00 | 3.56e+00 | 1.69e+00 | 7.94e+00 |
| | mo/10/3b | 3.22e+00 | 1.84e+00 | 2.79e+00 | 5.94e−01 | 7.36e+00 |
| | Adam | 1.76e+02 | 1.12e+02 | 1.68e+02 | 1.10e+01 | 4.05e+02 |
| 80% – 20% | so/0 | **7.14e−02** | **9.77e−03** | **6.78e−02** | **6.37e−02** | **1.12e−01** |
| | so/10 | 4.49e−01 | 4.51e−01 | 2.74e−01 | 9.07e−02 | 1.72e+00 |
| | mo/0/0 | **2.00e−01** | **1.07e−01** | 1.70e−01 | 6.81e−02 | **4.11e−01** |
| | mo/0/1a | 3.01e−01 | 3.38e−01 | 1.97e−01 | 7.50e−02 | 1.42e+00 |
| | mo/0/1b | 3.73e−01 | 2.37e−01 | 3.29e−01 | 9.34e−02 | 9.88e−01 |
| | mo/0/1c | 2.19e−01 | 2.02e−01 | 1.63e−01 | **5.66e−02** | 1.05e+00 |
| | mo/0/1r | 3.95e−01 | 3.07e−01 | 3.55e−01 | 9.37e−02 | 1.37e+00 |
| | mo/0/3a | 2.30e−01 | 2.75e−01 | **1.51e−01** | 6.57e−02 | 1.39e+00 |
| | mo/0/3b | 3.55e−01 | 6.27e−01 | 2.00e−01 | 7.90e−02 | 3.24e+00 |
| | mo/10/0 | 3.43e+00 | 1.55e+00 | 3.55e+00 | 4.92e−01 | 6.81e+00 |
| | mo/10/1a | 3.55e+00 | 1.71e+00 | 2.97e+00 | 1.40e+00 | 7.78e+00 |
| | mo/10/1b | 3.42e+00 | 1.84e+00 | 3.24e+00 | 5.92e−01 | 9.83e+00 |
| | mo/10/1c | 4.25e+00 | 2.39e+00 | 3.79e+00 | 4.01e−01 | 1.07e+01 |
| | mo/10/1r | 3.13e+00 | 1.55e+00 | 3.00e+00 | 5.14e−01 | 6.57e+00 |
| | mo/10/3a | 4.03e+00 | 2.11e+00 | 3.55e+00 | 1.56e+00 | 1.03e+01 |
| | mo/10/3b | 3.49e+00 | 1.47e+00 | 3.31e+00 | 1.18e+00 | 6.22e+00 |
| | Adam | 1.23e+02 | 1.04e+02 | 1.18e+02 | 2.64e+00 | 3.75e+02 |

although at the presence of a few outliers. This is clearly observed in Table 7, where we can verify the statistically equivalent performance of most of the algorithms with the exception of Adam which, however, was far more competitive than in the previous test cases. Also, the marginal yet observable overall superiority of mo/0/∗ compared to mo/10/∗ variants can be noticed.

### 3.6. Test Case 4: Real estate dataset with noise

Our last test case was based on the Real Estate dataset contaminated with Gaussian noise. Similarly to Test Case 2, Gaussian noise of maximum magnitude equal to 10% of the actual value was added to each function value of the training set, while the rest of the experimental settings remained unaltered.

The results are reported in Tables 8 and 9, and in Fig. 6. The addition of noise serves again as a tie-breaker, an observation that is aligned with that of Test Case 2. For small training ratios, the addition of noise brings closer the performance of the algorithms, as it is revealed in Table 9. However, for larger training ratios, the performance of the multi-objective VCB variants differs, with mo/0/0 occupying a salient position among the most effective ones. The rest of the results came with no surprise as so/10 was still slightly better than so/0 in terms of the average MSE, although not in terms of its median. The latter implies higher standard deviations in the MSE values, which can be interpreted as reduced robustness of the specific single-objective VCB variant under the effect of noise.

### 3.7. Sensitivity analysis

Sensitivity analysis is an essential ingredient of quality assurance for methods involving multiple parameters. The proposed VCB methods include several parameters that may have an impact on their performance. The dataset size as well as the number of mini-batches per set are among the most important ones. Thus, we analyzed the performance of two promising approaches,

**Table 5**
Wilcoxon rank-sum tests (row algorithm vs column algorithm) for the Mexican Hat function approximation with noise.

| | so/0 | so/10 | mo/0/0 | mo/0/1a | mo/0/1b | mo/0/1c | mo/0/1r | mo/0/3a | mo/0/3b | mo/10/0 | mo/10/1a | mo/10/1b | mo/10/1c | mo/10/1r | mo/10/3a | mo/10/3b | Adam |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T–V Ratio 60% – 40% | | | | | | | | | | | | | | | | | |
| so/0 | | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| so/10 | − | | = | = | + | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/0 | − | = | | = | + | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1a | − | = | = | | + | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1b | − | − | − | − | | − | − | − | − | + | + | + | + | + | + | + | + |
| mo/0/1c | − | = | = | = | + | | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1r | − | = | = | = | + | = | | = | = | + | + | + | + | + | + | + | + |
| mo/0/3a | − | = | = | = | + | = | = | | = | + | + | + | + | + | + | + | + |
| mo/0/3b | − | = | = | = | + | = | = | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | − | − | − | − | − | − | − | − | − | | − | = | − | − | = | − | + |
| mo/10/1a | − | − | − | − | − | − | − | − | − | + | | = | = | = | = | = | + |
| mo/10/1b | − | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | − | − | − | − | − | − | − | − | − | + | = | = | | = | = | = | + |
| mo/10/1r | − | − | − | − | − | − | − | − | − | + | = | = | = | | = | = | + |
| mo/10/3a | − | − | − | − | − | − | − | − | − | = | = | = | = | = | | = | + |
| mo/10/3b | − | − | − | − | − | − | − | − | − | + | = | = | = | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 70% – 30% | | | | | | | | | | | | | | | | | |
| so/0 | | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| so/10 | − | | − | = | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/0 | − | + | | = | + | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1a | − | = | = | | = | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1b | − | = | − | = | | − | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1c | − | = | = | = | + | | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1r | − | = | = | = | = | = | | = | = | + | + | + | + | + | + | + | + |
| mo/0/3a | − | = | = | = | = | = | = | | = | + | + | + | + | + | + | + | + |
| mo/0/3b | − | = | = | = | = | = | = | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | − | − | − | − | − | − | − | − | − | | = | = | = | = | + | = | + |
| mo/10/1a | − | − | − | − | − | − | − | − | − | = | | = | = | = | + | = | + |
| mo/10/1b | − | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | − | − | − | − | − | − | − | − | − | = | = | = | | = | + | = | + |
| mo/10/1r | − | − | − | − | − | − | − | − | − | = | = | = | = | | = | = | + |
| mo/10/3a | − | − | − | − | − | − | − | − | − | − | − | = | − | = | | = | + |
| mo/10/3b | − | − | − | − | − | − | − | − | − | = | = | = | = | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 80% – 20% | | | | | | | | | | | | | | | | | |
| so/0 | | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| so/10 | − | | = | = | = | = | = | − | = | + | + | + | + | + | + | + | + |
| mo/0/0 | − | = | | = | + | = | + | = | = | + | + | + | + | + | + | + | + |
| mo/0/1a | − | = | = | | + | = | = | = | = | + | + | + | + | + | + | + | + |
| mo/0/1b | − | = | − | − | | − | = | − | − | + | + | + | + | + | + | + | + |
| mo/0/1c | − | = | = | = | + | | + | = | = | + | + | + | + | + | + | + | + |
| mo/0/1r | − | = | − | = | = | − | | − | − | + | + | + | + | + | + | + | + |
| mo/0/3a | − | + | = | = | + | = | + | | = | + | + | + | + | + | + | + | + |
| mo/0/3b | − | = | = | = | + | = | + | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | − | − | − | − | − | − | − | − | − | | = | = | = | = | = | = | + |
| mo/10/1a | − | − | − | − | − | − | − | − | − | = | | = | = | = | = | = | + |
| mo/10/1b | − | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | − | − | − | − | − | − | − | − | − | = | = | = | | = | = | = | + |
| mo/10/1r | − | − | − | − | − | − | − | − | − | = | = | = | = | | = | = | + |
| mo/10/3a | − | − | − | − | − | − | − | − | − | = | = | = | = | = | | = | + |
| mo/10/3b | − | − | − | − | − | − | − | − | − | = | = | = | = | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |

namely so/0, and mo/0/0, by varying each one of the two parameters while retaining the rest fixed to their values given in Section 3.1.

### 3.7.1. Dataset size

The amount of data demanded in learning tasks depends on various factors, such as the complexity of the studied problem as well as of the learning algorithm. Common practice dictates performance analysis with respect to the dataset size. In our analysis, we distinguished three cases, namely using 40%, 70%, and 100% of the original dataset.

For the Mexican Hat problem, the complete dataset consisted of 40,000 data points. Therefore, the resulting number of data points for the three cases were 16,000, 28,000, and 40,000, respectively. Fig. 7 illustrates the median and the standard deviation of the achieved validation MSE for the so/0 and mo/0/0 variants over 25 independent experiments, for the case without noise (upper row) and with noise (lower row), for all the considered T–V ratios.

Evidently, the fluctuations of the obtained MSE as the dataset size increases become stronger for the multi-objective variant

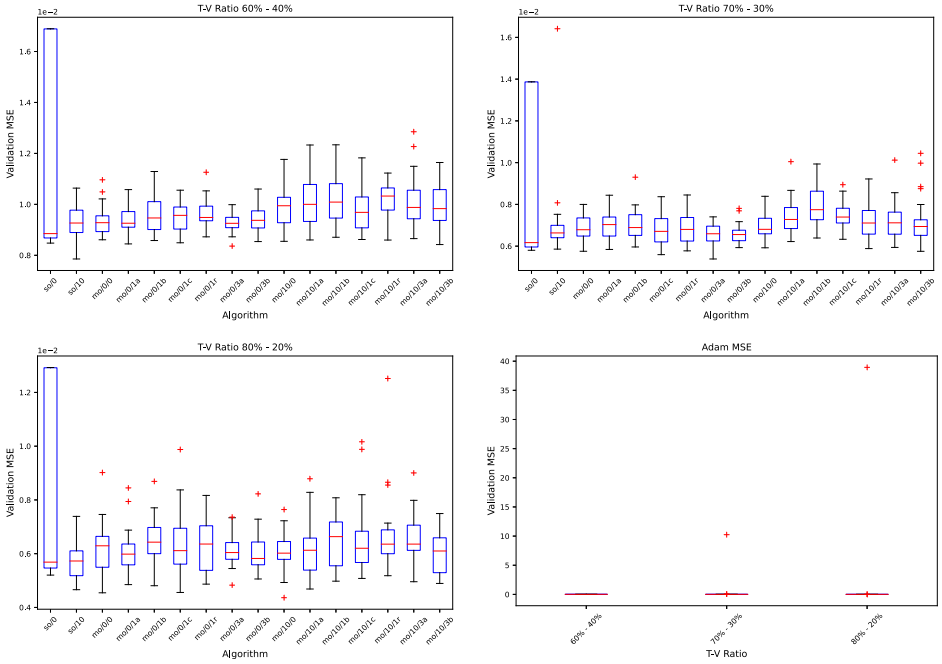


**Fig. 5.** MSE boxplots for the Real Estate dataset without noise.

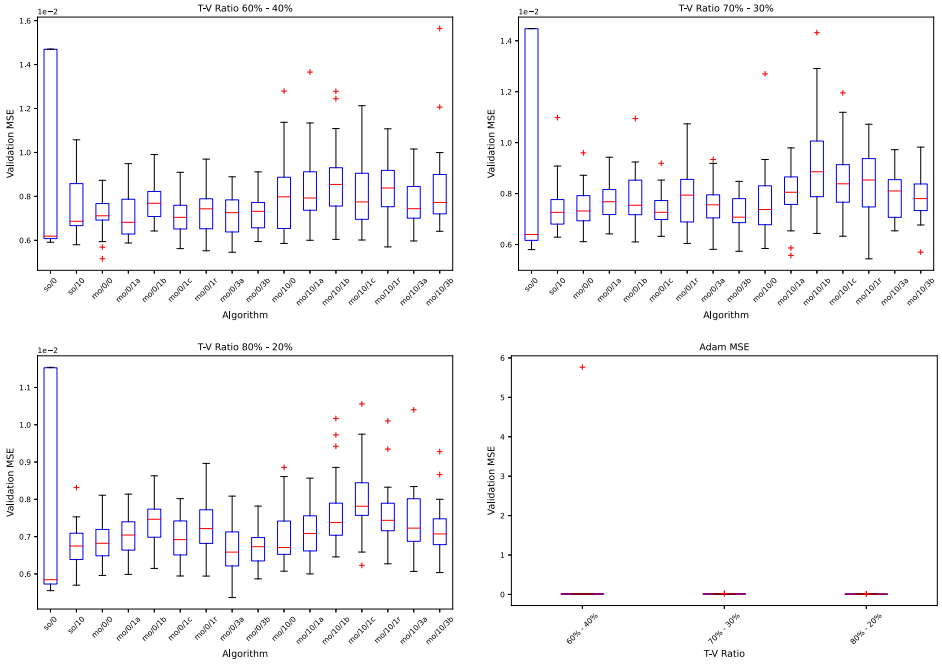


**Fig. 6.** MSE boxplots for the Real Estate dataset with noise.

than the single-objective one. This is aligned with the observations in Sections 3.3 and 3.4, where so/0 was shown to be robust, outperforming mo/0/0. The results were further verified using the Wilcoxon rank-sum test as reported in Table 10. The performance consistency was verified regardless of the addition of noise in the data.

An interesting observation is that the two variants become equivalent when only 40% of the dataset is used for the 60%–40% and 80%–20% T–V ratios. This indicates that using smaller datasets may require careful setting of the size of the training and validation sets, as it reasonably affects the performance of the solvers. In fact, as we can see in Fig. 7, using only the 0.4 fraction (i.e., 40%) of the dataset resulted in smaller differences of the MSE between the so/0 and mo/0/0 variants than those observed for higher fractions.

Regarding the Real Estate dataset, which consists of 414 data patterns, the 40% fraction consists of 166 data patterns, while the 70% fraction consists of 290 data patterns. Fig. 8 reveals that mo/0/0 was by far more robust than the so/0 approach, exhibiting smaller performance fluctuations both in the noisy as well as in the noiseless case. Moreover, there was a clear improving trend of the results as the dataset size increased.

Despite the evident differences in robustness, the results reported in Table 11 reveal that the two methods are statistically

**Table 6**
MSE statistics for the Real Estate dataset without noise.

| T–V Ratio | Algorithm | Mean | St.D. | Median | Min | Max |
|---|---|---|---|---|---|---|
| 60% — 40% | so/0 | 1.20e−02 | 4.09e−03 | **8.85e−03** | 8.48e−03 | 1.69e−02 |
| | so/10 | **9.29e−03** | **6.52e−04** | 9.27e−03 | **7.86e−03** | **1.06e−02** |
| | mo/0/0 | 9.36e−03 | 5.82e−04 | 9.28e−03 | 8.61e−03 | 1.10e−02 |
| | mo/0/1a | 9.39e−03 | 5.60e−04 | 9.26e−03 | 8.45e−03 | 1.06e−02 |
| | mo/0/1b | 9.59e−03 | 7.14e−04 | 9.47e−03 | 8.58e−03 | 1.13e−02 |
| | mo/0/1c | 9.54e−03 | 5.53e−04 | 9.57e−03 | 8.49e−03 | 1.06e−02 |
| | mo/0/1r | 9.61e−03 | 5.53e−04 | 9.49e−03 | 8.73e−03 | 1.13e−02 |
| | mo/0/3a | **9.28e−03** | **4.03e−04** | **9.26e−03** | **8.36e−03** | **9.99e−03** |
| | mo/0/3b | 9.50e−03 | 5.57e−04 | 9.37e−03 | 8.54e−03 | 1.06e−02 |
| | mo/10/0 | 9.87e−03 | 7.76e−04 | 9.95e−03 | 8.55e−03 | 1.18e−02 |
| | mo/10/1a | 1.02e−02 | 9.75e−04 | 1.00e−02 | 8.60e−03 | 1.23e−02 |
| | mo/10/1b | 1.02e−02 | 9.60e−04 | 1.01e−02 | 8.71e−03 | 1.23e−02 |
| | mo/10/1c | 9.75e−03 | 8.72e−04 | 9.69e−03 | 8.62e−03 | 1.18e−02 |
| | mo/10/1r | 1.01e−02 | 7.22e−04 | 1.03e−02 | 8.60e−03 | 1.12e−02 |
| | mo/10/3a | 1.01e−02 | 1.05e−03 | 9.88e−03 | 8.66e−03 | 1.28e−02 |
| | mo/10/3b | 9.91e−03 | 7.55e−04 | 9.83e−03 | 8.42e−03 | 1.16e−02 |
| | Adam | 1.28e−02 | 3.67e−03 | 1.10e−02 | 7.88e−03 | 2.05e−02 |
| 70% — 30% | so/0 | 8.85e−03 | 3.84e−03 | **6.17e−03** | **5.80e−03** | **1.39e−02** |
| | so/10 | **7.07e−03** | **2.01e−03** | 6.63e−03 | 5.86e−03 | 1.64e−02 |
| | mo/0/0 | 6.87e−03 | 6.42e−04 | 6.79e−03 | 5.76e−03 | 8.00e−03 |
| | mo/0/1a | 6.97e−03 | 6.57e−04 | 7.03e−03 | 5.84e−03 | 8.44e−03 |
| | mo/0/1b | 7.05e−03 | 7.47e−04 | 6.89e−03 | 5.96e−03 | 9.30e−03 |
| | mo/0/1c | 6.78e−03 | 7.22e−04 | 6.71e−03 | 5.59e−03 | 8.37e−03 |
| | mo/0/1r | 6.82e−03 | 6.73e−04 | 6.80e−03 | 5.77e−03 | 8.45e−03 |
| | mo/0/3a | **6.56e−03** | 5.40e−04 | 6.59e−03 | **5.39e−03** | **7.40e−03** |
| | mo/0/3b | 6.62e−03 | **4.67e−04** | **6.56e−03** | 5.93e−03 | 7.81e−03 |
| | mo/10/0 | 6.98e−03 | 6.23e−04 | 6.81e−03 | 5.92e−03 | 8.39e−03 |
| | mo/10/1a | 7.43e−03 | 9.03e−04 | 7.28e−03 | 6.22e−03 | 1.00e−02 |
| | mo/10/1b | 7.92e−03 | 1.02e−03 | 7.75e−03 | 6.39e−03 | 9.94e−03 |
| | mo/10/1c | 7.47e−03 | 6.37e−04 | 7.39e−03 | 6.33e−03 | 8.94e−03 |
| | mo/10/1r | 7.20e−03 | 8.40e−04 | 7.11e−03 | 5.88e−03 | 9.22e−03 |
| | mo/10/3a | 7.25e−03 | 9.56e−04 | 7.12e−03 | 5.94e−03 | 1.01e−02 |
| | mo/10/3b | 7.22e−03 | 1.16e−03 | 6.94e−03 | 5.75e−03 | 1.04e−02 |
| | Adam | 4.22e−01 | 2.05e+00 | 8.48e−03 | 5.83e−03 | 1.02e+01 |
| 80% — 20% | so/0 | 8.75e−03 | 3.77e−03 | **5.69e−03** | 5.20e−03 | 1.29e−02 |
| | so/10 | **5.75e−03** | **6.90e−04** | 5.73e−03 | **4.66e−03** | **7.39e−03** |
| | mo/0/0 | 6.21e−03 | 9.71e−04 | 6.29e−03 | 4.54e−03 | 9.02e−03 |
| | mo/0/1a | 6.04e−03 | 8.43e−04 | 5.98e−03 | 4.85e−03 | 8.45e−03 |
| | mo/0/1b | 6.50e−03 | 8.62e−04 | 6.43e−03 | 4.81e−03 | 8.69e−03 |
| | mo/0/1c | 6.39e−03 | 1.23e−03 | 6.11e−03 | 4.56e−03 | 9.88e−03 |
| | mo/0/1r | 6.33e−03 | 9.86e−04 | 6.36e−03 | 4.87e−03 | 8.17e−03 |
| | mo/0/3a | 6.16e−03 | **6.12e−04** | 6.04e−03 | 4.83e−03 | **7.36e−03** |
| | mo/0/3b | **6.02e−03** | 7.35e−04 | **5.82e−03** | 5.06e−03 | 8.22e−03 |
| | mo/10/0 | 6.07e−03 | 7.46e−04 | 6.02e−03 | **4.36e−03** | 7.64e−03 |
| | mo/10/1a | 6.09e−03 | 1.04e−03 | 6.13e−03 | 4.69e−03 | 8.79e−03 |
| | mo/10/1b | 6.45e−03 | 1.03e−03 | 6.63e−03 | 4.98e−03 | 8.08e−03 |
| | mo/10/1c | 6.58e−03 | 1.34e−03 | 6.20e−03 | 5.08e−03 | 1.02e−02 |
| | mo/10/1r | 6.69e−03 | 1.48e−03 | 6.36e−03 | 5.18e−03 | 1.25e−02 |
| | mo/10/3a | 6.56e−03 | 9.68e−04 | 6.36e−03 | 4.96e−03 | 9.00e−03 |
| | mo/10/3b | 6.04e−03 | 7.66e−04 | 6.10e−03 | 4.89e−03 | 7.49e−03 |
| | Adam | 1.57e+00 | 7.79e+00 | 7.33e−03 | 4.73e−03 | 3.89e+01 |

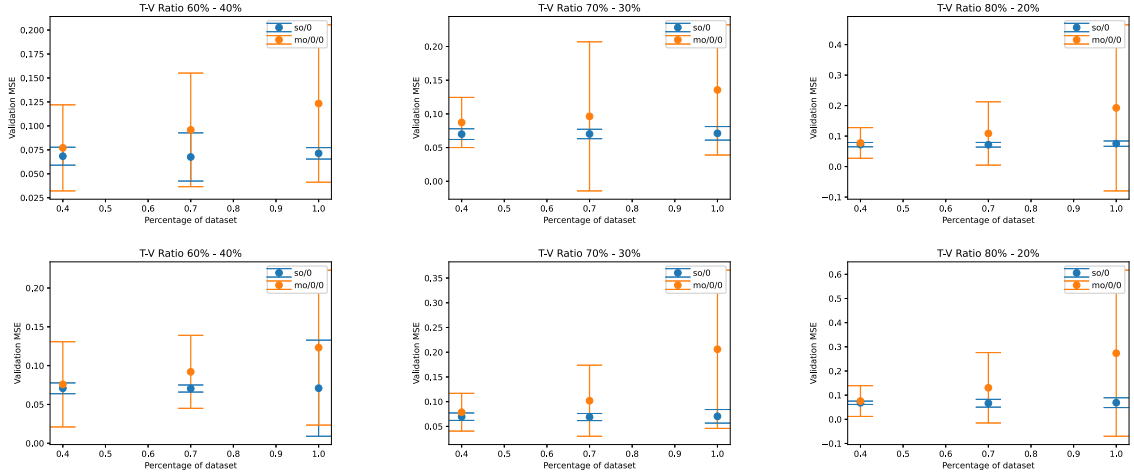


**Fig. 7.** Sensitivity analysis on the dataset size for the Mexican Hat problem without noise (upper row) and with noise (lower row).

**Table 7**
Wilcoxon rank-sum tests (row algorithm vs column algorithm) for the Real Estate dataset without noise.

| | so/0 | so/10 | mo/0/0 | mo/0/1a | mo/0/1b | mo/0/1c | mo/0/1r | mo/0/3a | mo/0/3b | mo/10/0 | mo/10/1a | mo/10/1b | mo/10/1c | mo/10/1r | mo/10/3a | mo/10/3b | Adam |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T–V Ratio 60% – 40% | | | | | | | | | | | | | | | | | |
| so/0 | | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = |
| so/10 | = | | = | = | = | = | = | = | = | + | + | + | = | + | + | + | + |
| mo/0/0 | = | = | | = | = | = | = | = | = | + | + | + | = | + | + | + | + |
| mo/0/1a | = | = | = | | = | = | = | = | = | + | + | + | = | + | + | + | + |
| mo/0/1b | = | = | = | = | | = | = | = | = | = | + | + | = | + | = | = | + |
| mo/0/1c | = | = | = | = | = | | = | = | = | = | + | + | = | + | = | = | + |
| mo/0/1r | = | = | = | = | = | = | | − | = | = | + | + | = | + | = | = | + |
| mo/0/3a | = | = | = | = | = | = | + | | = | + | + | + | = | + | + | + | + |
| mo/0/3b | = | = | = | = | = | = | = | = | | = | + | + | = | + | + | + | + |
| mo/10/0 | = | − | − | − | = | = | = | − | = | | = | = | = | = | = | = | + |
| mo/10/1a | = | − | − | − | − | − | − | − | − | = | | = | = | = | = | = | + |
| mo/10/1b | = | − | − | − | − | − | − | − | − | = | = | | = | = | = | = | + |
| mo/10/1c | = | = | = | = | = | = | = | = | = | = | = | = | | = | = | = | + |
| mo/10/1r | = | − | − | − | − | − | − | − | − | = | = | = | = | | = | = | + |
| mo/10/3a | = | − | − | − | = | = | = | − | − | = | = | = | = | = | | = | + |
| mo/10/3b | = | − | − | − | = | = | = | − | − | = | = | = | = | = | = | | + |
| Adam | = | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 70% – 30% | | | | | | | | | | | | | | | | | |
| so/0 | | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = | + |
| so/10 | = | | = | = | = | = | = | = | = | = | + | + | + | + | + | = | + |
| mo/0/0 | = | = | | = | = | = | = | = | = | = | + | + | + | = | = | = | + |
| mo/0/1a | = | = | = | | = | = | = | = | = | = | = | + | + | = | = | = | + |
| mo/0/1b | = | = | = | = | | = | = | − | − | = | = | + | + | = | = | = | + |
| mo/0/1c | = | = | = | = | = | | = | = | = | = | + | + | + | = | = | = | + |
| mo/0/1r | = | = | = | = | = | = | | = | = | = | + | + | + | = | = | = | + |
| mo/0/3a | = | = | = | = | + | = | = | | = | + | + | + | + | + | + | = | + |
| mo/0/3b | = | = | = | = | + | = | = | = | | + | + | + | + | + | + | + | + |
| mo/10/0 | = | = | = | = | = | = | = | − | − | | = | + | + | = | = | = | + |
| mo/10/1a | = | − | − | = | = | − | − | − | − | = | | = | = | = | = | = | + |
| mo/10/1b | = | − | − | − | − | − | − | − | − | − | = | | = | − | − | − | = |
| mo/10/1c | = | − | − | − | − | − | − | − | − | − | = | = | | = | = | − | + |
| mo/10/1r | = | − | = | = | = | = | = | − | − | = | = | + | = | | = | = | + |
| mo/10/3a | = | − | = | = | = | = | = | − | − | = | = | + | = | = | | = | + |
| mo/10/3b | = | = | = | = | = | = | = | = | − | = | = | + | + | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | = | − | − | − | − | |
| T–V Ratio 80% – 20% | | | | | | | | | | | | | | | | | |
| so/0 | | − | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = |
| so/10 | + | | = | = | + | = | + | + | = | = | = | + | + | + | + | = | + |
| mo/0/0 | = | = | | = | = | = | = | = | = | = | = | = | = | = | = | = | + |
| mo/0/1a | = | = | = | | + | = | = | = | = | = | = | = | = | + | + | = | + |
| mo/0/1b | = | − | = | − | | = | = | = | − | = | = | = | = | = | = | = | + |
| mo/0/1c | = | = | = | = | = | | = | = | = | = | = | = | = | = | = | = | + |
| mo/0/1r | = | − | = | = | = | = | | = | = | = | = | = | = | = | = | = | + |
| mo/0/3a | = | − | = | = | = | = | = | | = | = | = | = | = | = | = | = | + |
| mo/0/3b | = | = | = | = | + | = | = | = | | = | = | = | = | + | + | = | + |
| mo/10/0 | = | = | = | = | = | = | = | = | = | | = | = | = | = | + | = | + |
| mo/10/1a | = | = | = | = | = | = | = | = | = | = | | = | = | = | = | = | + |
| mo/10/1b | = | − | = | = | = | = | = | = | = | = | = | | = | = | = | = | + |
| mo/10/1c | = | − | = | = | = | = | = | = | = | = | = | = | | = | = | = | + |
| mo/10/1r | = | − | = | − | = | = | = | = | − | = | = | = | = | | = | = | + |
| mo/10/3a | = | − | = | − | = | = | = | = | − | − | = | = | = | = | | − | + |
| mo/10/3b | = | = | = | = | = | = | = | = | = | = | = | = | = | = | + | | + |
| Adam | = | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |

equivalent regarding their medians in almost all cases, with the only exception when 40% of the noisy dataset is used, with the training set occupying 60% of it. In that case, the neural network uses just 100 training patterns.

Summarizing, the sensitivity analysis on the dataset size for the selected variants verified the robustness and consistency of both variants under varying dataset size, with the single-objective variant being superior under bigger datasets, and the multi-objective variant dominating under smaller datasets.

### 3.7.2. Number of mini-batches per set

The second parameter considered in our sensitivity analysis was the number of mini-batches per set. For the large dataset of the Mexican Hat problem, four different numbers of mini-batches were examined, namely 2, 5, 10, and 20. Fig. 9 reveals that when employing 2 batches, the mo/0/0 variant exhibits strong variation in results. As the number of batches increases, its performance becomes more promising. The so/0 variant exhibits more stable behavior in terms of its median MSE value. Statistical comparisons of the two approaches are reported in Table 12, clearly verifying that, in all cases, the so/0 variant was statistically superior to mo/0/0. This result is in line with the previous analysis for the dataset size as well as with the main results presented in Sections 3.3 and 3.4.

On the other hand, the dataset used in the Real Estate problem is quite smaller in size. As a consequence, the studied number

**Table 8**
MSE statistics for the Real Estate dataset with noise.

| T–V Ratio | Algorithm | Mean | St.D. | Median | Min | Max |
|---|---|---|---|---|---|---|
| 60% – 40% | so/0 | 8.90e−03 | 4.07e−03 | **6.19e−03** | 5.91e−03 | 1.47e−02 |
| | so/10 | **7.47e−03** | **1.29e−03** | 6.87e−03 | **5.80e−03** | **1.06e−02** |
| | mo/0/0 | 7.15e−03 | 8.25e−04 | 7.12e−03 | **5.16e−03** | **8.73e−03** |
| | mo/0/1a | 7.25e−03 | 1.10e−03 | **6.82e−03** | 5.88e−03 | 9.49e−03 |
| | mo/0/1b | 7.71e−03 | 8.33e−04 | 7.69e−03 | 6.43e−03 | 9.90e−03 |
| | mo/0/1c | **7.09e−03** | 8.37e−04 | 7.05e−03 | 5.62e−03 | 9.10e−03 |
| | mo/0/1r | 7.31e−03 | 1.07e−03 | 7.43e−03 | 5.53e−03 | 9.70e−03 |
| | mo/0/3a | 7.18e−03 | 8.83e−04 | 7.26e−03 | 5.46e−03 | 8.89e−03 |
| | mo/0/3b | 7.25e−03 | **8.15e−04** | 7.32e−03 | 5.94e−03 | 9.12e−03 |
| | mo/10/0 | 8.05e−03 | 1.77e−03 | 7.98e−03 | 5.86e−03 | 1.28e−02 |
| | mo/10/1a | 8.37e−03 | 1.70e−03 | 7.93e−03 | 6.00e−03 | 1.37e−02 |
| | mo/10/1b | 8.75e−03 | 1.81e−03 | 8.55e−03 | 6.04e−03 | 1.28e−02 |
| | mo/10/1c | 8.09e−03 | 1.39e−03 | 7.75e−03 | 6.02e−03 | 1.21e−02 |
| | mo/10/1r | 8.42e−03 | 1.32e−03 | 8.38e−03 | 5.70e−03 | 1.11e−02 |
| | mo/10/3a | 7.72e−03 | 1.10e−03 | 7.44e−03 | 5.97e−03 | 1.02e−02 |
| | mo/10/3b | 8.32e−03 | 1.99e−03 | 7.73e−03 | 6.41e−03 | 1.56e−02 |
| | Adam | 2.41e−01 | 1.15e+00 | 1.07e−02 | 6.08e−03 | 5.77e+00 |
| 70% – 30% | so/0 | 1.01e−02 | 4.26e−03 | **6.39e−03** | **5.80e−03** | 1.45e−02 |
| | so/10 | **7.52e−03** | **1.03e−03** | 7.26e−03 | 6.29e−03 | **1.10e−02** |
| | mo/0/0 | 7.52e−03 | 7.99e−04 | 7.32e−03 | 6.11e−03 | 9.60e−03 |
| | mo/0/1a | 7.81e−03 | 8.71e−04 | 7.68e−03 | 6.42e−03 | 9.43e−03 |
| | mo/0/1b | 7.87e−03 | 1.05e−03 | 7.54e−03 | 6.10e−03 | 1.09e−02 |
| | mo/0/1c | 7.44e−03 | 6.73e−04 | 7.26e−03 | 6.32e−03 | 9.19e−03 |
| | mo/0/1r | 7.86e−03 | 1.11e−03 | 7.94e−03 | 6.04e−03 | 1.07e−02 |
| | mo/0/3a | 7.54e−03 | 9.18e−04 | 7.56e−03 | 5.81e−03 | 9.34e−03 |
| | mo/0/3b | **7.24e−03** | **6.62e−04** | **7.07e−03** | 5.73e−03 | **8.48e−03** |
| | mo/10/0 | 7.70e−03 | 1.42e−03 | 7.37e−03 | 5.84e−03 | 1.27e−02 |
| | mo/10/1a | 8.01e−03 | 1.01e−03 | 8.05e−03 | 5.57e−03 | 9.79e−03 |
| | mo/10/1b | 9.31e−03 | 1.98e−03 | 8.86e−03 | 6.44e−03 | 1.43e−02 |
| | mo/10/1c | 8.63e−03 | 1.41e−03 | 8.38e−03 | 6.33e−03 | 1.20e−02 |
| | mo/10/1r | 8.41e−03 | 1.49e−03 | 8.53e−03 | **5.44e−03** | 1.07e−02 |
| | mo/10/3a | 7.99e−03 | 1.00e−03 | 8.10e−03 | 6.54e−03 | 9.72e−03 |
| | mo/10/3b | 7.86e−03 | 8.36e−04 | 7.80e−03 | 5.70e−03 | 9.82e−03 |
| | Adam | 1.29e−02 | 4.02e−03 | 1.22e−02 | 6.45e−03 | 2.45e−02 |
| 80% – 20% | so/0 | 8.06e−03 | 2.89e−03 | **5.84e−03** | **5.55e−03** | 1.15e−02 |
| | so/10 | **6.77e−03** | **5.99e−04** | 6.75e−03 | 5.70e−03 | **8.31e−03** |
| | mo/0/0 | 6.86e−03 | 5.23e−04 | 6.82e−03 | 5.96e−03 | 8.11e−03 |
| | mo/0/1a | 7.01e−03 | **5.07e−04** | 7.04e−03 | 5.99e−03 | 8.14e−03 |
| | mo/0/1b | 7.39e−03 | 6.67e−04 | 7.47e−03 | 6.15e−03 | 8.63e−03 |
| | mo/0/1c | 6.93e−03 | 6.12e−04 | 6.92e−03 | 5.94e−03 | 8.02e−03 |
| | mo/0/1r | 7.28e−03 | 7.25e−04 | 7.22e−03 | 5.94e−03 | 8.96e−03 |
| | mo/0/3a | 6.69e−03 | 6.29e−04 | **6.59e−03** | **5.37e−03** | 8.09e−03 |
| | mo/0/3b | **6.68e−03** | 5.12e−04 | 6.73e−03 | 5.87e−03 | **7.82e−03** |
| | mo/10/0 | 7.05e−03 | 8.06e−04 | 6.71e−03 | 6.07e−03 | 8.86e−03 |
| | mo/10/1a | 7.16e−03 | 6.87e−04 | 7.08e−03 | 6.00e−03 | 8.57e−03 |
| | mo/10/1b | 7.62e−03 | 9.98e−04 | 7.38e−03 | 6.46e−03 | 1.02e−02 |
| | mo/10/1c | 7.98e−03 | 1.04e−03 | 7.82e−03 | 6.23e−03 | 1.06e−02 |
| | mo/10/1r | 7.56e−03 | 8.67e−04 | 7.44e−03 | 6.27e−03 | 1.01e−02 |
| | mo/10/3a | 7.43e−03 | 8.94e−04 | 7.23e−03 | 6.07e−03 | 1.04e−02 |
| | mo/10/3b | 7.20e−03 | 6.95e−04 | 7.07e−03 | 6.04e−03 | 9.28e−03 |
| | Adam | 8.74e−03 | 2.71e−03 | 8.35e−03 | 5.88e−03 | 1.75e−02 |

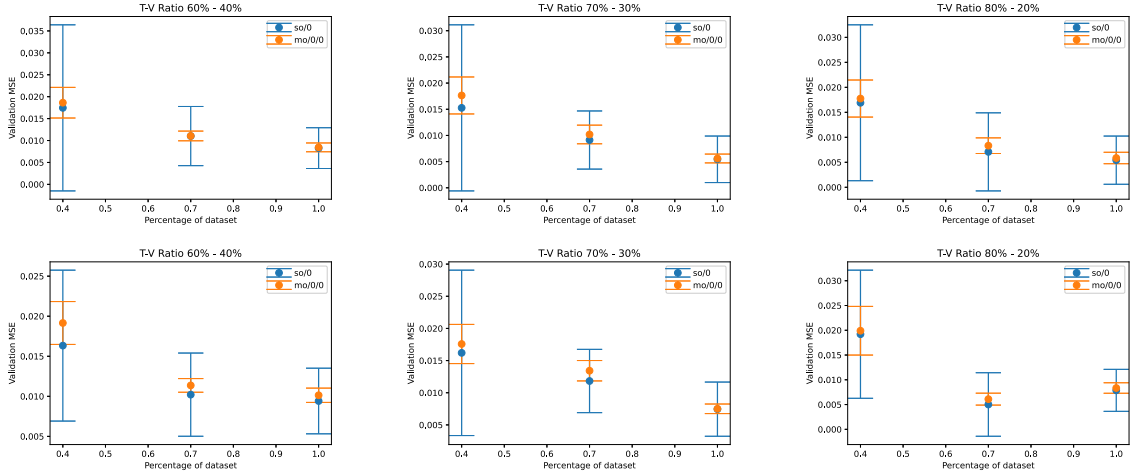


**Fig. 8.** Sensitivity analysis on the dataset size for the Real Estate dataset without noise (upper row) and with noise (lower row).

**Table 9**
Wilcoxon rank-sum tests (row algorithm vs column algorithm) for the Real Estate dataset with noise.

| | so/0 | so/10 | mo/0/0 | mo/0/1a | mo/0/1b | mo/0/1c | mo/0/1r | mo/0/3a | mo/0/3b | mo/10/0 | mo/10/1a | mo/10/1b | mo/10/1c | mo/10/1r | mo/10/3a | mo/10/3b | Adam |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T–V Ratio 60% – 40% | | | | | | | | | | | | | | | | | |
| so/0 | | = | = | = | + | = | = | = | = | = | = | = | = | = | = | + | + |
| so/10 | = | | = | = | = | = | = | = | = | = | + | + | = | + | = | + | + |
| mo/0/0 | = | = | | = | + | = | = | = | = | = | + | + | + | + | = | + | + |
| mo/0/1a | = | = | = | | = | = | = | = | = | = | + | + | + | + | = | + | + |
| mo/0/1b | − | = | − | = | | − | = | = | = | = | = | + | = | + | = | = | + |
| mo/0/1c | = | = | = | = | + | | = | = | = | + | + | + | + | + | = | + | + |
| mo/0/1r | = | = | = | = | = | = | | = | = | = | + | + | = | + | = | = | + |
| mo/0/3a | = | = | = | = | = | = | = | | = | = | + | + | + | + | = | + | + |
| mo/0/3b | = | = | = | = | = | = | = | = | | = | + | + | + | + | = | + | + |
| mo/10/0 | = | = | = | = | = | − | = | = | = | | = | = | = | = | = | = | + |
| mo/10/1a | = | − | − | − | = | − | − | − | − | = | | = | = | = | = | = | + |
| mo/10/1b | = | − | − | − | − | − | − | − | − | = | = | | = | = | − | = | + |
| mo/10/1c | = | = | − | − | = | − | = | − | − | = | = | = | | = | = | = | + |
| mo/10/1r | = | − | − | − | − | − | − | − | − | = | = | = | = | | − | = | + |
| mo/10/3a | = | = | = | = | = | = | = | = | = | = | = | + | = | + | | = | + |
| mo/10/3b | − | − | − | − | = | − | = | − | − | = | = | = | = | = | = | | + |
| Adam | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 70% – 30% | | | | | | | | | | | | | | | | | |
| so/0 | | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = |
| so/10 | = | | = | = | = | = | = | = | = | = | + | + | + | + | = | + | + |
| mo/0/0 | = | = | | = | = | = | = | = | = | = | + | + | + | + | = | = | + |
| mo/0/1a | = | = | = | | = | = | = | = | − | = | = | + | + | = | = | = | + |
| mo/0/1b | = | = | = | = | | = | = | = | − | = | = | + | + | = | = | = | + |
| mo/0/1c | = | = | = | = | = | | = | = | = | = | + | + | + | + | = | + | + |
| mo/0/1r | = | = | = | = | = | = | | = | = | = | = | + | = | = | = | = | + |
| mo/0/3a | = | = | = | = | = | = | = | | = | = | + | + | + | + | = | = | + |
| mo/0/3b | = | = | = | + | + | = | = | = | | = | + | + | + | + | + | + | + |
| mo/10/0 | = | = | = | = | = | = | = | = | = | | = | + | + | + | = | = | + |
| mo/10/1a | = | − | − | = | = | − | = | − | − | = | | + | = | = | = | = | + |
| mo/10/1b | = | − | − | − | − | − | − | − | − | − | − | | = | = | − | − | + |
| mo/10/1c | = | − | − | − | − | − | = | − | − | − | = | = | | = | = | − | + |
| mo/10/1r | = | − | − | = | = | − | = | − | − | − | = | = | = | | = | = | + |
| mo/10/3a | = | = | = | = | = | = | = | = | − | = | = | + | = | = | | = | + |
| mo/10/3b | = | − | = | = | = | − | = | = | − | = | = | + | + | = | = | | + |
| Adam | = | − | − | − | − | − | − | − | − | − | − | − | − | − | − | − | |
| T–V Ratio 80% – 20% | | | | | | | | | | | | | | | | | |
| so/0 | | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = |
| so/10 | = | | = | = | + | = | + | = | = | = | + | + | + | + | + | + | + |
| mo/0/0 | = | = | | = | + | = | + | = | = | = | = | + | + | + | + | = | + |
| mo/0/1a | = | = | = | | + | = | = | − | − | = | = | + | + | + | = | = | + |
| mo/0/1b | = | − | − | − | | − | = | − | − | − | = | = | + | = | = | = | = |
| mo/0/1c | = | = | = | = | + | | = | = | = | = | = | + | + | + | + | = | + |
| mo/0/1r | = | − | − | = | = | = | | − | − | = | = | = | + | = | = | = | + |
| mo/0/3a | = | = | = | + | + | = | + | | = | = | + | + | + | + | + | + | + |
| mo/0/3b | = | = | = | + | + | = | + | = | | = | + | + | + | + | + | + | + |
| mo/10/0 | = | = | = | = | + | = | = | = | = | | = | + | + | + | = | = | + |
| mo/10/1a | = | − | = | = | = | = | = | − | − | = | | = | + | = | = | = | + |
| mo/10/1b | = | − | − | − | = | − | = | − | − | − | = | | = | = | = | = | = |
| mo/10/1c | = | − | − | − | − | − | − | − | − | − | − | = | | = | − | − | = |
| mo/10/1r | = | − | − | − | = | − | = | − | − | − | = | = | = | | = | = | = |
| mo/10/3a | = | − | − | = | = | − | = | − | − | = | = | = | + | = | | = | = |
| mo/10/3b | = | − | = | = | = | = | = | − | − | = | = | = | + | = | = | | + |
| Adam | = | − | − | − | = | − | − | − | − | − | − | = | = | = | = | − | |

**Table 10**
Wilcoxon rank-sum tests of so/0 vs mo/0/0 under varying dataset size for the Mexican Hat problem.

| T–V Ratio | Without noise | | | With noise | | |
|---|---|---|---|---|---|---|
| | Percentage of dataset | | | Percentage of dataset | | |
| | 40% | 70% | 100% | 40% | 70% | 100% |
| 60% – 40% | = | + | + | = | + | + |
| 70% – 30% | + | + | + | + | + | + |
| 80% – 20% | = | + | + | = | + | + |

**Table 11**
Wilcoxon rank-sum tests of so/0 vs mo/0/0 under varying dataset size for the Real Estate dataset.

| T–V Ratio | Without noise | | | With noise | | |
|---|---|---|---|---|---|---|
| | Percentage of dataset | | | Percentage of dataset | | |
| | 40% | 70% | 100% | 40% | 70% | 100% |
| 60% – 40% | = | = | = | + | = | = |
| 70% – 30% | = | = | = | = | = | = |
| 80% – 20% | = | = | = | = | = | = |

of mini-batches was restricted to 2, 5, and 10, in order to retain satisfactory size. Fig. 10 reveals the increased robustness of the mo/0/0 approach compared to so/0, as well as their marginal performance differences as the number of mini-batches increased.

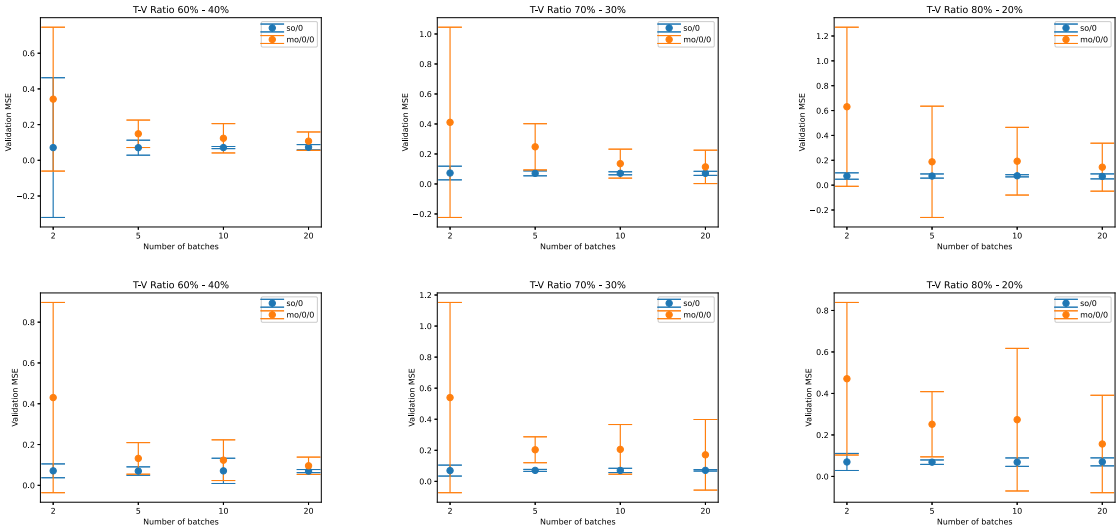


**Fig. 9.** Sensitivity analysis on the number of mini-batches per set for the Mexican Hat problem without noise (upper row) and with noise (lower row).

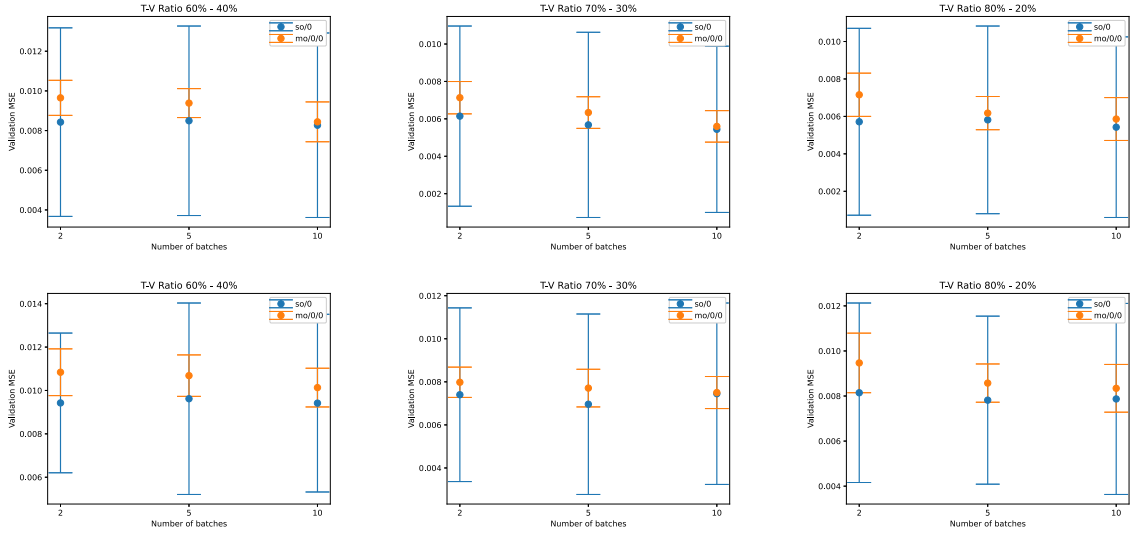


**Fig. 10.** Sensitivity analysis on the number of mini-batches per set for the Real Estate dataset without noise (upper row) and with noise (lower row).

**Table 12**
Wilcoxon rank-sum tests of so/0 vs mo/0/0 under varying number of mini-batches per set for the Mexican Hat problem.

| T–V Ratio | Without noise | | | | With noise | | | |
|---|---|---|---|---|---|---|---|---|
| | Mini-batches | | | | Mini-batches | | | |
| | 2 | 5 | 10 | 20 | 2 | 5 | 10 | 20 |
| 60% – 40% | + | + | + | + | + | + | + | + |
| 70% – 30% | + | + | + | + | + | + | + | + |
| 80% – 20% | + | + | + | + | + | + | + | + |

**Table 13**
Wilcoxon rank-sum tests of so/0 vs mo/0/0 under varying number of mini-batches per set for the Real Estate dataset.

| T–V Ratio | Without noise | | | With noise | | |
|---|---|---|---|---|---|---|
| | Mini-batches | | | Mini-batches | | |
| | 2 | 5 | 10 | 2 | 5 | 10 |
| 60% – 40% | = | = | = | + | = | = |
| 70% – 30% | = | = | = | = | = | = |
| 80% – 20% | = | = | = | = | = | = |

The multi-objective variant exhibits in all cases a slightly worse MSE median than so/0. Nevertheless, the statistical comparisons reported in Table 13 reveal their median MSE equivalence in almost all circumstances, with the exception of the 2-batches case of the noisy Real Estate dataset where so/0 achieved better median than mo/0/0.

Overall, when the amount of available data is large, the single-objective so/0 variant performs better. However, in small datasets, the mo/0/0 exhibits more robust results.

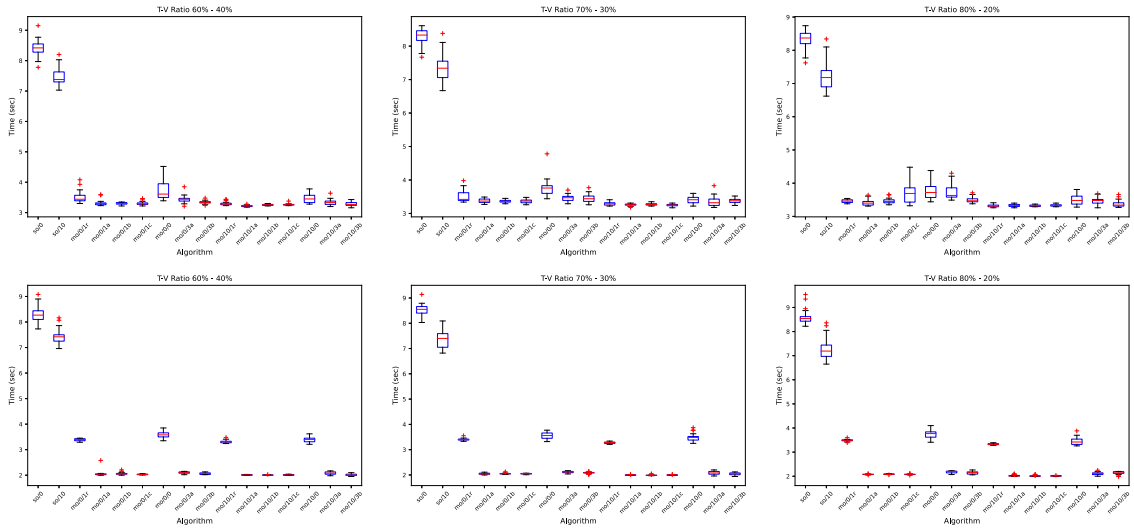


**Fig. 11.** Time requirements for the Mexican Hat problem without noise (1st row) and with noise (2nd row).

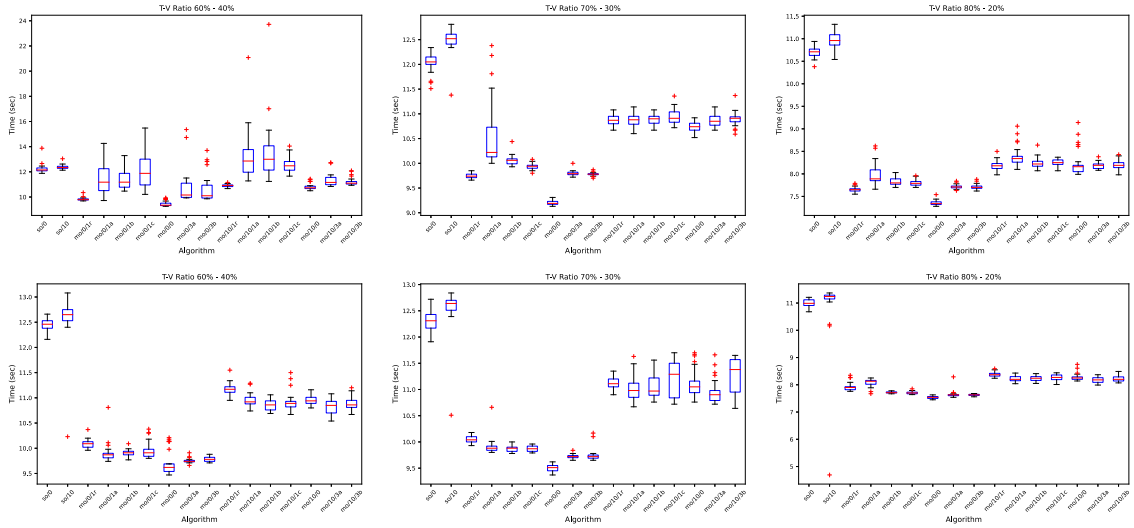


**Fig. 12.** Time requirements for the Real Estate problem without noise (1st row) and with noise (2nd row).

### *3.8. Time requirements*

The proposed methods were shown to achieve reasonably fast training. Figs. 11 and 12 illustrate the required running time (in seconds) for the studied variants in our main experiments analyzed in Sections 3.3–3.6. Interestingly, the multi-objective approaches offer faster training than the single-objective variants that employ gradients and line search. This observation underlines the usefulness of alternative training methods than the classical gradient-based approaches (such as SGD). Although running time cannot be perceived as a reliable performance measure as it is affected by external factors, it can add to our insight regarding the inherent properties and quality of the studied training algorithms.

## 4. Conclusions

VCB is a neural network training method that ameliorates the negative effect of diminishing step size in stochastic gradient-based training. It is based on the concurrent minimization of the average MSE of the network along with its variance over random sets of mini-batches. The current work extended our understanding of the standard VCB, which exploits a single-objective optimization model that can be solved also with enhanced quasi-Newton solvers. Moreover, the problem was redefined as a multi-objective problem, and it was tackled through the established MOPSO algorithm.

The two approaches were applied and compared on two problems that embrace small and large dataset, with and without noise. Each variant assumed two alternatives regarding the number of VCB cycles, namely a fixed number of 10 cycles and an alternative without a prespecified value. Also, for the multi-objective variants, 7 strategies for evaluating the Pareto set at each cycle, were considered. All variants were statistically compared among them as well as against the established Adam approach, using typical statistical analysis and significance testing. Moreover, the sensitivity of the proposed approaches on relevant parameters, namely the dataset size and the number of mini-batches per set, was further analyzed.

Overall, the studied VCB variants exhibited promising performance against the Adam method. They also revealed performance consistency for the multi-objective variants and problem

dependency for the single-objective variants, which were also more susceptible to unfavorable performance fluctuations in the presence of noise.

Evidently, the requirements of the VCB parameterization offer a rich ground for further research in order to fully understand the individual effect of each parameter, as well as its interplay with the employed optimizers and the dataset.

## CRediT authorship contribution statement

**Dimitra G. Triantali:** Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft, Writing – review & editing. **Konstantinos E. Parsopoulos:** Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft, Writing – review & editing. **Isaac E. Lagaris:** Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft, Writing – review & editing.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Dimitra G. Triantali reports financial support was provided by University of Ioannina.

## Data availability

Data will be made available on request.

## Acknowledgment

This research was supported by the project "Dioni: Computing Infrastructure for Big-Data Processing and Analysis" (MIS No. 5047222) co-funded by European Union (ERDF) and Greece through Operational Program "Competitiveness, Entrepreneurship and Innovation", NSRF 2014–2020.

## References

[1] T. Elsken, J.H. Metzen, F. Hutter, Neural architecture search: A survey, 2018, arXiv.
[2] X. Wang, J. Lin, J. Zhao, X. Yang, J. Yan, EAutoDet: Efficient architecture search for object detection, in: S. Avidan, G. Brostow, M. Cissé, G.M. Farinella, T. Hassner (Eds.), Computer Vision – ECCV 2022, Springer Nature Switzerland, Cham, 2022, pp. 668–684.
[3] E. Real, A. Aggarwal, Y. Huang, Q.V. Le, Regularized evolution for image classifier architecture search, 2018, CoRR abs/1802.01548.
[4] J. Vanschoren, Meta-learning: A survey, 2018, CoRR abs/1810.03548.
[5] Y. Li, Y. Shen, H. Jiang, W. Zhang, Z. Yang, C. Zhang, B. Cui, TransBO: Hyperparameter optimization via two-phase transfer learning, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, ACM, 2022, pp. 956–966.
[6] J. Behler, Perspective: Machine learning potentials for atomistic simulations, J. Chem. Phys. 145 (17) (2016) 170901:1–9.
[7] N. Artrith, B. Hiller, J. Behler, Neural network potentials for metals and oxides - first applications to copper clusters at zinc oxide, Phys. Status Solidi b 250 (6) (2013) 1191–1203.
[8] E.D. Cubuk, B.D. Malone, B. Onat, A. Waterland, E. Kaxiras, Representations in neural network based empirical potentials, J. Chem. Phys. 147 (2) (2017) 024104:1–5.
[9] J. Behler, M. Parrinello, Generalized neural-network representation of high-dimensional potential-energy surfaces, Phys. Rev. Lett. 98 (2007) 146401:1–4.
[10] P. Netrapalli, Stochastic gradient descent and its variants in machine learning, J. Indian Inst. Sci. 99 (2019) 201–213.
[11] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (3) (1951) 400–407.
[12] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the 3-Rd International Conference for Learning Representations, 2015.
[13] J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, J. Mach. Learn. Res. 12 (2011) 2121–2159.
[14] S.J. Reddi, S. Kale, S. Kumar, On the convergence of adam and beyond, in: Proceedings of the 2018 International Conference on Learning Representations, ICLR'18, 2018.
[15] L. Bottou, F. Curtis, J. Nocedal, Optimization methods for large-scale machine learning, SIAM Rev. 60 (2) (2018) 223–311.
[16] M. Li, T. Zhang, Y. Chen, A.J. Smola, Efficient mini-batch training for stochastic optimization, in: Proceedings of the 20-Th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'14, ACM, NY, USA, 2014, pp. 661–670.
[17] S. De, A. Yadav, D. Jacobs, T. Goldstein, Automated Inference with Adaptive Batches, in: A. Singh, J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, vol. 54, PMLR, 2017, pp. 1504–1513.
[18] A.G. Baydin, R. Cornish, D. Martínez-Rubio, M. Schmidt, F.D. Wood, Online learning rate adaptation with hypergradient descent, 2017, CoRR abs/1703.04782.
[19] P.L. Lagari, L.H. Tsoukalas, I.E. Lagaris, Variance counterbalancing for stochastic large-scale learning, Int. J. Artif. Intell. Tools 29 (5) (2020) 2050010:1–10.
[20] C.A.C. Coello, G.T. Pulido, M.S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 256–279.
[21] R. Fletcher, Practical Methods of Optimization, second ed., John Wiley & Sons, New York, 1987.
[22] C.A.C. Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, second ed., Springer, 2014.
[23] K. Deb, S. Agrawal, A. Pratab, A fast and elitist multiobjective genetic algorithm: NSGA-II, in: International Conference on Parallel Problem Solving from Nature, Springer, 2002, pp. 849–858.
[24] M. Clerc, J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. 6 (1) (2002) 58–73.
[25] K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through particle swarm optimization, Nat. Comput. 1 (2002) 235–306.
[26] K.E. Parsopoulos, M.N. Vrahatis, Particle Swarm Optimization and Intelligence: Advances and Applications, Information Science Publishing (IGI Global), 2010.
[27] D.S. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, Complex Syst. 2 (3) (1988) 321–355.
[28] J. Park, I.W. Sandberg, Universal approximation using radial-basis-function networks, Neural Comput. 3 (2) (1991) 246–257.
[29] J. Nocedal, S.J. Wright, Numerical Optimization, Springer, 2006.
[30] Real estate price prediction, 2018, URL https://www.kaggle.com/datasets/quantbruce/real-estate-price-prediction.