Computational Optimization and Applications (2026) 95:1–28
https://doi.org/10.1007/s10589-026-00813-0

Objective-function free multi-objective optimization: rate of convergence and performance of an Adagrad-like algorithm

Marianna De Santis[1,6] · **Gabriele Eichfelder**[2] · **Margherita Porcelli**[3,4,5,6]

Received: 5 February 2026 / Accepted: 14 July 2026 / Published online: 1 August 2026

Abstract

We propose an Adagrad-like algorithm for multi-objective unconstrained optimization that relies on the computation of a common descent direction only. Unlike classical local algorithms for multi-objective optimization, our approach does not rely on the dominance property to accept new iterates, which allows for a flexible and function-free optimization framework. New points are obtained using an adaptive stepsize that does not require neither knowledge of Lipschitz constants nor the use of line search procedures. The rate of convergence is analyzed and is shown to be $\mathcal{O}(1/\sqrt{k+1})$ with respect to the norm of the common descent direction. The method is extensively validated on a broad class of unconstrained multi-objective problems and simple multi-task learning instances, and compared against a first-order line search algorithm. Additionally, we present a preliminary study of the behavior under noisy multi-objective settings, highlighting the robustness of the method.

During the submission of this manuscript, we learned of independent work by Gonçalves, Grapiglia and Melo [18] posted on Optimization Online on January 30th, 2026, which proposes a similar Adagrad-like algorithm for multi-objective optimization. Nevertheless, the theoretical analysis of the method and the numerical experience are different.

✉ Margherita Porcelli
margherita.porcelli@unifi.it

Marianna De Santis
marianna.desantis@unifi.it

Gabriele Eichfelder
gabriele.eichfelder@tu-ilmenau.de

1 Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Firenze, Via di Santa Marta 3, 50139 Florence, Italy

2 Institute of Mathematics, Technische Universität Ilmenau, Po 10 05 65, 98684 Ilmenau, Germany

3 Dipartimento di Ingegneria Industriale, Università degli Studi di Firenze, Viale Morgagni 40/44, 50134 Firenze, Italy

4 ISTI–CNR, Pisa, Italy

5 INdAM Research Group GNCS, Rome, Italy

6 SIMAI-OPTIMA Group, Via Taurini, 19 00185 Roma, Italy

Keywords Multi-objective optimization · Objective-function-free optimization · Adagrad-like algorithm

1 Introduction

Recently, a class of algorithms for unconstrained nonconvex optimization has been proposed where the value of the (single) objective function is never computed. This class, referred to as Objective-Function-Free Optimization (OFFO) algorithms, has recently been very popular in the context of noisy problems, in particular in deep learning applications (see [11, 28], among many others), where they have shown remarkable insensitivity to the noise level. Also, in these applications evaluating the loss function often requires looking at millions of data points, making its repeated computation per iteration often prohibitive. This context is particularly suitable for the OFFO framework. OFFO contains several well-known algorithms such as Adagrad [5, 11], Adam [28], RMSprop [38], ADADELTA [41] which have been proposed and widely used in the machine learning community, emerging as state-of-the-art techniques to train neural networks. All these methods only rely on current and past gradient information to adaptively determine the step size at each iteration. The OFFO algorithms have been introduced to unify and to extend the complexity and convergence theory of many algorithms for nonconvex problems by using a trust-region framework [19, 22]. Indeed, the work [22] re-interpreted the deterministic Adagrad as a general trust-region method in infinity norm where, unlike standard methods, no objective is evaluated to measure the progress towards a minimizer and to enforce descent. These methods have been extended to several classes of problems including constrained optimization [1, 2, 23], multilevel optimization [21], and neural network pruning applications [33], including both deterministic and stochastic approaches.

In this work, we introduce the further challenge of optimizing multiple objective functions without evaluating any element function. This framework is particularly motivated by multi-task learning in neural networks, where a single model must simultaneously optimize multiple task-specific losses, see, e.g., [3, 34]. More broadly, stochastic multi-objective optimization problems—characterized by the presence of noise and multiple competing objectives—arise in several applications, including finance, energy, transportation, logistics, and supply chain management, see, for instance, the survey [24] and the references therein. [1]

The majority of local algorithms for multi-objective optimization uses the fact that if a point does not satisfy suitable necessary optimality conditions, then it can be easily improved, in the sense that a new point can be easily defined, with respect to a specific quality measure, which is usually the objective functions value. One aims at a new point which improves the value of at least one objective function (sometimes also all objective functions) and which is thus not dominated by the previous one. Most of the algorithms proposed in this respect, both for unconstrained and constrained multi-objective problems, extend the classical iterative scalar optimization algorithms, such

[1] Although noisy problems are a relevant motivation for our proposal, we consider here deterministic (noiseless) algorithms which are crucial to understand the behavior of stochastic ones, and provide some numerical experience on noisy multi-objective problems.

as the steepest descent [26], Newton [13, 17], external penalty [15] or interior point [16], just to name a few. The mentioned approaches produce sequences of points able to converge to single Pareto critical points. In particular, at every iteration, these algorithms look for a new improved point, that is, a point that dominates the previous one. Designing an algorithm that does not rely on the dominance property to accept new points is then a particular challenge in the context of multi-objective optimization.

Our framework is inspired by the OFFO work [22] by leveraging on the Adagrad-Norm algorithm [40] for deterministic non-convex optimization that relies on ℓ_2-norm trust-region subproblems. Our contribution can be summarized as follows:

1. We develop an adaptive strategy that uses a first-order common descent direction for which we provide a thorough theoretical characterization. The adaptive stepsize rule relies on the current and past common descent directions.
2. No knowledge of the Lipschitz constants neither the employment of a line search procedure (cfr [34]) is needed to define the adaptive stepsize.
3. We prove the convergence of the method with global convergence rate[2] of $\mathcal{O}(1/\sqrt{k+1})$ with respect to the norm of the common descent direction.
4. We extensively validate our method on a large class of unconstrained multi-objective problems and on simple multi-task problems, in comparison with a first-order line search algorithm.
5. We provide a preliminary study on the behavior of both our method and the line search based one on noisy multi-objective problems.

The paper is organized as follows. In Sect. 2, we review basic results on multi-objective optimization. Section 3 presents the `MO-Adagrad` algorithm and its theoretical analysis, including results on the common descent direction and a global convergence rate of $\mathcal{O}(1/\sqrt{k+1})$. In Sect. 4, we report our numerical experience. Finally, Sect. 5 concludes the paper.

2 Basics of multi-objective optimization

We consider the unconstrained multi-objective optimization problem

$$\min_{x\in\mathbb{R}^n} f(x) \tag{MOP}$$

with $f\colon \mathbb{R}^n \to \mathbb{R}^m$, and each $f_j\colon \mathbb{R}^n \to \mathbb{R}$, $j \in \{1,\dots,m\}$ being continuously differentiable. We use for any $m \in \mathbb{N}$ the set of indices $[m] := \{1,\dots,m\}$. We denote the gradient of one of the functions f_j in some point $x \in \mathbb{R}^n$ by $\nabla f_j(x)$. Moreover, we assume that the multi-objective optimization problem (MOP) has at least one weakly efficient point. Recall that a point $\bar{x} \in \mathbb{R}^n$ is called efficient for (MOP) if for any $x \in \mathbb{R}^n$ the conditions $f_j(x) \leq f_j(\bar{x})$ for all $j \in [m]$ already imply $f(x) = f(\bar{x})$, and it is called weakly efficient in case there is no $x \in \mathbb{R}^n$ with $f_j(x) < f_j(\bar{x})$ for

[2] Recall that given two sequences $\{\alpha_k\}$ and $\{\beta_k\}$ of nonnegative and positive scalars, respectively, we say that α_k is $\mathcal{O}(\beta_k)$ if there exists a finite constant $\kappa \in \mathbb{R}$ such that $\lim_{k\to\infty}(\alpha_k/\beta_k) \leq \kappa$.

all $j \in [m]$. For vectors $y^1, y^2 \in \mathbb{R}^m$ the inequalities $y^1 \leq y^2$ and $y^1 < y^2$ are understood componentwise.

In general, the task is to find efficient solutions for (MOP). Weak efficiency is a weaker notion, and we have that any efficient point for (MOP) is also weakly efficient for (MOP). Moreover, analogously to single-objective optimization, one can define local concepts, like locally weakly efficient and locally efficient points: a point $\bar{x} \in \mathbb{R}^n$ is called locally efficient for (MOP) if there exists a neighborhood $\mathcal{N}(\bar{x})$ of $\bar{x}$ such that for any $x \in \mathcal{N}(\bar{x})$ the condition $f(x) \leq f(\bar{x})$ already implies $f(x) = f(\bar{x})$, and it is called locally weakly efficient in case there exists a neighborhood $\mathcal{N}(\bar{x})$ of $\bar{x}$ such that there is no $x \in \mathcal{N}(\bar{x})$ with $f(x) < f(\bar{x})$.

Additionally, we assume, as it is often done in the literature for descent-type algorithms for multi-objective optimization, that the gradients of the functions f_j, $j \in [m]$ are Lipschitz continuous, i.e., for each $j \in [m]$ there exists a Lipschitz constant $L_j > 0$ with

$$\|\nabla f_j(x) - \nabla f_j(x')\|_2 \leq L_j \|x - x'\|_2 \text{ for all } x, x' \in \mathbb{R}^n.$$

We will make use of

$$L_{\max} := \max_{j \in [m]} L_j, \tag{1}$$

which denotes the maximum among the Lipschitz constants of the gradients ∇f_j, $j \in [m]$.

In our proofs, the function $\Phi \colon \mathbb{R}^n \to \mathbb{R}$ will play an important role, which we define by

$$\Phi(x) := \max_{j \in [m]} f_j(x). \tag{2}$$

This function is often used in the context of multi-objective trust-region methods [27]. There, for accepting a candidate point as a new iteration point one requires a sufficient decrease for this function Φ. This is a weaker criterion compared to requiring that a sufficient decrease is reached in each objective function f_j. The function Φ is bounded from below under our assumption:

Lemma 1 *Under our assumption that* (MOP) *has a weakly efficient point* $\bar{x} \in \mathbb{R}^n$ *there exists* $\Phi_{\text{low}} \in \mathbb{R}$ *such that*

$$\inf_{x \in \mathbb{R}^n} \Phi(x) \geq \Phi_{\text{low}} > -\infty.$$

Proof Define $\Phi_{\text{low}} := \min_{j \in [m]} f_j(\bar{x})$ and assume there exists a sequence $(x^k)_{k \in \mathbb{N}} \subseteq \mathbb{R}^n$ with $\lim_{k \to \infty} \Phi(x^k) = -\infty$, i.e., we have for all $j \in [m]$ that $\lim_{k \to \infty} f_j(x^k) = -\infty$. Then there exists $N \in \mathbb{N}$ such that for all $j \in [m]$ we have $f_j(x^N) < \Phi_{\text{low}}$ and thus $f_j(x^N) < f_j(\bar{x})$, in contradiction to $\bar{x}$ weakly efficient for (MOP). □

In this work, Φ will be used for deriving the convergence analysis, but will never be used in the proposed algorithm.

A necessary condition for a point $x \in \mathbb{R}^n$ for being weakly efficient for (MOP) is that x is a Pareto critical point for (MOP) as defined next, see, among others, [26]. The necessary and sufficient optimality conditions will be formulated in Lemma 6.

Definition 2 We call $x \in \mathbb{R}^n$ a Pareto critical point for (MOP) if there are scalars $\lambda_j \geq 0$, $j \in [m]$ with $\sum_{j=1}^m \lambda_j = 1$ and

$$\sum_{j=1}^{m} \lambda_j \nabla f_j(x) = 0.$$

Within numerical algorithms, the following function ω can be used for characterizing Pareto critical points. It is also called a proximity measure in the literature on constrained multi-objective optimization, e.g. in [12], then taking also the constraint functions into account.

Definition 3 We define the criticality measure $\omega\colon \mathbb{R}^n \to \mathbb{R}_+$ as the function which associates to each $x \in \mathbb{R}^n$ the optimal value of

$$\begin{aligned} \min_{\lambda \in \mathbb{R}^m} \ & \| \sum_{j=1}^{m} \lambda_j \nabla f_j(x) \|_2^2 \\ \text{s.t.} \ & \sum_{j=1}^{m} \lambda_j = 1, \quad \lambda \geq 0. \end{aligned} \tag{$\Omega(x)$}$$

The following result is easy to see:

Lemma 4 *A point $x \in \mathbb{R}^n$ is Pareto critical for (MOP) if and only if $\omega(x) = 0$.*

In the literature, one can often find the following characterization for Pareto criticality of a point $x \in \mathbb{R}^n$, see [26, Def. 2.2]:

$$\forall\, d \in \mathbb{R}^n \ \exists\, j \in [m] : \ \nabla f_j(x)^\top d \geq 0. \tag{3}$$

This condition is equivalent to that the optimal value of the optimization problem

$$\min_{\|d\|_2^2 \leq 1} \ \max_{j \in [m]} \nabla f_j(x)^\top d \tag{mgrad(x)}$$

is zero with optimal solution $d = 0$, see [26, Lemma 3]. The problem (mgrad(x)) was, for instance, used in [25, 36]. We show that this characterization is in fact equivalent to our definition above. While this is somewhat widely known, we give a formal proof below.

Lemma 5 *A point $x \in \mathbb{R}^n$ is a Pareto critical point for (MOP) if and only if (3) holds.*

Proof As we could not find a proof in the easy accessible literature of this well-known result, we give it here for completeness. A proof using Gordon's Theorem can be found in the Master's thesis [35]. First, assume that (3) holds, i.e., $d = 0$ is an optimal solution of (mgrad(x)) and thus $(0, 0)$ is an optimal solution of

$$\begin{aligned} \min_{(t,d) \in \mathbb{R}^{1+n}} \ & t \\ \text{s.t.} \ & \nabla f_j(x)^\top d - t \leq 0, \ \forall\, j \in [m], \\ & \|d\|_2^2 \leq 1. \end{aligned} \tag{mgrad'(x)}$$

For (mgrad'(x)), the Slater constraint qualification holds, and thus any optimal solution has to satisfy the KKT optimality conditions. Note that the constraint $\|d\|_2^2 \leq 1$ is not active in the optimal solution $(0, 0)$ while all the other constraints are active. Thus, there exist multipliers $\lambda_j \geq 0$, $j \in [m]$, with

$$1 + \sum_{j=1}^{m} \lambda_j (-1) = 0 \quad \text{and} \quad 0 + \sum_{j=1}^{m} \lambda_j \nabla f_j(x) = 0. \tag{4}$$

As a consequence, the point x is a Pareto critical point for (MOP). On the other hand, if x is Pareto critical for (MOP), then the conditions (4) are satisfied for some $\lambda_j \geq 0$, $j \in [m]$. Moreover, for $(0, 0)$ all constraints of the form $\nabla f_j(x)^\top d - t \leq 0$ are active, and thus the complementarity conditions are satisfied. As (mgrad'(x)) is a convex optimization problem, this is sufficient for $(0, 0)$ being an optimal solution of (mgrad'(x)). Hence, the condition (3) is satisfied. □

As a consequence, the optimal value of (mgrad(x)) is non-positive, and zero if and only if $x \in \mathbb{R}^n$ is Pareto critical for (MOP). Moreover, by Lemma 5 and, for instance, [13, Theorem 3.1], we have the following necessary and sufficient optimality condition.

Lemma 6 *If a point $x \in \mathbb{R}^n$ is locally weakly efficient for* (MOP)*, then x is Pareto critical. If the functions f_j, $j \in [m]$ are convex and $x \in \mathbb{R}^n$ is Pareto critical for* (MOP)*, then x is weakly efficient for* (MOP)*. If the functions f_j, $j \in [m]$ are even strictly convex and $x \in \mathbb{R}^n$ is Pareto critical for* (MOP)*, then x is efficient for* (MOP)*.*

Thereby the last statement was discussed for instance in [6], and it can easily be shown that under these conditions any weakly efficient point for (MOP) is also efficient for (MOP).

3 The multi-objective Adagrad algorithm

Building on ideas originally developed for single-objective objective-function-free optimization, we propose the algorithm outlined in Algorithm 1, named `MO-Adagrad`. At every iteration k, after computing a common descent direction for the objective functions, denoted by $-g_k^s \in \mathbb{R}^n$ and computed by solving ($\Omega(x)$) in $x = x^k$ (see Step 2), the update step s^k is obtained by scaling this direction with suitable weights w_k. These weights (see Step 3) accumulate the ℓ_2-norm of past descent directions in an analogous fashion as in the Adagrad-norm [40] algorithm.

In the following, we formally define our direction g_k^s and report a number of theoretical results related to it which are needed for the subsequent convergence rate analysis.

Algorithm 1 `MO-Adagrad`

1: Initialization: a starting point $x^0 \in \mathbb{R}^n$ and a constant $\varsigma \in (0, 1)$ are given. Set $k = 0$ and the initial weight scalar $w_{-1} = \sqrt{\varsigma}$.
2: Solve $(\Omega(x))$ for $x = x^k$ with optimal solution λ^k to obtain $g_k^s \in \mathbb{R}^n$ as in (6)
3: Compute the weights

$$w_k = \sqrt{(w_{k-1})^2 + \|g_k^s\|_2^2} \tag{5}$$

4: Compute the step

$$s^k = -\frac{1}{w_k} g_k^s$$

5: Define

$$x^{k+1} = x^k + s^k,$$

increment k by one and return to Step 2.

Definition 7 Let $x^k \in \mathbb{R}^n$ be any iteration point and denote with $\lambda^k \in \mathbb{R}^m$ an optimal solution of $(\Omega(x))$ for $x = x^k$. Then we define the vector $g_k^s \in \mathbb{R}^n$ by

$$g_k^s := \sum_{j=1}^{m} \lambda_j^k \nabla f_j(x^k) \tag{6}$$

and, in case of $\omega(x^k) > 0$, the negative of the normed vector of g_k^s by

$$n_k^s := -\frac{1}{\|g_k^s\|_2} g_k^s.$$

Observe that $\|g_k^s\|_2^2 = \omega(x^k)$ and thus, by Lemma 4, $g_k^s = 0$ if and only if $\omega(x^k) = 0$.

Remark 8 Note that when solving the problem $(\Omega(x))$ one searches for a vector with smallest Euclidean norm in the convex hull of the gradients of the functions f_j in x. As the norm is strictly convex, this vector, which is exactly g_k^s from the definition above, is uniquely defined.

For the following, we define a function $h\colon \mathbb{R}^n \to \mathbb{R}$ for some fixed $x \in \mathbb{R}^n$ by

$$h(d) := \left(\max_{j \in [m]} \nabla f_j(x)^\top d \right) + \frac{1}{2}\|d\|_2^2. \tag{7}$$

For the convergence analysis of our algorithm it is important to relate

$$\max_{j \in [m]} \nabla f_j(x^k)^\top (-g_k^s),$$

that is, the objective function of (mgrad(x)) evaluated at $x = x^k$ in $d = -g_k^s$, with the quantity $\omega(x^k)$. The needed result is stated in the forthcoming Lemma 10. For the proof of this Lemma 10, we need to first relate the value of the function h as defined

in (7) when evaluated at $x = x^k$ in $d = -g_k^s$ with $\omega(x^k)$. We also clarify that the obtained directions $-g_k^s$ and n_k^s are in fact descent directions in case x^k is not yet a Pareto critical point.

Lemma 9 *Let $x^k \in \mathbb{R}^n$ be any iteration point which is not a Pareto critical point for (MOP), that is, $\omega(x^k) > 0$. Then the following statements hold.*

(a) The direction n_k^s is an optimal solution of (mgrad(x)) for $x = x^k$.
(b) The direction $-g_k^s$ is the unique optimal solution of

$$\min_{d \in \mathbb{R}^n} \; h(d) \tag{pgrad(x)}$$

for $x = x^k$, and for the optimal value $h(-g_k^s)$ we have

$$\omega(x^k) = \|g_k^s\|_2^2 = -2h(-g_k^s), \tag{8}$$

with h as defined in (7).
(c) The direction n_k^s, and thus also $-g_k^s$, is a common descent direction of f in x^k, i.e., there exists $t_0 > 0$ such that

$$\forall j \in [m], \; \forall t' \in (0, t_0] : \; f_j(x^k + t' n_k^s) < f_j(x^k).$$

Proof We start with proving (a). To this end, we set

$$t^k := \max_{j \in [m]} \nabla f_j(x^k)^\top n_k^s,$$

and we show that (t^k, n_k^s) is feasible for the optimization problem (mgrad'(x)) for $x = x^k$ and satisfies the KKT optimality conditions for this problem with Lagrange multipliers λ_j^k, $j \in [m]$ and $\xi^k := \|g_k^s\|_2/2 \geq 0$. As (mgrad'(x)) is a convex optimization problem, this is sufficient for (t^k, n_k^s) being optimal for (mgrad'(x)) and thus for n_k^s being optimal for (mgrad(x)) for $x = x^k$.

First, note that $\|n_k^s\|_2^2 = 1$. The Lagrange function reads as

$$L(t, d, \lambda^k, \xi^k) = t + \sum_{j=1}^m \lambda_j^k (\nabla f_j(x^k)^\top d - t) + \xi^k (d^\top d - 1).$$

As λ^k is feasible for ($\Omega(x)$) for $x = x^k$ we obtain that the partial derivative of L as defined above w.r.t. t equals zero. For the partial derivatives w.r.t. the components of d we obtain the condition

$$\sum_{j=1}^m \lambda_j^k \nabla f_j(x^k) + 2\xi^k d = 0,$$

which is satisfied for $d = n_k^s$ by the definition of ξ^k and of n_k^s. It remains to show that the complementarity conditions are fulfilled, i.e., that the Lagrange-multipliers to the inactive constraints are zero. As $\|n_k^s\|_2^2 = 1$ it remains to show that for all $j \in [m]$ we have

$$\lambda_j^k(\nabla f_j(x^k)^\top n_k^s - t^k) = 0. \tag{9}$$

Let $J^k := \{j \in [m] \mid t^k = \nabla f_j(x^k)^\top n_k^s\}$. Note that $J^k \neq \emptyset$. Now assume there is an index $\ell \in [m] \setminus J^k$ with $\lambda_\ell^k > 0$ and let $\ell' \in J^k$. Then this implies $\nabla f_\ell(x^k)^\top n_k^s < \nabla f_{\ell'}(x^k)^\top n_k^s = t^k$. By the definition of n_k^s we get

$$\gamma := (\nabla f_{\ell'}(x^k) - \nabla f_\ell(x^k))^\top g_k^s < 0.$$

Next we define for any $\varepsilon \in (0, \lambda_\ell^k)$ the vector $\tilde{\lambda} \geq 0$ by $\tilde{\lambda}_j := \lambda_j^k$ for all $j \in [m]\setminus\{\ell, \ell'\}$ and

$$\tilde{\lambda}_\ell := \lambda_\ell^k - \varepsilon, \qquad \tilde{\lambda}_{\ell'} := \lambda_{\ell'}^k + \varepsilon.$$

Then $\tilde{\lambda}$ is feasible for $(\Omega(x))$ for $x = x^k$ as λ^k is feasible for $(\Omega(x))$ for $x = x^k$. We get for the objective function value of the problem $(\Omega(x))$ for $x = x^k$ that

$$\begin{aligned}
\|\sum_{j=1}^m \tilde{\lambda}_j \nabla f_j(x^k)\|_2^2 &= \|(\sum_{j=1}^m \lambda_j^k \nabla f_j(x^k)) + \varepsilon \nabla f_{\ell'}(x^k) - \varepsilon \nabla f_\ell(x^k)\|_2^2 \\
&= \|g_k^s + \varepsilon(\nabla f_{\ell'}(x^k) - \nabla f_\ell(x^k))\|_2^2 \\
&= \|g_k^s\|_2^2 + \varepsilon^2 \|\nabla f_{\ell'}(x^k) - \nabla f_\ell(x^k)\|_2^2 + 2\varepsilon(\nabla f_{\ell'}(x^k) \\
&\quad -\nabla f_\ell(x^k))^\top g_k^s \\
&= \|g_k^s\|_2^2 + \varepsilon(\varepsilon\|\nabla f_{\ell'}(x^k) - \nabla f_\ell(x^k)\|_2^2 + 2\gamma).
\end{aligned}$$

Recall that $\gamma < 0$. In case $\|\nabla f_{\ell'}(x^k) - \nabla f_\ell(x^k)\|_2^2 = 0$ this is a contradiction to $\|g_k^s\|_2^2$ being the optimal value of $(\Omega(x))$ for $x = x^k$. Otherwise, for ε small enough, we also get a contradiction to $\|g_k^s\|_2^2$ being the optimal value of $(\Omega(x))$ for $x = x^k$. Thus, the assertion is proven.

For proving part (b), instead of (pgrad(x)) we examine the following optimization problem

$$\begin{aligned}
\min_{(t,d)\in\mathbb{R}^{1+n}} \quad & t + \tfrac{1}{2}\|d\|_2^2 \\
\text{s.t. } & \nabla f_j(x)^\top d - t \leq 0, \ \forall\ j \in [m]
\end{aligned} \tag{pgrad'(x))}$$

for $x = x^k$. As this problem has a strictly convex objective function, there is not more than one optimal solution $(\bar{t}^k, \bar{d}^k)$ and we aim at showing that this is the point $\bar{t}^k := \max_{j\in[m]} f_j(x^k)^\top(-g_k^s)$ and $\bar{d}^k := -g_k^s$, which is obviously feasible. We do this by showing that the KKT conditions are satisfied, as those are sufficient for this convex optimization problem. It is easy to verify that in $(\bar{t}^k, \bar{d}^k)$ with the Lagrange-multipliers $\lambda^k \geq 0$ the derivative of the Lagrange function w.r.t. t and x vanishes. To see this, recall that $\sum_{j=1}^m \lambda_j^k = 1$ as λ^k is an optimal solution of $(\Omega(x))$ for $x = x^k$. Next, note

that the condition (9) can be written as follows, after multiplying the equation with $\|g_k^s\|_2$,

$$\lambda_j^k \left(\nabla f_j(x^k)^\top (-g_k^s) - \bar{t}^k \right) = 0$$

for all $j \in [m]$. With the same argumentation as in the proof of (a) we can argue that this condition has to be fulfilled, which is the complementarity condition of (pgrad'(x)) for $x = x^k$ in the point $(\bar{t}^k, \bar{d}^k)$. Thus, $(\bar{t}^k, \bar{d}^k)$ is a KKT point of (pgrad'(x)) and hence the unique optimal solution, for $x = x^k$. As a consequence, $\bar{d}^k = -g_k^s$ is the unique optimal solution of (pgrad(x)) for $x = x^k$.

It remains to show that (8) holds, and we do this by showing that the dual problem of (pgrad'(x)) is (next to some scalar multiplications and sign changes) the problem ($\Omega(x)$). In [34] it was already claimed that these problems are dual to each other, but the factor -0.5 has to be taken into account, see below. The formulation of the dual problem of (pgrad'(x)), including this factor, can be found in [7] on pages 413-414, and the calculation of $\bar{d}(\lambda)$ below can also be found in [14]. We give a rigorous proof here for completeness.

We always fix, as before, $x = x^k$. For formulating the objective function of the dual problem to (pgrad'(x)) we need to minimize the Langrange-function w.r.t. $t \in \mathbb{R}$ and $d \in \mathbb{R}^n$ for fixed $\lambda \geq 0$. As the Lagrange-function to (pgrad'(x)) is defined by

$$\begin{aligned} L(t, d, \lambda) &= t + \tfrac{1}{2} d^\top d + \textstyle\sum_{j=1}^m \lambda_j (\nabla f_j(x^k)^\top d - t) \\ &= t \, (1 - \textstyle\sum_{j=1}^m \lambda_j) + \tfrac{1}{2} d^\top d + \textstyle\sum_{j=1}^m \lambda_j \nabla f_j(x^k)^\top d, \end{aligned}$$

we get that the infimum is $-\infty$ unless $\sum_{j=1}^m \lambda_j = 1$. In case of a $\lambda \geq 0$ with $\sum_{j=1}^m \lambda_j = 1$ the infimum is attained in any $\bar{t}(\lambda) \in \mathbb{R}$ and

$$\bar{d}(\lambda) := -\sum_{j=1}^m \lambda_j \nabla f_j(x^k).$$

Thus the objective function of the dual problem reads, for any $\lambda \geq 0$ with $\sum_{j=1}^m \lambda_j = 1$, as

$$\begin{aligned} q(\lambda) &:= \inf_{(t,d) \in \mathbb{R} \times \mathbb{R}^n} L(t, d, \lambda) = \frac{1}{2} \bar{d}(\lambda)^\top \bar{d}(\lambda) + \sum_{j=1}^m \lambda_j \nabla f_j(x^k)^\top \bar{d}(\lambda) \\ &= -\frac{1}{2} \bar{d}(\lambda)^\top \bar{d}(\lambda). \end{aligned}$$

Thus, the dual problem can be formulated as

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m} \; & -\tfrac{1}{2} \|\bar{d}(\lambda)\|_2^2 \\ \text{s.t.} \; & \textstyle\sum_{j=1}^m \lambda_j = 1, \; \lambda \geq 0, \end{aligned}$$

which corresponds to

$$-\tfrac{1}{2}\min_{\lambda\in\mathbb{R}^m} \|\sum_{j=1}^m \lambda_j \nabla f_j(x^k)\|_2^2$$
$$\text{s.t. } \sum_{j=1}^m \lambda_j = 1,\ \lambda \geq 0$$

with optimal value, cf. ($\Omega(x)$) for $x = x^k$, equal to $-0.5\omega(x^k) = -0.5\|g_k^s\|_2^2$. As (pgrad'(x)) is convex and the Slater constraint qualification holds, we have strong duality. The optimal value of (pgrad'(x)) is $h(-g_k^s)$, and thus the assertion is shown.

For part (c), as n_k^s is optimal for (mgrad(x)) for $x = x^k$ by (a) and as the optimal value of (mgrad(x)) for $x = x^k$ has to be negative by Lemma 5 and by the non-criticality of x^k, we have $\max_{j\in[m]} \nabla f_j(x^k)^\top n_k^s < 0$. Note that it suffices to have that for all $j \in [m]$ it holds $\nabla f_j(x^k)^\top n_k^s < 0$ to have that n_k^s is a common descent direction, cf. [36, Lemma 2.5]. □

Above, we have shown that the direction $-g_k^s$ is a common descent direction. It has been called negative multi-gradient in [34]. The problem (pgrad(x)) as well as the reformulation (pgrad'(x)) have been studied in [26] for calculating a steepest descent direction. As noted there in Lemma 1 this problem can also be used to characterize Pareto criticality: a point x is Pareto critical if and only if the optimal value of (pgrad(x)) is zero. Otherwise, the optimal value is negative. It has also been stated that there are other possibilities to determine a so-called search direction, and the authors in [26] propose to solve a variant of (mgrad(x)) w.r.t. the norm $\|\cdot\|_\infty$. With Lemma 9 we have shown that, next to scaling, the directions obtained from (pgrad(x)) and (mgrad(x)) can be considered to be the same.

Finally, we relate the negative of the objective function of (mgrad(x)) evaluated at $x = x^k$ in $d = -g_k^s$ with $\omega(x^k)$ in the following lemma.

Lemma 10 *Let $x^k \in \mathbb{R}^n$ be any iteration point. Then*

$$\|g_k^s\|_2^2 = -\max_{j\in[m]} \nabla f_j(x^k)^\top (-g_k^s) \tag{10}$$

holds.

Proof First, assume that x^k is a Pareto critical point for (MOP). Then we have $\omega(x^k) = 0$ and $g_k^s = 0$. Then (10) trivially holds. Next, we assume that x^k is not a Pareto critical point for (MOP) and thus $\omega(x^k) > 0$. Then, by Lemma 9(b), we get $\|g_k^s\|_2^2 = -2h(-g_k^s)$ where the function h is defined in (7). By a simple calculation we derive (10). □

3.1 Convergence rate analysis

Recall that by (2) we have $\Phi(x) = \max_{j\in[m]} f_j(x)$ and that $L_{\max}$ was defined in (1). We start by stating a result on the descent in each iteration w.r.t. the values of the function Φ.

Lemma 11 *Suppose that Algorithm 1 is applied to* (MOP). *Then, for all* $k \geq 0$ *it holds*

$$\Phi(x^{k+1}) \leq \Phi(x^k) - \frac{\|g_k^s\|_2^2}{w_k} + \frac{L_{\max}}{2}\|s^k\|_2^2, \tag{11}$$

and

$$\Phi(x^0) - \Phi(x^{k+1}) \geq \sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell} - \frac{L_{\max}}{2} \sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell^2}. \tag{12}$$

Proof Recall that by Lemma 10 we have $-\|g_k^s\|_2^2 = \max_{j\in[m]} \nabla f_j(x^k)^\top(-g_k^s)$. Considering that

$$x^{k+1} - x^k = s^k = -\frac{1}{w_k} g_k^s,$$

by the mean value theorem applied to each function $f_j,\ j \in [m]$, we have:

$$\begin{aligned}
\Phi(x^{k+1}) &\leq \max_{j\in[m]} \left\{ f_j(x^k) + \nabla f_j(x^k)^T s^k + \tfrac{L_j}{2}\|s^k\|_2^2 \right\} \\
&\leq \Phi(x^k) + \frac{1}{w_k} \max_{j\in[m]} \nabla f_j(x^k)^\top(-g_k^s) + \frac{L_{\max}}{2}\|s^k\|_2^2 \\
&= \Phi(x^k) - \frac{\|g_k^s\|_2^2}{w_k} + \frac{L_{\max}}{2}\|s^k\|_2^2,
\end{aligned}$$

that is (11). Summing for $\ell = 0, \ldots, k$ gives

$$\Phi(x^{k+1}) \leq \Phi(x^0) - \sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell} + \frac{L_{\max}}{2} \sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell^2},$$

that is inequality (12). □

Lemma 11 gives a typical descent result when using Adagrad-like stepsizes, see e.g. [22, 23, 33]: when the second term in the right hand-side of inequality (12) is larger than the first term, the function Φ may increase from one iteration to the subsequent one. On the other hand, one expects that asymptotically the first term (linear in the weights) gets larger than the second (quadratic in the weights) and therefore to obtain a decrease in the function Φ. This is exactly what happens when choosing weights as done in the Adagrad methods and as in our proposed algorithm: the weights are increasing during the iterations and then asymptotically the linear term becomes dominant.

Remark 12 Note that by Lemma 9(c), as $w_k > 0$, the direction s^k is a common descent direction and thus a descent direction for Φ, i.e., there exists $t_0 > 0$ such that it holds $f_j(x^k + t' s^k) < f_j(x^k)$ for all $j \in [m]$, and, as a direct implication, also

$$\Phi(x^k + t' s^k) < \Phi(x^k)$$

for all $t' \in (0, t_0]$.

Thanks to Lemma 1 we have that a value Φ_{low} exists such that

$$\Phi(x) \geq \Phi_{low}, \quad \text{for all } x \in \mathbb{R}^n.$$

We are now ready to state the announced result on the global convergence rate of Algorithm 1. For that, the average of a sequence of values $\beta^\ell \in \mathbb{R}$ is defined as

$$\underset{\ell=0,\ldots,k}{\text{average}}\, \beta^\ell = \frac{1}{k+1} \sum_{\ell=0}^{k} \beta^\ell.$$

Also, the proof uses the technical results of Lemmas 14 and 15 which are reported in Appendix 6.

Theorem 13 *Let* $\Gamma_0 := \Phi(x^0) - \Phi_{\text{low}}$. *Suppose that Algorithm 1 is applied to* (MOP). *Then the following bound holds:*

$$\underset{\ell=0,\ldots,k}{\text{average}}\ \omega(x^\ell) = \underset{\ell=0,\ldots,k}{\text{average}}\ \|g_\ell^s\|_2^2 \leq \frac{\theta}{k+1}$$

with

$$\theta = \max\left\{\varsigma, \frac{\varsigma}{2} e^{\frac{2\Gamma_0}{L_{\max}}}, \frac{2048 {L_{\max}}^4}{\varsigma}\right\}$$

and $\varsigma \in (0, 1)$, *as selected in Step 1 of Algorithm 1.*

Proof Recall that by (8) we have $\omega(x^k) = \|g_k^s\|_2^2$ and we have further $\Gamma_0 = \Phi(x^0) - \Phi_{\text{low}} \geq \Phi(x^0) - \Phi(x^{k+1})$ for all k. From (12), we derive

$$\sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell} \leq \Gamma_0 + \frac{L_{\max}}{2} \sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell^2}. \tag{13}$$

Using Lemma 15 with $c_j = \|g_j^s\|_2^2$ and the fact that $w_k = \sqrt{\varsigma + \sum_{j=0}^{k} \|g_j^s\|_2^2}$ we have

$$\sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell^2} = \sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{\varsigma + \sum_{j=0}^{\ell} \|g_j^s\|_2^2} \leq \log\left(1 + \frac{1}{\varsigma} \sum_{\ell=0}^{k} \|g_\ell^s\|_2^2\right),$$

where log denotes the natural logarithmic function. Therefore, including the above inequalities in (13) we get

$$\sum_{\ell=0}^{k} \frac{\|g_\ell^s\|_2^2}{w_\ell} \leq \Gamma_0 + \frac{L_{\max}}{2} \log\left(1 + \frac{1}{\varsigma} \sum_{\ell=0}^{k} \|g_\ell^s\|_2^2\right). \tag{14}$$

Assume now that

$$\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2 \geq \max\left\{\varsigma, \frac{\varsigma}{2} e^{\frac{2\Gamma_0}{L_{\max}}}\right\}, \tag{15}$$

which implies

$$\Gamma_0 \leq \frac{L_{\max}}{2} \log\left(\frac{2}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2\right), \tag{16}$$

as well as

$$\varsigma + \sum_{\ell=0}^{k} \|g_\ell^s\|_2^2 \leq 2\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2,$$

which leads to

$$\log\left(1 + \frac{1}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2\right) \leq \log\left(\frac{2}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2\right). \tag{17}$$

Furthermore, recalling the definition of w_k in (5), we have that

$$w_k = \sqrt{\varsigma + \sum_{\ell=0}^{k} \|g_\ell^s\|_2^2} \leq \sqrt{2\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2}. \tag{18}$$

Then, observing that $w_\ell \leq w_k$ for all $\ell \leq k$ and using inequalities (16) and (18) in (14) using (17) we obtain

$$\frac{\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2}{\sqrt{2\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2}} \leq L_{\max} \log\left(\frac{2}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2\right),$$

that is

$$\frac{\sqrt{\varsigma}}{2}\sqrt{\frac{2}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2} \leq 2L_{\max} \log\left(\sqrt{\frac{2}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2}\right). \tag{19}$$

We can now apply Lemma 14 since (19) is identical to (21) with

$$a = \frac{\sqrt{\varsigma}}{2}, \quad b = 0, \quad c = 2L_{\max}, \quad u = \sqrt{\frac{2}{\varsigma}\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2},$$

and $u \geq 1$ due to (15), and obtain the bound

$$\sum_{\ell=0}^{k} \|g_\ell^s\|_2^2 \leq \frac{\varsigma}{2} \max\left\{1, \frac{64^2 {L_{\max}}^4}{\varsigma^2}\right\}.$$

Hence taking the average gives that

$$\underset{\ell\in\{0,\ldots,k\}}{\text{average}} \|g_\ell^s\|_2^2 \leq \max\left\{\frac{\varsigma}{2}, \frac{2048 {L_{\max}}^4}{\varsigma}\right\} \frac{1}{k+1}.$$

Suppose that (15) fails, then we would have

$$\underset{\ell\in\{0,\ldots,k\}}{\text{average}} \|g_\ell^s\|_2^2 \leq \max\left\{\varsigma, \frac{\varsigma}{2}e^{\frac{2\Gamma_0}{L_{\max}}}\right\}\frac{1}{k+1}.$$

□

Theorem 13 proves that the average $\omega(x^k)$ converges to zero. Furthermore, as

$$\min_{\ell\in\{0,\ldots,k\}} \|g_\ell^s\|_2 \leq \sqrt{\underset{\ell\in\{0,\ldots,k\}}{\text{average}} \|g_\ell^s\|_2^2} \leq \sqrt{\frac{\theta}{k+1}},$$

we get that the global convergence rate of the average norm of the common descent direction g_k^s is $\mathcal{O}\left(1/\sqrt{1+k}\right)$, in accordance with the original Adagrad algorithm.

4 Numerical results

In this section, we present the numerical tests conducted to evaluate the performance of `MO-Adagrad`. We implemented `MO-Adagrad` described in Algorithm 1 in MATLAB, and we set $\varsigma = 10^{-2}$ (see e.g. [23, 33]).

For comparison, we implemented a standard descent method in which $-g_k^s$ (obtained solving ($\Omega(x)$) for $x = x^k$) serves as common descent direction and the step length is determined by using an Armijo-like rule as proposed in [26]. The method is referred to as `MO-Descent` in the forthcoming sections. We report the scheme of `MO-Descent` in Algorithm 2 for completeness. In our implementation of `MO-Descent`, we set $\beta = 0.1$.

All tests have been run on an Intel® Core™ i5-14400F processor running at 2.80GHz under Linux.

Algorithm 2 `MO-Descent`

1: Initialization: a starting point $x^0 \in \mathbb{R}^n$ and a constant $\beta \in (0, 1)$ are given. Set $k = 0$.
2: Solve ($\Omega(x)$) for $x = x^k$ with optimal solution λ^k to obtain $g_k^s \in \mathbb{R}^n$ as in (6)
3: Set $t = 1$.
4: **while** $f_j(x^k - tg_k^s) > f_j(x^k) - \beta t \nabla f_j(x^k)^T g_k^s$ for some $j \in [m]$ **do**
5: $t = t/2$
6: **end while**
7: Set $\alpha^k = t$.
8: Define

$$x^{k+1} = x^k - \alpha^k g_k^s,$$

increment k by one and return to Step 2.

To solve ($\Omega(x)$) we adopt `fmincon` within MATLAB. Note that the variable space of ($\Omega(x)$) has dimension m, meaning that computing $-g_k^s$ through ($\Omega(x)$) is computationally convenient when dealing with a small number of objective functions.

Finally, for both methods, we say that a Pareto critical point is reached when $\|g_k^s\| \leq 10^{-3}$.

In our numerical experience section, we first provide an overview on the performance of `MO-Adagrad` on bi-objective unconstrained problems built from the CUTEst collection from [20] available in MATLAB form at https://github.com/GrattonToint/S2MPJ, as well as on standard unconstrained multi-objective problems from the literature (see, e.g., [37]). We also consider noisy problem instances to assess the behavior of `MO-Adagrad` under different noise levels. Then, we report the results of preliminary numerical experiments on the use of `MO-Adagrad` for training multi-task problems. Indeed, we considered two synthetic datasets: one represents a 4-classes classification task and a binary classification task, the other one consists of 2 binary classification tasks, both for a set of points in $\mathbb{R}^2$, see, e.g., [3].

4.1 Experiments on CUTEst instances

In a first set of experiments, we constructed bi-objective unconstrained instances from the single-objective unconstrained problems in the CUTEst collection [20] by adding a squared ℓ_2-norm regularizer as a second objective. We excluded all problems with optimal value equal to 0 in order to obtain proper bi-objective instances. Overall, we built 124 instances; the names of used CUTEst problems are reported in Table 1 along with their dimension that goes up to 4999.[3] All runs were initialized from the standard starting points provided by [20]. We stopped the algorithms as soon as a Pareto critical point is reached or in case a limit of 100000 number of gradient evaluations of the form $(\nabla f_1(x), \ldots, \nabla f_m(x))$ is reached. For a fair comparison, for `MO-Descent` we counted the number of function evaluations $f(x)$ divided by the problem dimension n and added this quantity to the number of gradient evaluations.

We report in Fig. 1 the performance profiles [10] related to the number of gradient (and weighted function) evaluations. Clearly, no function evaluations are required by `MO-Adagrad`. While `MO-Descent` requires a smaller number of gradient and function evaluations in case a Pareto critical point is reached, `MO-Adagrad` turns to be more robust solving the 89% of instances (`MO-Descent` only solves the 77% of instances).

4.2 Experiments on noisy problems

In a second set of experiments, we selected the unconstrained problems used in [37] as benchmark multi-objective optimization instances. Moreover, we constructed 10 bi-objective unconstrained instances combining single-objective unconstrained problems in the CUTEst collection [20] sharing the same dimension. In this case, we considered three different starting points: the starting point for the first original problem, the starting point for the second original problem and the average of the two. For the

[3] The problem dimensions are derived from the CUTEst test set available online at https://github.com/GrattonToint/S2MPJ. Although these dimensions may occasionally differ from those in the official CUTEst repository, we opted to use the MATLAB version.

Table 1 CUTEst instances considered to build bi-objective problems by including squared ℓ_2-norm regularizer as a second objective

Problem	n	Problem	n	Problem	n	Problem	n
ALLINITU	4	EGGCRATE	2	LSC1LS	3	PENALTY2	10
ARGLINA	200	EIGENALS	6	LSC2LS	3	POWELLBSLS	2
ARGLINB	10	EIGENBLS	6	LUKSAN12LS	98	POWERSUM	10
ARWHEAD	10	ELATVIDU	2	LUKSAN17LS	100	QING	5
BARD	3	ENGVAL1	10	LUKSAN22LS	100	QUARTC	10
BDQRTIC	10	ENGVAL2	3	MANCINO	10	ROSENBR	2
BEALE	2	ERRINROS	10	MARATOSB	2	ROSSIMP1	2
BIGGS6	6	ERRINRSM	10	METHANL8LS	31	ROSSIMP2	2
BOXBODLS	2	EXPFIT	2	MEXHAT	2	ROSSIMP3	2
BRKMCC	2	EXTROSNB	10	MEYER3	3	ROSZMAN1LS	4
BROWNAL	10	FLETBV3M	10	MGH17LS	5	S308	2
BROWNDEN	4	FLETCBV2	10	MGH17SLS	5	SENSORS	5
BROYDN3DLS	5	FLETCBV3	10	MODBEALE	10	SINQUAD2	10
CHNROSNB	5	FREUROTH	4	MSQRTALS	25	SINQUAD	10
CHNRSNBM	5	GAUSSIAN	3	MSQRTBLS	25	SPMSRTLS	4999
CLUSTERLS	2	GENROSE	10	n10FOLDTRLS	4	STREG	4
COOLHANSLS	9	GROWTHLS	3	NONCVXU2	10	THURBERLS	7
CRAGGLVY	10	GULF	3	NONCVXUN	10	TOINTGOR	50
CUBE	2	HATFLDD	3	NONDIA	10	TOINTGSS	10
CURLY10	15	HATFLDE	3	OSBORNEA	5	TOINTQOR	50
CYCLIC3LS	12	HIMMELBCLS	2	OSBORNEB	11	TQUARTIC	10
CYCLOOCFLS	20	HIMMELBH	2	OSCIGRAD	10	TRIDIA	5
DANIWOODLS	2	HYDCAR6LS	29	PALMER1C	8	TRIGON2	10

Table 1 continued

Problem	*n*	Problem	*n*	Problem	*n*	Problem	*n*
DENSCHNB	2	INDEFM	10	PALMER1D	7	VARDIM	10
DENSCHNC	2	JENSMP	2	PALMER2C	8	WATSON	12
DENSCHND	3	JUDGE	2	PALMER3C	8	WAYSEA1	2
DENSCHNF	2	KSSLS	10	PALMER4C	8	WOODS	4000
DEVGLA2	5	LANCZOS1LS	6	PALMER5C	6	YATP2CLS	35
DIXON3DQ	10	LANCZOS2LS	6	PALMER6C	8	YATP2LS	35
EDENSCH	10	LANCZOS3LS	6	PALMER7C	8	YFITU	3
EG2	10	LIARWHD	10	PALMER8C	8	ZANGWIL2	2

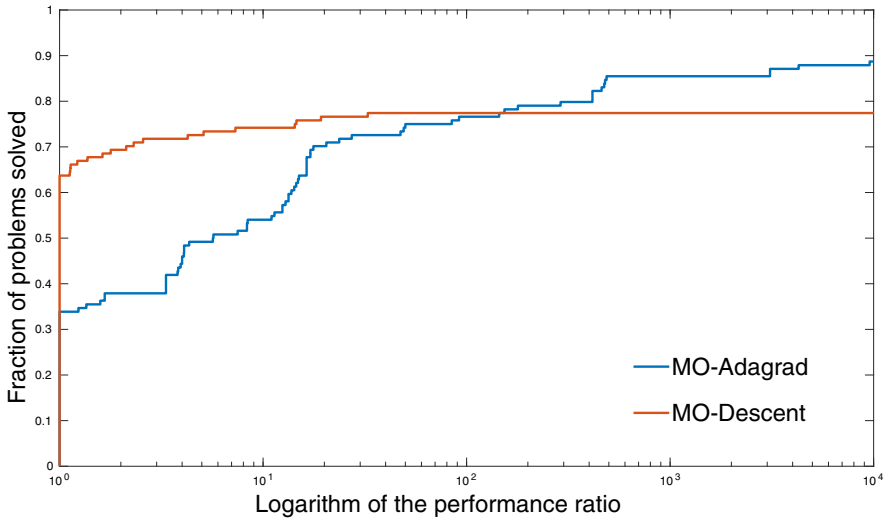

Fig. 1 Performance profiles ($\log_{10}$-scale) on the number of gradient (and function) evaluations on instances derived from CUTEst problems by including ℓ_2-norm regularizer as a second objective

benchmark problems, we used 10 randomly generated starting points. The results in the forthcoming table are reported on average.

As before, we stopped the algorithms as soon as a Pareto critical point is reached or in case the limit of 100000 number of gradient (and function in case of `MO-Descent`) evaluations is reached. As the objective-function-free methods are mainly known for their ability of being robust to noise, we evaluated our approach under different levels of noise added both to the objectives and to the gradients. Specifically, we considered a noise level of $\rho \in \{5\%, 15\%, 25\%\}$ by adding a relative Gaussian noise with unit variance (see, e.g., [23]). We perturbed each element function as follows

$$\tilde{f}_j(x) = f_j(x)(1 + \rho\mathcal{N}(0, 1)), \qquad j \in [m]$$

where $\mathcal{N}(0, 1)$ is the standard normal distribution. Gradients are perturbed componentwise, analogously.

We report in Table 2 the results obtained in terms of gradient and function evaluations. Consistent with our first set of experiments, in the noise-free setting, `MO-Adagrad` exhibits fewer failures than `MO-Descent`, the latter of which fails on 20 instances. When noise is added to both the gradients and the objective functions, `MO-Descent` shows improved performance, whereas `MO-Adagrad` remains largely unchanged[4]. In particular, `MO-Descent` considerably reduces its number of failures for all the considered values of ρ (both `MO-Descent` and `MO-Adagrad` solve the 5 benchmark problems for the 10 starting points and the different noise levels within the limit of gradient and function evaluations).

[4] We also ran experiments with $\rho = 10^{-3}$ and $\rho = 10^{-4}$. While the results are close to those of the noise-free setting, a small amount of noise allowed `MO-Descent` to recover from some failures, consistent with our observations for $\rho = 0.05$.

Table 2 Results on the instances obtained combining CUTEst problems and on the benchmark instances. For the CUTEst problems, three starting points are considered: the starting point for the first original problem, the starting point for the second original problem and the average of the starting points of the two original problems. For the benchmark instances, we use 10 different randomly generated starting points. All the results are averaged in gradient (and function) evaluations over the number of instances solved (reported in brackets)

Problem	n	no noise		$\rho = 5\%$	
		MO-Descent	MO-Adagrad	MO-Descent	MO-Adagrad
BROWNDEN-ALLINITU	4	-	**13(1)**	5484.3(1)	**14(1)**
BROWNAL-ARWHEAD	10	4.9(2)	**25.7(3)**	3237.5(3)	**866.7(3)**
BROWNAL-VARDIM	10	**657.4(3)**	11830.7(3)	**165.9(3)**	10253.3(3)
ARWHEAD-VARDIM	10	78.0(1)	**6.7(3)**	1039.4(2)	**126.3(3)**
ZANGWIL2-ROSENBR	2	-	**64.0(3)**	2729.2(3)	**219.0(3)**
ZANGWIL2-CUBE	2	56.0(1)	**94.0(3)**	700.2(3)	**99.0(3)**
ZANGWIL2-WAYSEA1	2	-	**217.7(3)**	1881.3(3)	**173.7(3)**
ROSENBR-WAYSEA1	2	-	**4887.3(3)**	22722.7(3)	**5156.0(3)**
ROSENBR-CUBE	2	1215.5(3)	**581(3)**	1107.7(3)	**772.0(3)**
WAYSEA1-CUBE	2	-	**719(2)**	9594.3(2)	**334.5(2)**
Lovison 3	2	**24.2(10)**	1685.8(10)	1995.6(10)	**1115.2(10)**
Lovison 4	2	**15.6(10)**	3422.4(10)	**270.2(10)**	3356.2(10)
MOP1	2	**3.0(10)**	2536.7(10)	**17.1(10)**	2517.1(10)
T1	2	**31.8(10)**	3844.7(10)	**837.0(10)**	1944.8(10)
T2	2	12.2(10)	**7.1(10)**	58.1(10)	**7.2(10)**
Problem	n	$\rho = 15\%$		$\rho = 25\%$	
		MO-Descent	MO-Adagrad	MO-Descent	MO-Adagrad
BROWNDEN-ALLINITU	4	**23526.9(2)**	16(1)	8693.75(1)	**19(1)**
BROWNAL-ARWHEAD	10	680.5(3)	**135.3(3)**	259.2(3)	**57.7(3)**
BROWNAL-VARDIM	10	**746.7(3)**	7576.3(3)	**338.9(3)**	4570.3(3)
ARWHEAD-VARDIM	10	663.3(3)	**48.7(3)**	421.2(3)	**25.3(3)**

Table 2 continued

Problem	n	$\rho = 15\%$		$\rho = 25\%$	
		`MO-Descent`	`MO-Adagrad`	`MO-Descent`	`MO-Adagrad`
ZANGWIL2-ROSENBR	2	6036.8(3)	**362.3(3)**	4361.3(3)	**214.0(3)**
ZANGWIL2-CUBE	2	1126.0(3)	**381.7(3)**	2302.2(3)	**432.7(3)**
ZANGWIL2-WAYSEA1	2	737.7(3)	**93.7(3)**	559.0(3)	**79.0(3)**
ROSENBR-WAYSEA1	2	53622.8(3)	**6964.0(3)**	37562.0(3)	**5334.0(3)**
ROSENBR-CUBE	2	1377.0(3)	**696.3(3)**	**861.3(3)**	2365.0(3)
WAYSEA1-CUBE	2	4939.0(2)	**123(2)**	7984.0(2)	**679.5(2)**
Lovison 3	2	9847.5(10)	**1265.9(10)**	23562.0(10)	**1506.5(10)**
Lovison 4	2	**789.0(10)**	3190.6(10)	**1778.3(10)**	2875.0(10)
MOP1	2	**4.5(10)**	2401.6(10)	**4.5(10)**	2214.6(10)
T1	2	2627.6(10)	**2464.1(10)**	**1443.5(10)**	2093.6(10)
T2	2	65.1(10)	**7.2(10)**	68.2(10)	**6.6(10)**

Bold indicates the best performing algorithm among those compared

At first glance, this suggests that the presence of noise helps `MO-Descent` in identifying Pareto critical points. To investigate this further, we examined the locations of the Pareto critical points in the criterion space. We report in Table 3 the distance between the Pareto critical points detected in the presence of noise and those obtained in the noise-free setting (on average on the runs sharing the same starting points).[5] The results show that the points found by `MO-Descent` exhibit larger distances compared to those obtained by `MO-Adagrad` for the majority of the instances. While this indicates that noise can drive exploration into different regions of the criterion space, it may also represent an undesirable behavior: `MO-Descent` may fail to reliably converge to the same solutions as in the noiseless setting, whereas `MO-Adagrad` demonstrates more robust convergence, highlighting its reliability in noisy scenarios.

4.3 Experiments on multi-task geometric classification

We consider a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i^{(1)}, y_i^{(2)})\}_{i=1}^N$ consisting of $N = 10000$ points uniformly sampled in the square $[-2, 2]^2 \subseteq \mathbb{R}^2$. In the first example, called *Quadrants-Circle*, points are labeled with respect to two different criteria: Task 1 to belong to one of the four quadrants of $\mathbb{R}^2$ (4-classes classification task); Task 2 to be inside or outside the unit circle centered in the origin (binary classification task).

Since Task 2 (Circle) is not linearly separable in the original 2D space, the input vector $\mathbf{x}_i$ is obtained via a feature engineering [4] that includes quadratic terms:

$$\mathbf{x}_i = [1, x_{i,1}, x_{i,2}, x_{i,1}^2, x_{i,2}^2]^\top \in \mathbb{R}^5$$

where the first element 1 represents the bias term, $x_{i,1}$ and $x_{i,2}$ are the components of the point $x_i \in [-2, 2]^2$, for $i = 1, \ldots, N$. The labels are defined as follows:

- $y_i^{(1)} \in \{1, 2, 3, 4\}$: Quadrant Label.
- $y_i^{(2)} \in \{0, 1\}$: Circle Label (1 if inside, 0 if outside).

As prediction models, we considered the Softmax Regression for Task 1 and the Logistic Regression for Task 2. Finally, we employed categorical Cross-Entropy and the binary Cross-Entropy as loss functions for the two tasks.

Overall, the final bi-objective problem is:

$$\min_{\mathbf{W}_1, \mathbf{w}_2} \; (J_1(\mathbf{W}_1), J_2(\mathbf{w}_2))$$

where $\mathbf{W}_1 \in \mathbb{R}^{5\times 4}$ and $\mathbf{w}_2 \in \mathbb{R}^5$ denote the weights of Task 1 and Task 2, respectively. The (non-competing) objectives are

$$J_1(\mathbf{W}_1) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{4} \mathbb{1}(y_i^{(1)} = k) \log \left(\frac{\exp(\mathbf{w}_{1,k}^\top \mathbf{x}_i)}{\sum_{j=1}^4 \exp(\mathbf{w}_{1,j}^\top \mathbf{x}_i)} \right),$$

[5] Note that, for ARWHEAD-VARDIM, `MO-Descent` with $\rho = 5\%$ identified Pareto critical points from starting points different from the one that led to a solution in the noise-free setting; consequently, the distance could not be computed.

Table 3 Distance from the Pareto critical points detected with no noise

Problem	MO-Descent	MO-Adagrad	MO-Descent	MO-Adagrad	MO-Descent	MO-Adagrad
	$\rho = 5\%$		$\rho = 15\%$		$\rho = 25\%$	
BROWNAL-ARWHEAD	4.57	**2.28**	5.13	**2.48**	7.25	**2.64**
BROWNAL-VARDIM	**0.01**	0.02	**0.01**	0.04	**0.01**	0.06
ARWHEAD-VARDIM	-	**564.17**	10340.18	**225.24**	10139.62	**1343.01**
ZANGWIL2-CUBE	176.16	**27.15**	**1.71**	81691.15	**10.25**	90851.88
ROSENBR-CUBE	0.28	**0.05**	< **1e-2**	0.13	< **1e-2**	0.33
Lovison 3	100.37	**32.58**	142.87	**113.09**	412.79	**210.07**
Lovison 4	15.41	**1.45**	27.22	**4.09**	25.21	**8.81**
MOP1	0.25	**0.11**	3.77	**0.46**	3.91	**0.76**
T1	37.56	**8.03**	59.61	**22.38**	82.10	**35.01**
T2	**0.05**	0.06	**0.16**	0.17	0.28	0.28

Bold indicates the best performing algorithm among those compared

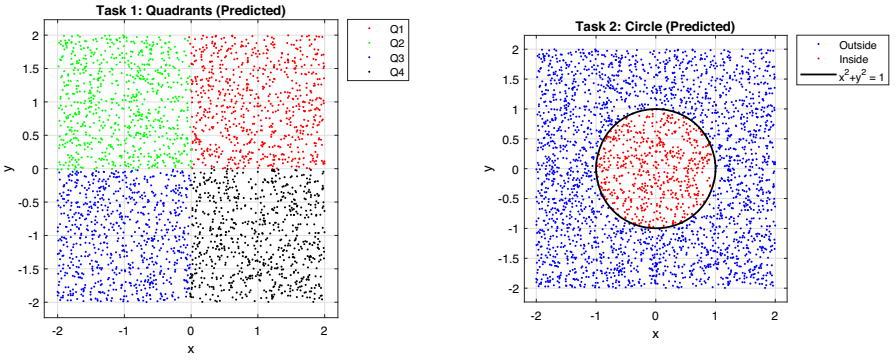

Fig. 2 *Quadrants-Circle*: Task 1 on the left, Task 2 on the right. Different colors denote different labels for the points

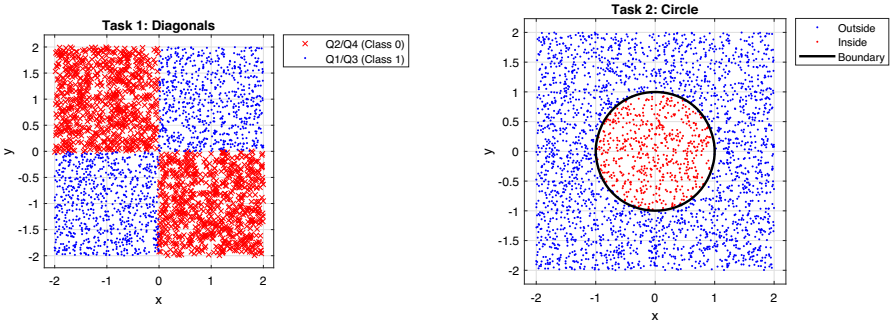

Fig. 3 *Diagonals-Circle*: Task 1 on the left, Task 2 on the right. Different colors denote different labels for the points

and

$$J_2(\mathbf{w}_2) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i^{(2)}\log(\hat{y}_i^{(2)}) + (1-y_i^{(2)})\log(1-\hat{y}_i^{(2)})\right], \tag{20}$$

where $\mathbb{1}$ denotes the indicator function, $\hat{y}_i^{(2)} = \sigma(\mathbf{w}_2^\top \mathbf{x}_i)$ with σ the sigmoid function and $\mathbf{w}_{1,j}$ denotes the jth column of $\mathbf{W}_1$, for $j = 1, \ldots, 4$.

In the second example, named *Diagonals-Circle*, Task 1 is a non-linear binary classification task based on the XOR logic of the Cartesian quadrants. The model must distinguish between points in the main diagonal (Quadrants 1 and 3) and the anti-diagonal (Quadrants 2 and 4). This requires the inclusion of an interaction feature $(x \cdot y)$ to achieve linear separability. In this case, we used the objective function J_2 in (20) for both objectives.

In both examples, the data set is randomly split into a training set and a test set made up of 8000 and 2000 samples, respectively. A representation of the predicted labels on the testing set for the two multi-task problems obtained with `MO-Adagrad` is reported in Figs. 2 and 3.

In Table 4 we report the results obtained with `MO-Descent` and `MO-Adagrad` on the two multi-task examples. In particular, we let the solvers run for 1000 iterations,

Table 4 Results on the multi-task geometric classification instances

Problem	MO-Descent				MO-Adagrad		
	g_{eval}	f_{eval}	time	$\mathcal{A}$	g_{eval}	time	$\mathcal{A}$
Quadrants - Circle	306	3957	24.9	99.8%	388	**6.9**	99.6%
Diagonals - Circle	577	6324	27.3	99.9%	970	**10.0**	99.1%

Bold indicates the best performing algorithm among those compared

and we retrieved the number of gradient/function evaluations (g_{eval} / f_{eval}) and elapsed time in seconds (time) employed in the training phase, to get the maximum accuracy $\mathcal{A}$ on the testing set. The accuracy $\mathcal{A}$ is measured considering the minimum of the testing accuracies in the two tasks. We observe that the two solvers get comparable accuracies (slightly in favor of `MO-Descent`) but `MO-Adagrad` is faster by a factor 3 than `MO-Descent`. This is due to the objective-function-free nature of `MO-Adagrad` that makes it particularly suitable when, as in this multi-task framework, evaluating the objective function is roughly as expensive as evaluating the gradient.

5 Conclusions

In this work, we propose a novel Adagrad-like algorithm, `MO-Adagrad`, for unconstrained multi-objective optimization. Our method does not need line search based techniques or dominance-based criteria for accepting new points, while still converging to Pareto critical points. Through extensive numerical tests on a broad set of bi-objective instances, we show the robustness of `MO-Adagrad` in comparison with a line search based first-order method. The method looks robust and promising to deal with noisy problems, as known for its single-objective counterpart. Further preliminary experiments have been devoted to training multi-task problems, suggesting the potential ability of `MO-Adagrad` to deal with multi-task learning problems.

Future work will be devoted to extend `MO-Adagrad` to compute sets of nondominated Pareto points, as opposed to the single Pareto point currently produced. Recent advances in multi-objective optimization have seen growing interest in local algorithms that generate sequences of sets, with the aim of approximating the Pareto set of a multi-objective problem [8, 9, 29–32]. However, avoiding the use of dominance-based acceptance criteria presents a significant challenge when generating a sequence of sets, rather than a sequence of individual points. Consequently, future work will focus on developing alternative approaches, beyond multistart procedures, to tackle this complexity.

6 Technical lemmas

We report in this section two technical results. The proof of the first one can be found in [2] and is reported for the sake of completeness. The proof of Lemma 15 can be

found in [22, Lemma 3.1]. In the following, with log we denote the natural logarithmic function.

Lemma 14 *Suppose that*

$$au \leq b + c\log(u), \tag{21}$$

for some $a, c > 0$*, a scalar* $b \in \mathbb{R}$ *and* $u \geq 1$*. Then*

$$u \leq \max\left\{e^{b/c}, \frac{4c^2}{a^2}\right\}.$$

Proof Suppose that $u \geq e^{b/c}$. Then $b \leq c\log(u)$ and using (21) we get (see [39, eq. (14)])

$$au \leq 2c\log(u) \leq 2c\log(1+u) \leq \frac{2cu}{\sqrt{1+u}}.$$

Hence $a\sqrt{1+u} \leq 2c$, which is to say that $u \leq (2c/a)^2 - 1$, yielding the desired bound. □

Lemma 15 *Let* $\{c_k\}_{k\geq 0}$ *be a non-negative sequence and* $\xi > 0$*. Then*

$$\sum_{j=0}^{k} \frac{c_j}{\xi + \sum_{\ell=0}^{j} c_\ell} \leq \log\left(\frac{\xi + \sum_{j=0}^{k} c_j}{\xi}\right).$$

Acknowledgements The authors would like to thank the anonymous referee for their detailed and constructive comments that allowed us to improve the quality of our paper.

Author Contributions All authors contributed equally to the writing of this article. All authors reviewed the manuscript.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. The work of M.P. was partially supported by INdAM-GNCS under the project CUP E53C24001950001.
The research of M.P. was partially granted by the Italian Ministry of University and Research (MUR) through the PRIN 2022 "MOLE: Manifold constrained Optimization and LEarning", code: 2022ZK5ME7 MUR D.D. financing decree n. 20428 of November 6th, 2024 (CUP B53C24006410006).
A research visit of G.E. at University of Florence with M.P. and M.d.S. was funded by the Erasmus+ Programme of the European Union.

Data Availability The data presented in this manuscript are reproducible through the implementation publicly available on https://github.com/mariannadesantis/MO-ADAGRAD.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Consent for publication Not applicable.

Consent to participate Not applicable.

Ethical approval Not applicable.

References

1. Bellavia, S., Gratton, S., Morini, B., Toint, P.L.: Fast stochastic second-order adagrad for nonconvex bound-constrained optimization (2025). arXiv:2505.06374
2. Bellavia, S., Gratton, S., Morini, B., Toint, P.L.: An objective-function-free algorithm for general smooth constrained optimization (2026). arXiv:2602.11770
3. Bellavia, S., Della Santa, F., Papini, A.: ATE-SG: alternate through the epochs stochastic gradient for multi-task neural networks. Optim. Methods Softw. 1–33 (2026)
4. Bishop, C.M.: Pattern Recognition and Machine Learning. Chapter 4: Linear Models for Classification. Springer, New York (2006)
5. Brendan McMahan, H., Streeter, M.: Adaptive bound optimization for online convex optimization (2010). arXiv:1002.4908
6. Burachik, R., Kaya, C.Y., Rizvi, M.M.: A new scalarization technique and new algorithms to generate Pareto fronts. SIAM J. Optim. **27**(2), 1010–1034 (2017)
7. Chen, J., Li, G.-X., Yang, X.-M.: Variable metric method for unconstrained multiobjective optimization problems. J. Oper. Res. Soc. China **11**, 409–438 (2022)
8. Cocchi, G., Liuzzi, G., Lucidi, S., Sciandrone, M.: On the convergence of steepest descent methods for multiobjective optimization. Comput. Optim. Appl. **77**, 1–27 (2020)
9. Custódio, A.L., Madeira, J.F.A., Vaz, A.I.F., Vicente, L.N.: Direct multisearch for multiobjective optimization. SIAM J. Optim. **21**(3), 1109–1140 (2011)
10. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
11. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**(7), 2121–2159 (2011)
12. Eichfelder, G., Warnow, L.: Proximity measures based on KKT points for constrained multi-objective optimization. J. Global Optim. **80**, 63–86 (2020)
13. Fliege, J., Drummond, L.M.G., Svaiter, B.F.S.: Newton's method for multiobjective optimization. SIAM J. Optim. **20**(2), 602–626 (2009)
14. Fliege, J., Vaz, A.I., Vicente, L.N.: Complexity of gradient descent for multiobjective optimization. Optim. Methods Softw. **34**(5), 949–959 (2019)
15. Fukuda, E.H., Drummond, L.M.G., Raupp, F.M.P.: An external penalty-type method for multicriteria. TOP **24**, 493–513 (2016)
16. Fukuda, E.H., Drummond, L.M.G., Raupp, F.M.P.: A barrier-type method for multiobjective optimization. Optimization **69**(11), 2471–2487 (2020)
17. Gonçalves, M.L.N., Lima, F.S., Prudente, L.F.: Globally convergent Newton-type methods for multiobjective optimization. Comput. Optim. Appl. **83**(2), 403–434 (2022)
18. Gonçalves, M.L.N., Grapiglia, G.N., Melo, J.G.: An adaptive line-search-free multiobjective gradient method and its iteration-complexity analysis. Optimization Online.https://optimization-online.org/?p=33379, 30/01/2026
19. Grapiglia, G.N., Stella, G.F.D.: An adaptive trust-region method without function evaluations. Comput. Optim. Appl. **82**(1), 31–60 (2022)
20. Gratton, S., Toint, P.L.: S2MPJ and CUTEst optimization problems for Matlab, Python and Julia. Optim. Methods Softw. **40**(4), 1–33 (2025)
21. Gratton, S., Kopaničáková, A., Toint, P.L.: Multilevel objective-function-free optimization with an application to neural networks training. SIAM J. Optim. **33**(4), 2772–2800 (2023)

22. Gratton, S., Jerad, S., Toint, P.L.: Complexity of a class of first-order objective-function-free optimization algorithms. Optim. Methods Softw. **41**(2), 1–31 (2024)
23. Gratton, S., Jerad, S., Toint, P.L.: Complexity of Adagrad and other first-order methods for nonconvex optimization problems with bounds constraints (2024). arXiv:2406.15793
24. Gutjahr, W.J., Pichler, A.: Stochastic multi-objective optimization: a survey on non-scalarizing methods. Ann. Oper. Res. **236**, 475–499 (2016)
25. Jerinkić, N.K., Rutešić, L., Trombini, I.: ASMOP: additional sampling stochastic trust region method for multi-objective problems (2025). arXiv:2506.10976
26. Jörg Fliege and Benar Fux Svaiter: Steepest descent methods for multicriteria optimization. Math. Methods Oper. Res. **51**(3), 479–494 (2000)
27. Kely, D.V., Villacorta, P.R., Soubeyran, O.A.: A trust-region method for unconstrained multiobjective problems with applications in satisficing processes. J. Optim. Theory Appl. **160**(3), 865–889 (2014)
28. Kingma, D.P.: Adam: a method for stochastic optimization (2014). arXiv:1412.6980
29. Lapucci, M., Mansueto, P.: Improved front steepest descent for multi-objective optimization. Oper. Res. Lett. **51**(3), 242–247 (2023)
30. Lapucci, M., Mansueto, P.: A limited memory Quasi-Newton approach for multi-objective optimization. Comput. Optim. Appl. **85**(1), 33–73 (2023)
31. Lapucci, M., Mansueto, P., Pucci, D.: Effective front-descent algorithms with convergence guarantees. SIAM J. Optim. **36**(2), 597–625 (2026)
32. Liuzzi, G., Lucidi, S., Rinaldi, F.: A derivative-free approach to constrained multiobjective nonsmooth optimization. SIAM J. Optim. **26**(4), 2744–2774 (2016)
33. Porcelli, M., Seraghiti, G., Toint, P.L.: prunAdag: an adaptive pruning-aware gradient method. Comput. Optim. Appl. **93**(1), 85–119 (2026)
34. Suyun Liu and Luís Nunes Vicente: The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. Ann. Oper. Res. **339**(3), 1119–1148 (2024)
35. Thomann, J.: Decomposed descent methods in multiobjective optimization. Master's thesis, Technische Universität Ilmenau (2015)
36. Thomann, J.: A trust region approach for multi-objective heterogeneous optimization. PhD thesis, Technische Universität Ilmenau, 2019. Technische Universität Ilmenau, Dissertation (2019)
37. Thomann, J., Eichfelder, G.: A trust-region algorithm for heterogeneous multiobjective optimization. SIAM J. Optim. **29**(2), 1017–1047 (2019)
38. Tieleman, T., Hinton, G.: Lecture 6.5-rmsprop, coursera: neural networks for machine learning, vol. 6, University of Toronto (2012) (Technical Report)
39. Topsøe, F.: Some bounds for the logarithmic function. Inequal. Theory Appl. **4**, 137 (2007)
40. Ward, R., Xiaoxia, W., Bottou, L.: Adagrad stepsizes: sharp convergence over nonconvex landscapes. J. Mach. Learn. Res. **21**(219), 1–30 (2020)
41. Zeiler, M.D.: ADADELTA: an adaptive learning rate method (2012). arXiv:1212.5701