
Tight analyses of first-order methods with error feedback

Anonymous Author(s)

Affiliation

Address

email

Abstract

Communication between agents often constitutes a major computational bottleneck in distributed learning. One of the most common mitigation strategies is to compress the information exchanged, thereby reducing communication overhead. To counteract the degradation in convergence associated with compressed communication, error feedback schemes—most notably EF and EF²¹—were introduced. In this work, we provide a *tight analysis* of both of these methods. Specifically, we find the Lyapunov function that yields the best possible convergence rate for each method—with matching lower bounds. This principled approach yields sharp performance guarantees and enables a rigorous, apples-to-apples comparison between EF, EF²¹, and Compressed Gradient Descent. Our analysis is carried out in a simplified yet representative setting, which allows for clean theoretical insights and fair comparison of the underlying mechanisms.

1 Introduction

Over the past decade, distributed optimization has become a cornerstone of large-scale machine learning. This shift is driven by major increases in the size of models and training data, as well as increasing societal concerns about data ownership and privacy. Ultimately, solutions in which training is distributed across a network of n agents, each retaining its own local data, under the coordination of a central server, have emerged as one of the most natural and efficient solutions to this problem [1, 2]. Formally, the goal is to solve the following minimization problem:

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{n} \sum_{i=1}^n f_i(x) \right\}. \quad (1)$$

Classical methods such as distributed gradient descent and its stochastic variants achieve linear speedups in iteration complexity with respect to the number of agents. However, they often suffer from significant *communication overhead*, as gradients or model updates must be exchanged frequently over bandwidth-limited channels [3–5]. As the scale of models keep increasing, this communication bottleneck has been identified early on as a critical limitation, prompting the development of methods aimed at reducing communication costs. Two main strategies are favored: scarcely communicating with the central server, known as *local iterations* [see e.g. 1, 6]—and transmitting *compressed updates*, which aim to reduce the size of the exchanged information. Compression mechanisms can be applied to reduce communication either from agents to the server [3, 7–14] or from the server to the agents [15–23]. This paper focuses on methods using compression operators, which encompass a variety of strategies, including selecting only a fraction of the weights to be transmitted (e.g., the top K coordinates [8]) or communicating low-precision updates via quantization [7].

Formally, a compression operator is a possibly random mapping $\mathcal{C} : \mathcal{X} \rightarrow \mathcal{X}$, such that $\mathcal{C}(X)$ can be encoded (almost surely or on average) with a lower number of bits than X . The most

Algorithm 1 Compressed gradient descent (CGD)

1: **initialization:** $x_0 \in \mathbb{R}^d, \eta > 0$
2: **for** $k = 0, 1, 2, \dots, N$ **do**
3: Agent $i \in [n]$ compresses $\nabla f_i(x_k)$ and communicates $m_k^{(i)} := \mathcal{C}(\nabla f_i(x_k))$
4: Server updates $x_{k+1} \leftarrow x_k - \eta \cdot \frac{1}{n} \sum_{i=1}^n m_k^{(i)}$
5: **end for**

34 natural algorithm leveraging communication compression with a centralized server is the *compressed*
35 *gradient descent* algorithm (CGD), which is described in Algorithm 1. The main idea is to perform
36 a distributed gradient step, with the compression operator \mathcal{C} applied to the gradient of each agent
37 before communication. Although this compression scheme reduces the communication cost, it comes
38 at the expense of non-convergence in any practical setting [24].

39 To assess the general impact of compression schemes on the rate of convergence, one typically
40 leverages the fact that these compressors all satisfy generic assumptions. These include unbiasedness,
41 i.e., $\mathbb{E}[\mathcal{C}(x)] = x$ for any $x \in \mathcal{X}$, together with relatively bounded variance, which states that
42 $\mathbb{E}[\|\mathcal{C}(x) - x\|^2] \leq \omega \|x\|^2$ for any $x \in \mathcal{X}$ [7, 11, 9, 25–27, 12, 28, 19, 20, 29–31], or *contractive-*
43 *ness* [32–34, 21, 24], defined as follows:

44 **Assumption 1** (Contractive compression operator). *The compression operator \mathcal{C} is a deterministic*
45 *function such that, for some $\epsilon \in [0, 1]$,*

$$\text{for all } x \in \mathbb{R}^d, \quad \|x - \mathcal{C}(x)\|^2 \leq \epsilon \|x\|^2. \quad (2)$$

46 The standard way to improve CGD is to leverage the asymmetry of information: each agent has access
47 to the exact gradient before compression and can therefore track the discrepancy between the exact
48 gradient and the transmitted (compressed) message. This discrepancy can be stored and used as a
49 correction term in subsequent iterations—a principle that lies at the heart of *error feedback techniques*.
50 The most basic mechanism used is known as *classic error feedback* (EF), where each agent stores the
51 difference between the true gradient and its compressed version locally, and incorporates this error
52 into the next round of communication. This method, outlined in Algorithm 2, was first introduced
53 in [3] and later analyzed in [32, 11, 10, 8, 35]. Notably, this method converges in many practical
54 settings, effectively addressing the convergence issues of CGD.

Algorithm 2 Classic error feedback (EF)

1: **initialization:** $x_0 \in \mathbb{R}^d, \eta > 0, e_0^{(i)} = 0$ for $i = 1, \dots, n$
2: **for** $k = 0, 1, 2, \dots, N$ **do**
3: Agent $i \in [n]$ compresses $e_k^{(i)} + \eta \nabla f_i(x_k)$ and communicates $m_k^{(i)} := \mathcal{C}(e_k^{(i)} + \eta \nabla f_i(x_k))$
4: Agent $i \in [n]$ updates $e_k^{(i)} \leftarrow e_k^{(i)} + \eta \nabla f_i(x_k) - \mathcal{C}(e_k^{(i)} + \eta \nabla f_i(x_k))$
5: Server updates $x_{k+1} \leftarrow x_k - \frac{1}{n} \sum_{i=1}^n m_k^{(i)}$
6: **end for**

55 More recently, a variant of the classic error feedback mechanism, known as EF²¹ was introduced
56 by Richtarik et al. [14], and is presented in Algorithm 3. Unlike classic error feedback, EF²¹ focuses
57 on communicating a gradient estimate that is more robust to the variance observed in gradients
58 received from *different* agents around the minimum of finite sum objectives. This method has since
59 been extended in several directions [e.g. 23, 36–38].

60 Error feedback techniques are widely regarded as highly effective, and EF was described as “*com-*
61 *pression for free*” as early as 2019 [10]. Despite that, and the abundant literature on the topic,
62 the precise impact of error feedback techniques on performance remains difficult to assess. Com-
63 parison is complicated by the diversity of settings under which methods are analyzed: different
64 function classes (smooth, convex, or nonconvex), a range of algorithmic enhancements (acceleration,
65 adaptivity, variance reduction, etc.), and a variety of performance measures (different Lyapunov func-
66 tions) [39, 40, 38, 41–44]. While some works provide insightful counter-examples—e.g., Beznosikov
67 et al. [24] show that classic error feedback effectively addresses the limitations of CGD in distributed
68 settings—many others simply propose a Lyapunov function and establish an upper bound without

Algorithm 3 Error Feedback 21 – EF²¹

```
1: initialization:  $x_0 \in \mathbb{R}^d$ ; step size  $\eta > 0$ ;  $d_0^{(i)} = \mathcal{C}(\nabla f_i(x_0))$  for  $i = 1, \dots, n$ ;  
2: for  $k = 0, 1, 2, \dots, N$  do  
3:   Server updates  $x_{k+1} \leftarrow x_k - \eta \cdot \frac{1}{n} \sum_{i=1}^n d_k^{(i)}$   
4:   Agent  $i \in [n]$  compresses  $\nabla f_i(x_{k+1}) - d_k^{(i)}$  and communicates  $m_k^{(i)} := \mathcal{C}(\nabla f_i(x_{k+1}) - d_k^{(i)})$   
5:   Agent  $i \in [n]$  updates  $d_{k+1}^{(i)} \leftarrow d_k^{(i)} + m_k^{(i)}$   
6: end for
```

69 demonstrating its tightness. As a result, claims about “compression for free” are often based on
70 comparisons between potentially loose guarantees, which may not reliably reflect real algorithmic
71 performance. Typically, the length and complexity of the proofs involved makes it very difficult to
72 ensure results’ tightness, and most proofs are constructed in ad-hoc fashions. Consequently, it is
73 difficult to determine which methods are actually worst-case optimal based on upper bounds whose
74 tightness is not always assured.

75 As a result, even remarkably simple questions remain only partially answered:

What is the optimal convergence rate that each method can attain?
Given an optimization setting, what method should we choose?
How should each method be optimally tuned?

76

77 Our goal is to provide definitive answers to parts of these questions. In this paper, we take a
78 complementary perspective to the existing literature and offer a tight, principled comparison of the
79 three methods. Specifically, we derive their optimal tuning, identify an optimal Lyapunov function
80 for each method, and compute the *exact* optimal convergence rate for *any* Lyapunov function within
81 our class of candidate Lyapunov functions.

82 To make this comparison sharp and transparent, we adopt a deliberately simple yet representative
83 setup: we consider smooth and strongly convex functions in the single-agent setting ($n = 1$). While
84 simple, this regime is widely recognized as a crucial stepping stone—not only for building intuition,
85 but also as its own theoretical contribution [e.g., 32, 10]. In this context, *tightness* means that we
86 identify the best possible Lyapunov function within a given class *and* compute the exact worst-case
87 convergence rate over the class of problems considered.

88 Our methodology draws on the *performance estimation* framework [45, 46], which enables the
89 numerical derivation of exact convergence rates for a wide range of first-order methods. In particular,
90 recent advances [47, 48] demonstrate how to automatically search for optimal Lyapunov functions.
91 While these approaches are primarily numerical, we build upon insights from their underlying proof
92 structures [49] to derive new analytical results.

93 **Contributions.** We make the following contributions:

- 94 1. From a methodological standpoint, this paper is the first to apply the performance estimation
95 methodology to the analysis of algorithms originating from the Federated Learning community and
96 incorporating compression schemes. Leveraging this methodology, numerically and analytically,
97 opens numerous doors towards a more precise and reliable understanding of federated and
98 distributed learning methods.
- 99 2. We provide a *tight* analysis of EF and EF²¹, and compare them with compressed gradient descent
100 in the single-agent setting, on L -smooth, μ -strongly convex functions. In particular, we give
101 an analytical formula of the best possible contraction rate, by analyzing an optimal Lyapunov
102 function within a class of candidate Lyapunov functions defined in Definition 1. Furthermore, we
103 provide the optimal tuning for the step size in both those algorithms.
- 104 3. We demonstrate that those rates are achieved, proving that our analysis is tight.
- 105 4. We conclude that the complexities of EF and EF²¹ are perfectly identical in that particular
106 situation, and that CGD outperforms both methods, both in terms of settings in which it can
107 converge, and optimal convergence rate.

108 5. Further, we contribute to the process of deriving *simple* Lyapunov functions for first-order methods,
109 and extend known results for fixed-step methods to the setting of methods using compression.

110 **Paper outline.** The rest of the paper is organized as follows. In Section 2, we provide background
111 on the relevant existing results for CGD, EF, and EF²¹. We also provide the necessary background
112 on the techniques from the performance estimation literature needed to outline the methodology
113 we use, as well as the definition of the classes of Lyapunov functions used. Section 3 contains
114 the main contribution of the paper, namely the tight convergence guarantees for CGD, EF, and
115 EF²¹, with matching lower bounds. Section 4 details the methodology we use to derive the results,
116 and provides references to the formal results required to justify this approach. It also contains a
117 number of numerical results that illustrate the equivalence between EF and EF²¹, and performance
118 characteristics of the three methods. Section 5 summarizes the results of the paper and provides a
119 discussion of the results in relation to the points brought up in the introductory section.

120 **Notations:** We denote \mathbb{S}^ℓ the symmetric matrices, and denote \mathbb{S}_+^ℓ the set of positive semi definite
121 matrices. For any two matrices $A \in \mathbb{S}^\ell$ and $B \in \mathbb{S}^d$, we denote $A \otimes B$ the Kronecker product.

122 2 Background

123 In this section, we briefly overview relevant existing results from the field of distributed optimiza-
124 tion, the necessary background on the performance estimation framework, provide the rest of the
125 assumptions we will need, and specify the notion of Lyapunov functions used in this paper.

126 2.1 Theoretical results on CGD, EF, EF²¹

127 In the single agent case, we leverage the equivalence between compressed gradient descent (CGD),
128 under Assumption 1 and specific instances of the *inexact gradient method*. The former corresponds to
129 the particular case of Algorithm 1 with $n = 1$. In the second, a centralized computing unit optimizing
130 a function f has access to an oracle on the gradient that provides erroneous values, a setup dating back
131 to d’Aspremont [50]. In particular, the model in which the error on the gradient is relatively bounded,
132 i.e., for any x , the oracle queried at point x outputs a value g such that $\|g - \nabla f(x)\|^2 \leq \epsilon \|x\|^2$, is
133 studied in [51]. Authors show that inexact gradient method then enjoys tight convergence guarantees
134 for any step size $\eta > 0$, with respect to the functional residual, Euclidean norm distance to the
135 solution and gradient norm. However, CGD is known to diverge when applied using stochastic
136 gradient oracles, and to non-smooth functions [10]. Interestingly, it is also known to diverge in the
137 multi-worker setting [24]. Studying CGD is important in its own right, because when the compression
138 operator is chosen as the sign function, and the algorithm is applied in the stochastic setting (i.e.,
139 signSGD), there is a connection to Adam both in the convex [52], and non-convex setting [53].

140 In Richtarik et al. [14], the authors study the multi-worker setting, with (potentially) randomized
141 compression operators. They establish a $\mathcal{O}(k^{-1})$ convergence rate on Lipschitz smooth, and a linear
142 rate under additional assumption that the functions satisfy the Łojasiewicz inequality. These results
143 are obtained using a Lyapunov function; however, without tightness guarantees—neither for the choice
144 of Lyapunov function, nor for the convergence rate itself. Extensions of EF²¹ have been proposed,
145 including adaptations to stochastic gradients [39], and the introduction of a momentum term to
146 improve sample complexity in the stochastic setting [36].

147 2.2 Performance estimation

148 Performance estimation tools [54, 55, 46] enable to obtain tight (i.e., exact worst-case) numerical
149 guarantees for convergence rates for multiple choices of Lyapunov functions. To that end, the
150 estimation of the worst case convergence rate is written as an SDP, that is solved numerically using
151 classical solver like MOSEK [56]. Numerical results thus correspond to approximations of the exact
152 worst case rate for an algorithm over a class of functions, and are not to be confused with data, initial
153 point, or problem dependent quantities.

154 This framework has been made accessible through software packages in both Python [57] and
155 Matlab [58], enabling researchers to easily apply these tools. To construct optimal Lyapunov
156 functions for first-order methods, advanced performance estimation techniques within this framework

157 have been developed [47, 48, 59], based on the Dual version of the aforementioned SDP. Particularly
 158 relevant is the approach of Taylor et al. [47], which formulates the search for *quadratic* Lyapunov
 159 functions as a feasibility problem with a candidate contraction rate. By performing bisection on this
 160 rate, the method identifies the smallest contraction rate for which a valid Lyapunov function exists.

161 Another relevant line of work we leverage to discover the analytical form of the Lyapunov lies in the
 162 field of *symbolic regression*, which aims to solve supervised learning tasks over the space of simple
 163 analytic expressions. Recent advances have focused on using genetic programming to search this
 164 space. Software packages have been developed in both Python and Julia, making these techniques
 165 more accessible [60].

166 2.3 Definitions & Notation

167 We have already introduced the notion of a contractive compression operator in Assumption 1. To
 168 position our contribution within the broader literature, we now specify that our analysis is restricted
 169 to the setting of smooth, strongly convex functions:

170 **Assumption 2.** *The function f is L -smooth, i.e., for all $x, y \in \mathbb{R}^d$, we have $f(y) \leq f(x) +$
 171 $\nabla f(x)^\top (y - x) + \frac{L}{2} \|y - x\|^2$.*

172 **Assumption 3.** *The function f is μ -strongly convex, i.e., for all $x, y \in \mathbb{R}^d$, we have $f(y) \geq$
 173 $f(x) + \nabla f(x)^\top (y - x) + \frac{\mu}{2} \|y - x\|^2$.*

174 We will use the notation $\mathcal{F}_{\mu,L}$ to denote the set of smooth, strongly convex functions with parameters μ
 175 and L . We will denote $\kappa := \frac{L}{\mu}$ the condition number. For any objective function $f \in \mathcal{F}_{\mu,L}$, we denote
 176 $x_\star := \arg \min_{x \in \mathbb{R}^d} f$ its minimizer, and $f_\star := \min_{x \in \mathbb{R}^d} f(x)$ its minimum value.

177 **Lyapunov functions.** We now formally define the class of Lyapunov functions under consideration.

178 We formally denote $\mathcal{M} : \mathbb{R}^{\ell \times d} \times \mathbb{R}^d \times \mathcal{F} \rightarrow \mathbb{R}^{\ell \times d} \times \mathbb{R}^d$ a first-order method acting on a set of
 179 functions \mathcal{F} , for an integer $\ell \in \mathbb{N}$. Such a method, for a function $f \in \mathcal{F}$, is applied to an initial *state*
 180 $\xi_0 \in \mathbb{R}^{\ell \times d}$ and iterate $x_0 \in \mathbb{R}^d$ and generates a sequence $\{\xi_k\}_{k \geq 0}$ of states, and a sequence $\{x_k\}_{k \geq 0}$
 181 of iterations. The *states* encode the external information the algorithms may depend on, beyond the
 182 current iterate, in particular will encode memory-type states in error feedback algorithms. The integer
 183 ℓ is thus typically small, from 0 to 3 in general. The exact form of the states used will be specified in
 184 the context of each algorithm.

185 **Definition 1** (Candidate Lyapunov functions). *A function $\mathcal{V} : \mathbb{R}^{\ell \times d} \times \mathbb{R}^d \rightarrow \mathbb{R}$ is called a candidate*
 186 *Lyapunov function for f if it satisfies the following conditions:*

- 187 1. (Non-negativity) $\mathcal{V}(\xi, x; f) \geq 0$, for any $\xi \in \mathbb{R}^{\ell \times d}$, $x \in \mathbb{R}^d$,
- 188 2. (Zero at fixed-point) $\mathcal{V}(\xi, x; f) = 0$ if and only if $x = x_\star$ and $\xi = \xi_\star$ and for a unique $\xi_\star \in \mathbb{R}^{\ell \times d}$.
- 189 3. (Meaningfully lower bounded) there exists a positive semidefinite matrix $A \in \mathbb{S}^\ell$ and a scalar
 190 $a \geq 0$ such that $\mathcal{V}(\xi, x; f) \geq (\xi - \xi_\star)^\top (A \otimes I_d) (\xi - \xi_\star) + a(f(x) - f_\star)$ and $\text{Tr}(A) + a = 1$.

191 The lower bound in item 3 of our definition requires some justification: it ensures that the Lyapunov
 192 function provides control over meaningful quantities in optimization, such as the distance to the fixed
 193 point, gradient norm, algorithm-dependent quantities and the functional residual.

194 The class of candidate Lyapunov functions is thus given by

$$\mathbb{V}_\ell = \{(P, p) \in \mathbb{S}_+^\ell \times \mathbb{R}^+ : \text{Tr}(P) + p = 1\}. \quad (3)$$

195 For any $(P, p) \in \mathbb{V}_\ell$ we denote $\mathcal{V}_{(P,p)}$ the Lyapunov functions of the form:

$$\mathcal{V}_{(P,p)}(\xi, x; f) = (\xi - \xi_\star)^\top (P \otimes I_d) (\xi - \xi_\star) + p(f(x) - f_\star). \quad (4)$$

196 We seek candidate Lyapunov functions $\mathcal{V} : \mathbb{R}^{\ell \times d} \times \mathbb{R}^d \times \mathcal{F} \rightarrow \mathbb{R}$ that satisfy the recurrence

$$\mathcal{V}(\xi_{k+1}, x_{k+1}; f) \leq \rho \cdot \mathcal{V}(\xi_k, x_k; f), \quad (5)$$

for some constant $\rho < 1$ and for all $k \geq 0$, uniformly over the class \mathcal{F} . Finding the *optimal* Lyapunov function within a parameterized class, for a method \mathcal{M} , then amounts to solving the following problem, that will be our main objective in the following:

$$\rho^*(\mathcal{M}) := \min_{(P,p) \in \mathbb{V}_\ell} \left\{ \max_{\substack{f \in \mathcal{F}_{\mu,L}, \\ (\xi_0, x_0) \in \mathbb{R}^{\ell \times d} \times \mathbb{R}^d}} \frac{\mathcal{V}_{(P,p)}(\xi_1, x_1; f)}{\mathcal{V}_{(P,p)}(\xi_0, x_0; f)} : (\xi_1, x_1) = \mathcal{M}(\xi_0, x_0; f) \right\}. \quad (6)$$

Note that guaranteeing the contraction between step 1 and step 0, uniformly over (ξ_0, x_0) equivalently provides uniform contraction between any two consecutive steps as in (5).

3 Main results

In this section, we provide answers to the questions stated in the introduction for our setting. We begin by showing some numerical results on the performance of each method, and then provide the precise statements of all of our theoretical results.

3.1 Numerical performance of all methods

In order to compare classic error feedback and EF²¹ with the performance of compressed gradient descent, we first need to specify the state-variables under consideration when analyzing each method. Those are given by:

$$\xi_k^{\text{CGD}} = \begin{bmatrix} x_k \\ \nabla f(x_k) \\ \mathcal{C}(\eta \nabla f(x_k)) \end{bmatrix}, \quad \xi_k^{\text{EF}} = \begin{bmatrix} x_k \\ \nabla f(x_k) \\ \mathcal{C}(e_k + \eta \nabla f(x_k)) \\ e_k \end{bmatrix}, \quad \xi_k^{\text{EF}^{21}} = \begin{bmatrix} x_k \\ \nabla f(x_k) \\ d_k \end{bmatrix}, \quad (7)$$

where all variables are defined as in Algorithm 1, Algorithm 2 and Algorithm 3, respectively.

We are now ready to present the numerical results on the performance of each method. Figure 1 shows contour plots of the contraction factor for each method. That is, for a fine grid of both the step size η in Algorithms 1 to 3, and the parameter ϵ in Assumption 1, we compute numerically the value of the best possible (in terms of Lyapunov) worst-case contraction rate over the class $\mathcal{F}_{\mu,L}$, as given by (6). A darker blue point indicates a stronger contraction $\rho^*(\mathcal{M})$ (i.e., a better rate). A red point indicates that the method is non-convergent for that setup of (ϵ, η) . We observe that, while those results are numerical only at that point, they indicate that the performance of EF and EF²¹ are perfectly identical in our setting. This numerical equivalence is supported by Table 1: the maximum absolute difference between contraction factors for EF and EF²¹ is of the order of 10^{-5} to 10^{-7} . This first fact is a very surprising observation. Indeed while EF and EF²¹ are known to be identical [14, see Section 4.2] in the very specific case of using a deterministic positively homogeneous and additive compression operator, EF and EF²¹ remain grounded in fundamentally different motivations at first sight: on the

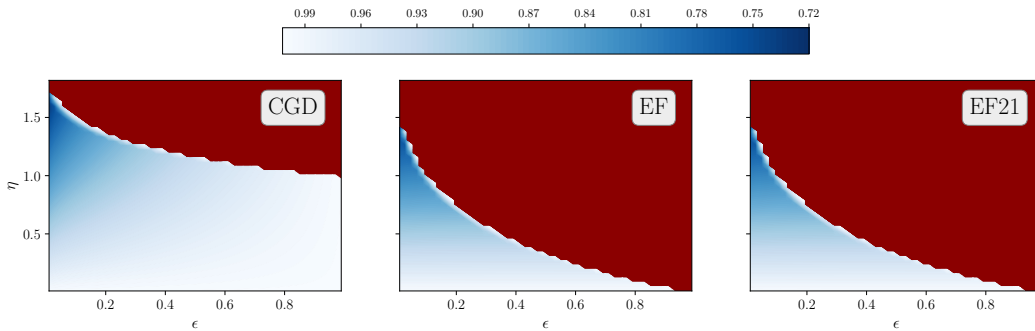


Figure 1: Single row of contour plots showing performance of CGD, EF, and EF²¹ as a function of step size η and compression parameter ϵ , with regions of non-convergence marked in red. The regions of non-convergence were computed using PEPit by finding cycles of length 2.

	$\kappa = 10.0$	$\kappa = 4.0$	$\kappa = 2.0$
Absolute error	1.2e-05	4.9e-07	3.5e-07

Table 1: Maximum absolute difference of contraction factor for EF and EF²¹, computed over grid of $\epsilon \in [0.01, 0.8]$ and $\eta \in [0.01, \frac{2}{L+\mu}]$ for $L = 1$, and varying μ .

	$\kappa = 2.0$	$\kappa = 4.0$	$\kappa = 10.0$
Absolute Error	2.1e-07	2.9e-07	3.0e-07

Table 2: Maximum absolute difference of contraction factor for CGD when allowing any combination of terms in our Lyapunov function, compared to the contraction achieved by the functional residual. Same grid as Table 1.

one hand, EF accounts for the errors introduced by the compression step, while on the other hand, EF²¹ subtracts a control variate from the gradient prior to compression. Proving that the best possible convergence rate they can obtain, for any tuning ϵ, η , is similar (but achieved for a *different* Lyapunov function) was, to the best of our knowledge, never established in the literature. It thus constitutes a significant step towards better understanding of their connections.

A second observation can be made from those plots: the region of non-convergence is by far larger for EF and EF²¹ than for CGD. In particular, there exist multiple tunings, for which incorporating any of the two types of error feedback, actually prevent convergence.

While given for a single (μ, L) in Figure 1 and Table 1, similar results hold for all values of (μ, L) that were tried numerically, and several examples are given in Appendix C, together with details on these numerical experiments, including how we computed the regions of non-convergence for these methods.

Further, we *tune* each algorithm by picking the optimal step size for each method. We compute the rate $\inf_{\eta} \rho^*(\mathcal{M}_{\eta})$, for $\mathcal{M} \in \{\text{CGD}, \text{EF}, \text{EF}^{21}\}$, where \mathcal{M}_{η} corresponds to the method with step size η . Results are given in Figure 2, for three values of κ , naemly, 10, 4 and 2. In each setup, uniformly over the value ϵ of the compression rate parameter, CGD achieves a rate which is strictly better than EF and EF²¹. These results challenge the prevailing intuition that error feedback ensures convergence comparable to that of uncompressed methods, and even demonstrate that in the single agent and deterministic regime, error feedback is actually always detrimental to the convergence.

Finally, we note that the functional residual constitutes an optimal Lyapunov function for CGD, as shown in Table 2. This result is not very surprising, given that a tight analysis of the functional residual for optimal step size was given for *inexact gradient descent* by De Klerk et al. [51]. This rate has been shown to be tight in the same work¹.

In the next two sections, we provide analytical results on respectively EF and EF²¹.

¹Following the same line of reasoning as in our remark on the tightness of our Lyapunov functions in Section 5, we can then show that the functional residual is an optimal Lyapunov function for CGD.

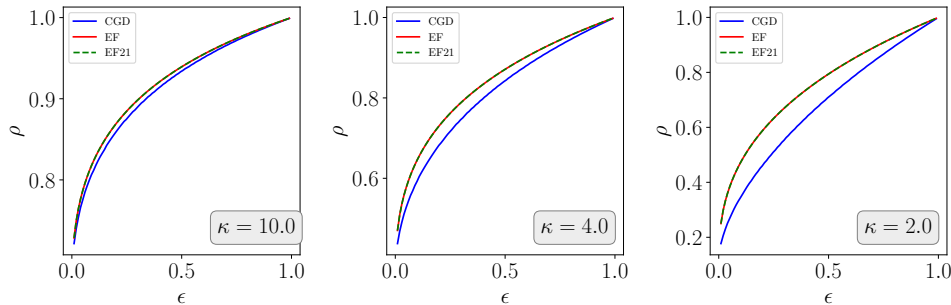


Figure 2: Single row of line plots showing the performance of CGD (blue) and EF (red) as a function of the compression factor ϵ , for $L = 1$, and varying μ .

247 3.2 Exact convergence rate and optimal tuning for Classic error feedback EF

248 We begin by stating the main result of this section, which is a tight rate of convergence for the classic
249 error feedback algorithm within our class of candidate Lyapunov functions.

250 **Theorem 1.** *Consider running Algorithm 2, i.e., EF, with a compression operator \mathcal{C} satisfying*
251 *Assumption 1 for some $\epsilon \in [0, 1]$ on any function satisfying Assumptions 2, and 3. Let the step size be*
252 *given by*

$$\eta^* = \left(\frac{2}{L + \mu} \right) \cdot \left(\frac{1 - \sqrt{\epsilon}}{1 + \sqrt{\epsilon}} \right). \quad (8)$$

253 Then, we have that

$$\rho^*(\text{EF}_{\eta^*}) = \sqrt{\epsilon} + \frac{1}{4}(1 + \sqrt{\epsilon})(L - \mu)\lambda, \quad (9)$$

254 where

$$\lambda := \frac{\eta^*}{L + \mu} \left[(1 - \sqrt{\epsilon})(L - \mu) + (1 + \sqrt{\epsilon})\sqrt{(L - \mu)^2 + 16L\mu\frac{\sqrt{\epsilon}}{(1 + \sqrt{\epsilon})^2}} \right]. \quad (10)$$

255 A Lyapunov function achieving the rate in (9), with ξ^{EF} defined in (14), is given by

$$\mathcal{V}(\xi^{\text{EF}}, x; f) := \|x - x_\star\|^2 - 2(x - x_\star)^\top e + \left(1 + \frac{1}{\sqrt{\epsilon}}\right) \cdot \|e\|^2 = \|x - x_\star - e\|^2 + \frac{1}{\sqrt{\epsilon}}\|e\|^2, \quad (11)$$

256 Finally, the step size in (8) is worst-case optimal for EF: $\forall \eta \geq 0$, we have $\rho^*(\text{EF}_\eta) \geq \rho^*(\text{EF}_{\eta^*})$.

257 Importantly, (9) indicates that the rate is tight, that is, there exist $f \in \mathcal{F}_{\mu, L}$ and $(P, p) \in \mathbb{V}_\ell$ such that
258 the rate is exactly achieved. This is demonstrated formally in the proof, given in Appendix B.1.

259 For completeness, and to support our theoretical results, as well as show the tightness of the numerical
260 results obtained through Performance estimation, we also provide figures comparing the empirically
261 observed optimal step sizes for worst-case instances to our step size η^* for different values of ϵ and
262 μ/L are found in Figure 8 given in Appendix C.4. Numerical and analytical values perfectly match.

263 3.3 Exact convergence rate and optimal tuning for EF²¹

264 We now state our main result on the EF²¹ algorithm, which is also tight within our class of candidate
265 Lyapunov functions.

266 **Theorem 2.** *Consider running Algorithm 3 with a compression operator satisfying Assumption 1*
267 *for some $\epsilon \in [0, 1]$ on any function satisfying Assumptions 2, and 3. Let the step size be given by η^**
268 *in (8). Then, we have that*

$$\rho^*(\text{EF}_{\eta^*}^{21}) = \left[\sqrt{\epsilon} + \frac{1}{4}(1 + \sqrt{\epsilon})(L - \mu)\lambda \right], \quad (12)$$

269 where λ is given by (10). A Lyapunov function achieving the rate in (12) is given by

$$\mathcal{V}(\xi^{\text{EF}^{21}}, x; f) := (1 + \sqrt{\epsilon}) \cdot \|g\|^2 - 2g^\top d + \|d\|^2 = \|g - d\|^2 + \sqrt{\epsilon} \cdot \|d\|^2. \quad (13)$$

270 Finally, the step size η^* is worst-case optimal for this algorithm.

271 The proof of this theorem is given in Appendix B.2. This second theoretical results provides an
272 analytical confirmation of the surprising phenomenon observed numerically in Subsection 3.1, in
273 particular in Figure 1: EF and EF²¹ have the exact same optimal guarantee. Furthermore, the optimal
274 step size is also the same, and can also be argued to be worst-case optimal in our setting using the
275 same arguments as in Section 3.2. However, the optimal Lyapunov function is very different.

276 3.4 Tightness over multiple iterations and choice of state variables

277 A relevant question regarding the analysis above, is the tightness of the Lyapunov functions provided
278 in Theorems 1 and 2, over multiple steps. That is, for $K \geq 2$, is it possible to improve the rate of a
279 method $\mathcal{M} \in \{\text{EF}, \text{EF}^{21}, \text{CGD}\}$ by considering, with \mathcal{M}^K the method running over K iterations:

$$\rho_K^*(\mathcal{M}) := \left(\min_{(P, p) \in \mathbb{V}_\ell} \left\{ \max_{\substack{f \in \mathcal{F}_{\mu, L}, \\ (\xi_0, x_0) \in \mathbb{R}^{\ell \times d} \times \mathbb{R}^d}} \frac{\mathcal{V}_{(P, p)}(\xi_K, x_K; f)}{\mathcal{V}_{(P, p)}(\xi_0, x_0; f)} : (\xi_K, x_K) = \mathcal{M}^K(\xi_0, x_0; f) \right\} \right)^{1/K}.$$

280 We provide a numerical answer to that question in Appendix C.2 for all algorithms discussed in the
 281 paper, showing that the analysis of the single-state Lyapunov functions used in this work are tight
 282 even if we consider multiple iterations. To that end, we plot the worst-case contractions for multiple
 283 iterations computed using PEPit [57], on the optimal Lyapunov given by (11) and (13).

284 **Lyapunov Tightness.** Second, to prove that the Lyapunov functions we use in Theorems 1 and 2 are
 285 tight, one has to show that the rate of convergence for any other candidate Lyapunov function from
 286 our class is lower bounded by the rate we obtain. We consider a quadratic function, used in the latter
 287 sections of our proofs to prove tightness: asymptotic expressions for all the state-variables are the
 288 same, up to an iteration-independent constant. Consequently, when computing ratios between any set
 289 of Lyapunov functions, these constants cancel out. The only thing remaining is the term that actually
 290 depends the iteration count, which is exactly equal to the rate given in Theorems 1 and 2.

291 4 Methodology

292 In this section, we present the methodology used to obtain the results in Section 3. The approach
 293 builds on the framework developed by Taylor et al. [47]. We extend it to cover methods using
 294 compression operators under Assumption 1. We provide formal statements and proofs in Appendix A
 295 for completeness. Obtaining the proofs of Theorems 1 and 2 requires the combination of advanced
 296 performance estimation results (e.g. optimal Lyapunov search), several tricks, as well as symbolic
 297 computation and symbolic regression frameworks.

298 In particular, first, to solve the problem (6), we begin by addressing the inner maximization problem
 299 for a fixed contraction factor ρ . This amounts to checking the feasibility of a semidefinite program,
 300 detailed in Appendix A. We then apply bisection on ρ to identify the smallest admissible contraction
 301 factor. However, the Lyapunov function given is rarely unique, and most solutions obtained numeri-
 302 cally vary significantly with problem parameters such as the compression factor ϵ . To address this,
 303 we propose to use rank minimization heuristics—specifically, the logdet heuristic [61]. This enables to
 304 obtain a unique set of Lyapunov functions with low rank, structurally simpler. Finally, we proceed
 305 by eliminating redundant coefficients in the matrix P and the scalar p , to arrive at the concise forms
 306 presented in Section 3. At the end of this process, the Lyapunov function coefficients were found to
 307 be mutually dependent, reducing the problem to identifying a closed-form expression for any one of
 308 them. To estimate such a coefficient, we applied symbolic regression using the PySR Python package
 309 [60]. This approach proved highly effective at finding simple yet *optimal* Lyapunov functions. To
 310 arrive at simple and readable proofs, we leverage the computer algebra system of *Mathematica* [62].

311 We emphasize this methodology as we believe it has broad applicability to other problems in ML.

312 5 Conclusion

313 In this paper, we provided tight analyses of EF and EF²¹ using Lyapunov functions, with guarantees
 314 on both the Lyapunov functions themselves and the convergence rates achieved. Notably, both
 315 algorithms exhibit the same convergence rate in our setting, and through a remark made in the
 316 discussion below, this gives us tight rates for any of the candidate Lyapunov functions we considered
 317 in our class. We also observed that their performance is strictly worse than that of compressed
 318 gradient descent—an outcome that, we believe, challenges the intuition of many in the field.

319 Our analysis is confined to the single-agent setting, both as a fundamental building block, and as a
 320 source of intuition for the multi-agent case. Importantly, CGD cannot serve as a baseline in the multi-
 321 agent setting as fails to converge with more than one agent [24]. In contrast, EF²¹ was specifically
 322 designed to improve convergence over EF in the multi-agent setting. Yet, its convergence rate in the
 323 single-agent case matches exactly that of EF—raising the question of how these two methods compare
 324 in a rigorous analysis with multiple agents. The findings of this paper raise two compelling questions:

- 325 • *Does the performance of EF and EF²¹ differ in the multi-agent setting?*
- 326 • *Are there more effective error compensation mechanisms yet to be discovered?*

327 We leave these questions for future work and conclude by emphasizing that the methodology used is
 328 likely applicable to a broad range of problems in optimization for machine learning. We look forward
 329 to seeing it extended and applied in future research.

References

- [1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, April 2017. ISSN: 2640-3498.
- [2] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and Open Problems in Federated Learning. *arXiv:1912.04977 [cs, stat]*, December 2019. arXiv: 1912.04977.
- [3] Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Fifteenth Annual Conference of the International Speech Communication Association*. Citeseer, 2014.
- [4] Trishul Chilimbi, Yutaka Suzue, Johnson Apacible, and Karthik Kalyanaraman. Project adam: Building an efficient and scalable deep learning training system. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 571–582, 2014.
- [5] Nikko Strom. Scalable distributed DNN training using commodity GPU cloud computing. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.
- [6] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning (ICML)*, 2020.
- [7] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding. *Advances in Neural Information Processing Systems*, 30:1709–1720, 2017.
- [8] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cedric Renggli. The Convergence of Sparsified Gradient Methods. *Advances in Neural Information Processing Systems*, 31:5973–5983, 2018.
- [9] Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed Learning with Compressed Gradient Differences. *arXiv:1901.09269 [cs, math, stat]*, June 2019. arXiv: 1901.09269.
- [10] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error Feedback Fixes SignSGD and other Gradient Compression Schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR, May 2019. ISSN: 2640-3498.
- [11] Jiaxiang Wu, Weidong Huang, Junzhou Huang, and Tong Zhang. Error Compensated Quantized SGD and its Applications to Large-scale Distributed Optimization. In *International Conference on Machine Learning*, pages 5325–5333. PMLR, July 2018. ISSN: 2640-3498.
- [12] Samuel Horvath, Chen-Yu Ho, Ludovít Horvath, Atal Narayan Sahu, Marco Canini, and Peter Richtárik. Natural compression for distributed deep learning. In *Mathematical and Scientific Machine Learning*, pages 129–141. PMLR, 2022.
- [13] Zhize Li, Dmitry Kovalev, Xun Qian, and Peter Richtárik. Acceleration for Compressed Gradient Descent in Distributed and Federated Optimization. In *International Conference on Machine Learning*, pages 5895–5904. PMLR, November 2020. ISSN: 2640-3498.

- [14] Peter Richtarik, Igor Sokolov, and Ilyas Fatkhullin. EF21: A New, Simpler, Theoretically Better, and Practically Faster Error Feedback. In *Advances in Neural Information Processing Systems*, volume 34, pages 4384–4396. Curran Associates, Inc., 2021.
- [15] Ibrahim El Khalil Harrane, Rémi Flamary, and Cédric Richard. On reducing the communication cost of the diffusion lms algorithm. *IEEE Transactions on Signal and Information Processing over Networks*, 5(1):100–112, 2018.
- [16] Hanlin Tang, Chen Yu, Xiangru Lian, Tong Zhang, and Ji Liu. DoubleSqueeze: Parallel Stochastic Gradient Descent with Double-pass Error-Compensated Compression. In *International Conference on Machine Learning*, pages 6155–6165. PMLR, May 2019. ISSN: 2640-3498.
- [17] Xiaorui Liu, Yao Li, Jiliang Tang, and Ming Yan. A Double Residual Compression Algorithm for Efficient Distributed Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 133–143, June 2020. ISSN: 1938-7228 Section: Machine Learning.
- [18] Shuai Zheng, Ziyue Huang, and James Kwok. Communication-Efficient Distributed Blockwise Momentum SGD with Error-Feedback. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- [19] Constantin Philippenko and Aymeric Dieuleveut. Artemis: tight convergence guarantees for bidirectional compression in Federated Learning. *arXiv:2006.14591 [cs, stat]*, November 2020. arXiv: 2006.14591.
- [20] Constantin Philippenko and Aymeric Dieuleveut. Preserved central model for faster bidirectional compression in distributed settings. *Advances in Neural Information Processing Systems*, 34, 2021.
- [21] Eduard Gorbunov, Dmitry Kovalev, Dmitry Makarenko, and Peter Richtarik. Linearly Converging Error Compensated SGD. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20889–20900. Curran Associates, Inc., 2020.
- [22] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and Communication-Efficient Federated Learning From Non-i.i.d. Data. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2019. ISSN 2162-2388. doi: 10.1109/TNNLS.2019.2944481. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.
- [23] Ilyas Fatkhullin, Igor Sokolov, Eduard Gorbunov, Zhize Li, and Peter Richtárik. EF21 with Bells & Whistles: Practical Algorithmic Extensions of Modern Error Feedback, October 2021. arXiv:2110.03294 [cs, math].
- [24] Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On Biased Compression for Distributed Learning. *arXiv:2002.12410 [cs, math, stat]*, February 2020. arXiv: 2002.12410.
- [25] Sélim Chraïbi, Ahmed Khaled, Dmitry Kovalev, Peter Richtárik, Adil Salim, and Martin Takáč. Distributed fixed point methods with compressed iterates. *arXiv preprint arXiv:1912.09925*, 2019.
- [26] Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent. In *International Conference on Artificial Intelligence and Statistics*, pages 680–690. PMLR, 2020.
- [27] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. FedPAQ: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2021–2031. PMLR, June 2020. ISSN: 2640-3498.
- [28] Dmitry Kovalev, Elnur Gasanov, Alexander Gasnikov, and Peter Richtarik. Lower bounds and optimal algorithms for smooth and strongly convex decentralized optimization over time-varying networks. *Advances in Neural Information Processing Systems*, 34:22325–22335, 2021.

- [29] Farzin Haddadpour, Mohammad Mahdi Kamani, Aryan Mokhtari, and Mehrdad Mahdavi. Federated learning with compression: Unified analysis and sharp guarantees. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2350–2358. PMLR, 13–15 Apr 2021. URL <https://proceedings.mlr.press/v130/haddadpour21a.html>.
- [30] Zhize Li and Peter Richtárik. Canita: Faster rates for distributed convex optimization with communication compression. *Advances in Neural Information Processing Systems*, 34:13770–13781, 2021.
- [31] Sarit Khirirat, Hamid Reza Feyzmahdavian, and Mikael Johansson. Distributed learning with compressed gradients. *arXiv preprint arXiv:1806.06573*, 2018.
- [32] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with Memory. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 4447–4458. Curran Associates, Inc., 2018.
- [33] Nikita Iykin, Daniel Rothchild, Enayat Ullah, Vladimir Braverman, Ion Stoica, and Raman Arora. Communication-efficient Distributed SGD with Sketching. *Advances in Neural Information Processing Systems*, 32:13144–13154, 2019.
- [34] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication. In *International Conference on Machine Learning*, pages 3478–3487. PMLR, May 2019. ISSN: 2640-3498.
- [35] Sebastian U Stich and Sai Praneeth Karimireddy. The error-feedback framework: Better rates for SGD with delayed gradients and compressed updates. *Journal of Machine Learning Research*, 21:1–36, 2020.
- [36] Ilyas Fatkhullin, Alexander Tyurin, and Peter Richtárik. Momentum provably improves error feedback! *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [37] Kaja Gruntkowska, Alexander Tyurin, and Peter Richtárik. EF21-P and Friends: Improved Theoretical Communication Complexity for Distributed Optimization with Bidirectional Compression, September 2022. arXiv:2209.15218 [cs, math].
- [38] Dmitry Makarenko, Elnur Gasanov, Rustem Islamov, Abdurakhmon Sadiev, and Peter Richtárik. Adaptive compression for communication-efficient distributed training. *arXiv preprint arXiv:2211.00188*, 2022.
- [39] Ilyas Fatkhullin, Igor Sokolov, Eduard Gorbunov, Zhize Li, and Peter Richtárik. Ef21 with bells & whistles: Practical algorithmic extensions of modern error feedback. *arXiv preprint arXiv:2110.03294*, 2021.
- [40] Peter Richtárik, Igor Sokolov, Ilyas Fatkhullin, Elnur Gasanov, Zhize Li, and Eduard Gorbunov. 3pc: Three point compressors for communication-efficient distributed training and a better theory for lazy aggregation. *arXiv preprint arXiv:2202.00998*, 2022.
- [41] Kaja Gruntkowska, Alexander Tyurin, and Peter Richtárik. Ef21-p and friends: Improved theoretical communication complexity for distributed optimization with bidirectional compression. *arXiv preprint arXiv:2209.15218*, 2022.
- [42] Haoyu Zhao, Boyue Li, Zhize Li, Peter Richtárik, and Yuejie Chi. Beer: Fast $\mathcal{O}(1/t)$ rate for decentralized nonconvex optimization with communication compression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [43] Yujun Wang, Lu Lin, and Jinghui Chen. Communication-compressed adaptive gradient method for distributed nonconvex optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 6292–6320, 2022.
- [44] Ron Dorfman, Shay Vargaftik, Yaniv Ben-Itzhak, and Kfir Y. Levy. Docofl: Downlink compression for cross-device federated learning. *arXiv preprint arXiv:2302.00543*, 2023.

- [45] Yoel Drori. *Contributions to the Complexity Analysis of Optimization Algorithms*. PhD thesis, Tel-Aviv University, 2014.
- [46] Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. Exact worst-case performance of first-order methods for composite convex optimization. *SIAM Journal on Optimization*, 27(3): 1283–1313, 2017.
- [47] Adrien Taylor, Bryan Van Scoy, and Laurent Lessard. Lyapunov functions for first-order methods: Tight automated convergence guarantees. In *International Conference on Machine Learning (ICML)*, 2018.
- [48] Manu Upadhyaya, Sebastian Banert, Adrien B. Taylor, and Pontus Giselsson. Automated tight lyapunov analysis for first-order methods. *preprint arXiv:2302.06713*, 2023.
- [49] Baptiste Goujaud, Aymeric Dieuleveut, and Adrien Taylor. On fundamental proof structures in first-order optimization. In *Conference on Decision and Control (CDC)*, 2023.
- [50] Alexandre d’Aspremont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19(3):1171–1183, 2008.
- [51] Etienne De Klerk, Francois Glineur, and Adrien B Taylor. Worst-case convergence analysis of inexact gradient and newton methods through semidefinite programming performance estimation. *SIAM Journal on Optimization*, 30(3):2053–2082, 2020.
- [52] Lukas Balles and Philipp Hennig. Dissecting adam: The sign, magnitude and variance of stochastic gradients. In *International Conference on Machine Learning*, pages 404–413. PMLR, 2018.
- [53] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.
- [54] Yoel Drori and Marc Teboulle. Performance of first-order methods for smooth convex minimization: a novel approach. *Mathematical Programming*, 145(1):451–482, 2014.
- [55] Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. Smooth strongly convex interpolation and exact worst-case performance of first-order methods. *Mathematical Programming*, 161(1-2):307–345, 2017.
- [56] ApS MOSEK. *MOSEK Optimizer API for C 9.3.6*, 2019. URL <https://docs.mosek.com/latest/capi/index.html>.
- [57] Baptiste Goujaud, Céline Moucer, François Glineur, Julien Hendrickx, Adrien Taylor, and Aymeric Dieuleveut. PEPit: computer-assisted worst-case analyses of first-order optimization methods in Python. *preprint arXiv:2201.04040*, 2022.
- [58] Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. Performance estimation toolbox (PESTO): automated worst-case analysis of first-order optimization methods. In *Conference on Decision and Control (CDC)*, 2017.
- [59] Adrien Taylor and Francis Bach. Stochastic first-order methods: non-asymptotic and computer-aided analyses via potential functions. In *Conference on Learning Theory (COLT)*, 2019.
- [60] Miles Cranmer. Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl, May 2023. URL <http://arxiv.org/abs/2305.01582>. arXiv:2305.01582 [astro-ph, physics:physics].
- [61] Maryam Fazel, Haitham Hindi, and Stephen P. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *American Control Conference (ACC)*, 2003.
- [62] Wolfram Research, Inc. Mathematica, Version 14.2. URL <https://www.wolfram.com/mathematica>. Champaign, IL, 2024.
- [63] Baptiste Goujaud, Aymeric Dieuleveut, and Adrien Taylor. Counter-examples in first-order optimization: a constructive approach. *IEEE Control Systems Letters*, 2023. (See [arXiv 2303.10503](https://arxiv.org/abs/2303.10503) for complete version with appendices).

Organization of the appendix

A Feasibility problems with compressors	14
B Missing proofs	21
C Additional numerical results	24

This appendix provides additional content and details complementing the paper. In particular, Appendix A details the general methodology used to search for Lyapunov functions. The complete missing proofs for the main results of the paper are presented in Appendix B. Finally, Appendix C presents additional numerical results that informally motivate a few choices made in the paper and provide numerical validation of our claims.

A Feasibility problems with compressors

This section presents the methodology used to search for Lyapunov functions. In a nutshell, we formulate the Lyapunov search problem as a quasi-convex optimization problem involving linear matrix inequalities. Those problems are typically solved through the use of an iterative procedure involving a binary search with semidefinite solvers—we use MOSEK [56] all along. The main steps taken here can be seen as a generalization of the procedure proposed in [47] to the setup of compression, in particular Algorithms 2 and 3—in fact, we also simplify a few steps that are not needed for our purposes. We start by reviewing the technique on a simpler example in Appendix A.1 before detailing the more tricky formulations involving compression.

A.1 Feasibility problem for gradient descent

To introduce the ideas required to understand the techniques used to find Lyapunov functions for Algorithms 2 and 3, we first consider the simpler case of finding Lyapunov functions for **gradient descent** on smooth strongly convex functions. That is, we consider the algorithm:

$$x_1 = x_0 - \eta \nabla f(x_0). \quad (\text{GN}_\eta)$$

The goal of this subsection is to review the steps that enable to obtain $\rho^*(\text{GN}_\eta)$, as defined in Eq. (6), proceeding by a bisection search, each test requiring verifying the feasibility of a convex problem.

Our starting point is to consider the following state variable for GD:

$$\xi_k^{\text{GD}} = \begin{bmatrix} x_k \\ \nabla f(x_k) \end{bmatrix} \quad (14)$$

and a natural family of Lyapunov candidates (which corresponds to a subset of Eq. (3)) of the form

$$\begin{aligned} \mathcal{V}_P(\xi_k^{\text{GD}}; f) &= \mathcal{V}_P(x_k, \nabla f(x_k); f) := \begin{bmatrix} x_k - x_\star \\ \nabla f(x_k) \end{bmatrix}^\top (P \otimes I_d) \begin{bmatrix} x_k - x_\star \\ \nabla f(x_k) \end{bmatrix} \\ &= P_{11} \|x_k - x_\star\|^2 + P_{22} \|\nabla f(x_k)\|^2 + 2P_{12} \langle \nabla f(x_k); x_k - x_\star \rangle, \end{aligned} \quad (15)$$

where $P \in \mathbb{S}^d$ is positive semidefinite and we require $\text{Tr}(P) = 1$ (without loss of generality, for normalization purposes to avoid $P = 0$).

The problem we want to solve, is that of finding the *best* Lyapunov function among our candidates, that is, the one for which the ratio $\frac{\mathcal{V}_P(x_1, \nabla f(x_1))}{\mathcal{V}_P(x_0, \nabla f(x_0))}$ can be upper bounded by the smallest possible constant, uniformly over all optimization problems within the family we consider. That is, we search for the Lyapunov candidate with the smallest possible ρ such that

$$\frac{\mathcal{V}_P(x_1, \nabla f(x_1); f)}{\mathcal{V}_P(x_0, \nabla f(x_0); f)} \leq \rho \quad (16)$$

is valid for all L -smooth μ -strongly convex function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ (for any dimension $d \in \mathbb{N}$) and all possible $x_0, x_1, x_\star \in \mathbb{R}^d$ compatible with f and $x_1 = x_0 - \eta \nabla f(x_0)$. That is, we aim to solve

$$\rho^*(\text{GN}_\eta) = \min_{P \succ 0} \left(\max_{\substack{d \in \mathbb{N} \\ f \in \mathcal{F}_{\mu, L} \\ x_0, x_\star \in \mathbb{R}^d}} \left\{ \frac{\mathcal{V}_P(x_1, \nabla f(x_1); f)}{\mathcal{V}_P(x_0, \nabla f(x_0); f)} \text{ s.t. } \text{Tr}(P) = 1, x_1 = x_0 - \eta \nabla f(x_0) \right\} \right) \quad (17)$$

where $\eta > 0$ is the step size. One can reformulate this problem directly using tools from the performance estimation literature [54, 55], but the resulting problem is not convex in the variable ρ . One way of addressing this is to reduce it to, for a given contraction factor ρ , the problem of finding *some* Lyapunov function that achieves this ρ —if one exists. This problem, on the other hand, is convex, and we can simply perform bisection search on ρ to find the smallest possible contraction factor.

We now introduce some notation to simplify the statement of our Lyapunov finding problem:

$$\sigma_\rho(x_1, g_1, x_0, g_0; P) := \mathcal{V}_P(x_1, g_1; f) - \rho \mathcal{V}_P(x_0, g_0; f). \quad (18)$$

To understand how to

1. **Step 1:** *verifying* a given Lyapunov function and rate as a convex problem

2. **Step 2:** *verifying* a given rate as a convex problem.

Step 1: verifying a given Lyapunov function and rate as a convex problem.

For a Lyapunov parameter $P \in \mathbb{V}_2$ and a tentative rate $\rho > 0$ fixed, we can then state the problem of *verifying* a given Lyapunov function as that of showing that the minimum value of the following is non-positive:

$$\begin{aligned} 0 \geq & \sup_{\substack{d \in \mathbb{N} \\ f \in \mathcal{F}_{\mu, L} \\ x_0, x_\star \in \mathbb{R}^d}} \sigma_\rho(x_1, \nabla f(x_1), x_0, \nabla f(x_0); P; f) \\ \text{s.t. } & x_1 = x_0 - \eta \nabla f(x_0) \\ & \nabla f(x_\star) = 0 \end{aligned} \quad (19)$$

The constraint that f is a smooth strongly convex function is easily encoded using interpolation conditions [55]. This allows us to work with *sampled points* from f rather than the infinite dimensional set $\mathcal{F}_{\mu, L}$. We introduce the notation

$$\phi_{ij} := f_i - f_j - g_j^\top (x_i - x_j) - \frac{1}{2L} \|g_i - g_j\|^2 - \frac{\mu}{2(1 - \mu/L)} \|x_i - x_j - \frac{1}{L}(g_i - g_j)\|^2, \quad (20)$$

where the notation (x_i, g_i, f_i) is used to denote a sampled triplet from f , such that $f(x_i) = f_i$ and $\nabla f(x_i) = g_i$ for all $i \in \{0, 1, \star\}$. This lets us rephrase our problem as

$$\begin{aligned} 0 \geq & \sup_{\substack{d \in \mathbb{N} \\ x_\star, x_0, g_\star, g_0, g_1 \in \mathbb{R}^d \\ f_0, f_1 \in \mathbb{R}}} \sigma_\rho(x_1, g_1, x_0, g_0; P; f) \\ \text{s.t. } & \phi_{ij} \geq 0 \quad \forall i, j \in \{0, 1, \star\} \\ & x_1 = x_0 - \eta \nabla f(x_0) \\ & g_\star = 0 \end{aligned} \quad (21)$$

The above problem is not convex due to the interpolation constraints. To address this, we reformulate it as a semidefinite program (SDP). Let $G = B^\top B$ where B is the following $(3 \times d)$ matrix (note that we use $x_0 - x_\star$ for convenience here, as well as $f_i - f_\star$; this amounts to set without loss of generality $x_\star = 0$ and $f_\star = 0$, which is what is done in other sections):

$$B = [x_0 - x_\star, g_0, g_1],$$

and let $\mathbf{f} := \begin{bmatrix} f_0 - f_\star \\ f_1 - f_\star \end{bmatrix}$. In other words, $G \succcurlyeq 0$ is the Gram matrix of the entries of B :

$$\begin{bmatrix} \|x_0 - x_\star\|^2 & \langle g_0, x_0 - x_\star \rangle & \langle g_1, x_0 - x_\star \rangle \\ \langle g_0, x_0 - x_\star \rangle & \|g_0\|^2 & \langle g_1, g_0 \rangle \\ \langle g_1, x_0 - x_\star \rangle & \langle g_1, g_0 \rangle & \|g_1\|^2 \end{bmatrix} \succcurlyeq 0.$$

A convenient notation is to define basis (row) vectors in $\bar{x}_i, \bar{g}_i \in \mathbb{R}^3$ and $\bar{f}_i \in \mathbb{R}^2$ that allow to “pick” elements in B and \mathbf{f} . That is, we set them to have

$$x_i - x_\star = B \bar{x}_i^\top, \quad g_i = B \bar{g}_i^\top, \quad f_i - f_\star = \bar{f}_i \mathbf{f}. \quad (22)$$

584 More precisely: $\bar{x}_i = \mathbf{e}_1^\top \in \mathbb{R}^3$, $\bar{g}_0 = \mathbf{e}_2^\top \in \mathbb{R}^3$, $\bar{g}_1 = \mathbf{e}_3^\top \in \mathbb{R}^3$, $\bar{x}_1 = \bar{x}_0 - \eta \bar{g}_0$, $\bar{x}_\star = \mathbf{0}_3^\top \in \mathbb{R}^3$
 585 along with $\bar{f}_0 = \mathbf{e}_1^\top \in \mathbb{R}^2$, $\bar{f}_1 = \mathbf{e}_2^\top \in \mathbb{R}^2$, $\bar{f}_\star = \mathbf{0}_2^\top \in \mathbb{R}^2$. We can conveniently rewrite the different
 586 parts of our problem (21) using the notation

$$\begin{aligned}\|x_i - x_\star\|^2 &= \bar{x}_i B^\top B \bar{x}_i^\top = \text{Tr}(\bar{x}_i^\top \bar{x}_i G), \\ \|g_i\|^2 &= \bar{g}_i B^\top B \bar{g}_i^\top = \text{Tr}(\bar{g}_i^\top \bar{g}_i G), \\ \langle g_i, x_j - x_\star \rangle &= \bar{g}_i B^\top B \bar{x}_j^\top = \text{Tr}((\bar{g}_i \odot \bar{x}_j) G)\end{aligned}$$

587 where $\bar{g}_i \odot \bar{x}_j = \frac{1}{2}(\bar{g}_i^\top \bar{x}_j + \bar{x}_j^\top \bar{g}_i)$ (\odot denotes the symmetric outer product—which allows rewriting
 588 everything in terms of traces with symmetric matrices). Now, one can reformulate the different terms
 589 needed to arrive to the desired semidefinite formulation of the problem; let us start with

$$\begin{aligned}\mathcal{V}_P(x_0, \nabla f(x_0); f) &= P_{11} \text{Tr}(\bar{x}_0^\top \bar{x}_0 G) + P_{22} \text{Tr}(\bar{g}_0^\top \bar{g}_0 G) + 2P_{12} \text{Tr}((\bar{x}_0 \odot \bar{g}_0) G) \\ &= \text{Tr}(A_0^\top P A_0 G),\end{aligned}$$

590 with $A_0 := \begin{bmatrix} \bar{x}_0 \\ \bar{g}_0 \end{bmatrix}$. Similarly, we can write $\mathcal{V}_P(x_1, \nabla f(x_1); f) = \text{Tr}(A_1^\top P A_1 G)$ with $A_1 := \begin{bmatrix} \bar{x}_1 \\ \bar{g}_1 \end{bmatrix}$
 591 and also define matrices C_{ij} ($i, j \in \{0, 1, \star\}$) such that $\phi_{ij} = c_{ij} \mathbf{f} + \text{Tr}(C_{ij} G)$ with

$$\begin{aligned}c_{ij} &= \bar{f}_i - \bar{f}_j \\ C_{ij} &= -\bar{g}_j \odot (\bar{x}_i - \bar{x}_j) - \frac{1}{2L}(\bar{g}_i - \bar{g}_j)^\top (\bar{g}_i - \bar{g}_j) \\ &\quad - \frac{\mu}{2(1-\mu/L)}(\bar{x}_i - \bar{x}_j - \frac{1}{L}(\bar{g}_i - \bar{g}_j))^\top (\bar{x}_i - \bar{x}_j - \frac{1}{L}(\bar{g}_i - \bar{g}_j))\end{aligned}$$

592 Ultimately, this allows reformulating (21) conveniently as verifying that (which requires solving a
 593 standard semidefinite problem):

$$\begin{aligned}0 &\geq \sup_{\substack{G \succcurlyeq 0 \\ \mathbf{f}}} \text{Tr}(A_1^\top P A_1 G) - \rho \text{Tr}(A_0^\top P A_0 G) \\ \text{s.t.} \quad &\text{Tr}(C_{ij} G) + \mathbf{f}^\top c_{ij} \geq 0 \quad \forall i, j \in \{0, 1, \star\}.\end{aligned}\tag{23}$$

594 A first consequence is that the “validity” of a given Lyapunov \mathcal{V}_P for a given rate ρ (i.e., satisfying
 595 Eq. (16) or equivalently Eq. (27)) is expressed as the convex problem Eq. (23).

596 Step 2: verifying a rate ρ for the best Lyapunov as a convex problem.

597 To formally find a convenient condition that guarantees the above problem is non-positive, we
 598 consider its standard Lagrangian dual (one can show that there is no duality gap in such problems,
 599 see, e.g., [55]) which amount to verify the existence of some $\nu_{ij} \geq 0$ (dual variables) such that

$$\begin{aligned}0 &\geq \inf_{\nu_{ij} \geq 0} 0 \\ \text{s.t.} \quad &A_1^\top P A_1 - \rho A_0^\top P A_0 - \sum_{i,j \in \{0,1,\star\}} \nu_{ij} C_{ij} \succcurlyeq 0 \\ &\sum_{i,j \in \{0,1,\star\}} \nu_{ij} c_{ij} = 0.\end{aligned}\tag{24}$$

600 The problem has finally reduced to showing that for a given matrix P , the small-sized problem (24)
 601 is feasible.

602 **Key idea:** As this problem is linear in P (when ρ is fixed), we can directly use it to search for a valid
 603 P that verifies the Lyapunov inequality for a given ρ :

$$\text{feasible}_{\substack{P \succcurlyeq 0, \\ \nu_{ij} \geq 0}} \left\{ \begin{array}{l} A_1^\top P A_1 - \rho A_0^\top P A_0 - \sum_{i,j} \nu_{ij} C_{ij} \succcurlyeq 0 \\ \sum_{i,j} \nu_{ij} c_{ij} = 0 \\ \text{Tr}(P) = 1. \end{array} \right. \tag{GD-SDP}$$

604 **Conclusion.** Ultimately, for a given ρ , we have thus expressed the existence for a Lyapunov
 605 verifying a rate ρ as a convex feasibility problem.

606 We can use bisection techniques to solve for the minimal ρ such that the problem Eq. (GD-SDP) is
 607 feasible, to identify numerically both the best rate and the best Lyapunov.

608 A.2 Feasibility problem for EF

609 Using the same ideas as in Appendix A.1, this section states the feasibility problem we solve to
 610 identify Lyapunov functions for Algorithm 2, i.e. EF. In short, this requires adapting the two steps
 611 presented in Appendix A.1 to incorporate compression. Practically speaking, **Step 1** must be adapted
 612 to a slightly larger problem (with a few more states both in the Gram matrix G and in the Lyapunov
 613 candidates to incorporate compression). Then, **Step 2** follows by observing the same effect as before:
 614 the verification step (**Step 1**) amounts to check feasibility of a linear matrix inequality that is linear in
 615 (P, p) , which allows searching for the Lyapunov using binary search over ρ .

616 We recall that by Eq. (14), we have

$$\xi_i^{\text{EF}} = \begin{bmatrix} x_i \\ \nabla f(x_i) \\ \mathcal{C}(e_i + \eta \nabla f(x_i)) \\ e_i \end{bmatrix}.$$

617 The state space thus has dimension 4,

$$\mathbb{V}_4 = \{(P, p) \in \mathbb{S}_+^4 \times \mathbb{R}^+ : \text{Tr}(P) + p = 1\}.$$

618 For any $(P, p) \in \mathbb{V}_4$ we thus consider $\mathcal{V}_{(P,p)}$ Lyapunov functions of the form:

$$\mathcal{V}_{(P,p)}(\xi^{\text{EF}}, x; f) = (\xi^{\text{EF}} - \xi_\star^{\text{EF}})^\top (P \otimes I_d)(\xi^{\text{EF}} - \xi_\star^{\text{EF}}) + p(f(x) - f_\star). \quad (25)$$

619 (where we impose $\text{Tr}(P) + p = 1$ to avoid the trivial solution $(P, p) = 0$). Similarly, to (18), we
 620 define here

$$\sigma_\rho^{\text{EF}}(\xi_1^{\text{EF}}, \xi_0^{\text{EF}}; (P, p); f) := \mathcal{V}_{(P,p)}(\xi_1^{\text{EF}}, x_1; f) - \rho \mathcal{V}_{(P,p)}(\xi_0^{\text{EF}}, x_0; f). \quad (26)$$

621 Again, we say that for $(P, p) \in \mathbb{V}_4$, a Lyapunov $\mathcal{V}_{(P,p)}$ satisfies rate ρ for the iterates of EF, if we
 622 have for σ_ρ^{EF} given by (26), that

$$\begin{aligned} 0 &\geq \sup_{\substack{d \in \mathbb{N} \\ f \in \mathcal{F}_{\mu, L} \\ x_0, x_\star \in \mathbb{R}^d}} \sigma_\rho^{\text{EF}}(\xi_1^{\text{EF}}, \xi_0^{\text{EF}}; (P, p); f) \\ \text{s.t. } & (x_1, \xi_1^{\text{EF}}) = \text{EF}(x_0, \xi_0^{\text{EF}}; f) \\ & \nabla f(x_\star) = 0 \end{aligned} \quad (27)$$

623 Formally, we prove the following lemma.

624 **Lemma 1.** For a given $\rho > 0$, the existence of a (nontrivial) Lyapunov $\mathcal{V}_{(P,p)}$ (see Eq. (25)) satisfying
 625 rate ρ , for a $(P, p) \in \mathbb{V}_\ell$, for EF algorithm, can be written as

$$\text{feasible} \left\{ \begin{array}{l} 0 \succcurlyeq \Delta V_P(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} M_{ij} + \sum_{i \in \{0,1\}} \gamma_i \cdot C_i^{\text{EF}} \\ 0 \geq \Delta v_p(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} m_{ij} \\ 0 \preccurlyeq P \\ 0 \leq p \\ 1 = \text{Tr}(P) + p \end{array} \right. \quad (\text{EF-SDP})$$

626 for matrices $(M_{ij})_{i,j \in \{0,1,\star\}}$ given below in Eq. (34), $(C_i)_{i \in \{0,1\}}$ given below in Eq. (36), and
 627 $\Delta V_P(\rho), \Delta v_p(\rho)$ given below in Eqs. (31) and (32).

628 *Proof.* The proof is decomposed into several steps that correspond to adapt the technical ingredients
 629 from Appendix A.1.

630 **“Picking” vectors.** We first introduce notations, similar to Eq. (22). We begin by defining the
 631 following basis vectors $\bar{x}_i, \bar{g}_i, \bar{c}_i, \bar{e}_i \in \mathbb{R}^8$ such that:

$$\bar{x}_i := \mathbf{e}_{i+1}^\top, \quad \bar{g}_i := \mathbf{e}_{i+3}^\top, \quad \bar{c}_i := \mathbf{e}_{i+5}^\top, \quad \bar{e}_i := \mathbf{e}_{i+7}^\top, \quad i \in \{0, 1\}, \quad (28)$$

632 where \mathbf{e}_i is the i -th basis vector in dimension 8. Similarly, let $\bar{f}_i \in \mathbb{R}^2$ such that

$$\bar{f}_i := \mathbf{e}_i^\top, \quad i \in \{0, 1\}, \quad (29)$$

633 where \mathbf{e}_i is the i -th basis vector in dimension 2.

634 As in the example of the previous section (see in particular Eq. (22)), the point of these vectors is to
 635 “pick” the relevant parts of the Gram matrix information (which is larger in this section). This lets us
 636 encode our interpolation conditions for our feasibility problem cleanly.

637 We also define row vectors that correspond to the fixed-point as:

$$\bar{x}_\star := \mathbf{0}_8^\top, \quad \bar{g}_\star := \mathbf{0}_8^\top, \quad \bar{c}_\star := \mathbf{0}_8^\top, \quad \bar{e}_\star := \mathbf{0}_8^\top, \\ \bar{f}_\star := \mathbf{0}_2^\top.$$

638 Now, we define our method in terms of our basis vectors:

$$\bar{x}_1 = \bar{x}_0 - \bar{c}_0, \\ \bar{e}_1 = \bar{e}_0 + \eta \bar{g}_0 - \bar{c}_0. \quad (30)$$

639 **Translating Eq. (26) with those notations.** First, we encode the decrease in the linear and quadratic
 640 terms in Eq. (26) as

$$\Delta V_P(\rho) := \begin{bmatrix} \bar{x}_1 - \bar{x}_\star \\ \bar{g}_1 - \bar{g}_\star \\ \bar{c}_1 - \bar{c}_\star \\ \bar{e}_1 - \bar{e}_\star \end{bmatrix}^\top P \begin{bmatrix} \bar{x}_1 - \bar{x}_\star \\ \bar{g}_1 - \bar{g}_\star \\ \bar{c}_1 - \bar{c}_\star \\ \bar{e}_1 - \bar{e}_\star \end{bmatrix} - \rho \begin{bmatrix} \bar{x}_0 - \bar{x}_\star \\ \bar{g}_0 - \bar{g}_\star \\ \bar{c}_0 - \bar{c}_\star \\ \bar{e}_0 - \bar{e}_\star \end{bmatrix}^\top P \begin{bmatrix} \bar{x}_0 - \bar{x}_\star \\ \bar{g}_0 - \bar{g}_\star \\ \bar{c}_0 - \bar{c}_\star \\ \bar{e}_0 - \bar{e}_\star \end{bmatrix}, \quad (31)$$

$$\Delta v_p(\rho) := p(\bar{f}_1 - \bar{f}_\star) - \rho \cdot p(\bar{f}_0 - \bar{f}_\star), \quad (32)$$

642 where $\rho > 0$ is the contraction factor to be verified.

643 With such notations, we have that:

$$\sigma_\rho^{\text{EF}}(\xi_1^{\text{EF}}, \xi_0^{\text{EF}}; (P, p); f) = \text{Tr} \left((\Delta V_P(\rho)) G^{\text{EF}} \right) + (\Delta v_p(\rho))^\top F^{\text{EF}} \quad (33)$$

where $G^{\text{EF}} = (B^{\text{EF}})^\top B^{\text{EF}}$ is the Gram matrix of vectors

$$B^{\text{EF}} = [x_0, x_1, \nabla f(x_0), \nabla f(x_1), \mathcal{C}(e_0 + \eta \nabla f(x_0)), \mathcal{C}(e_1 + \eta \nabla f(x_1)), e_0, e_1],$$

644 and $F^{\text{EF}} = (f(x_0), f(x_1))$.

645 **Interpolation conditions for class $\mathcal{F}_{(\mu, L)}$.** We now need to introduce interpolation conditions for
 646 Assumptions 2 and 3, which can be written as

$$m_{ij} := (L - \mu)(\bar{f}_i - \bar{f}_j)^\top, \quad M_{ij} := \frac{1}{2} \begin{bmatrix} \bar{x}_i \\ \bar{x}_j \\ \bar{g}_i \\ \bar{g}_j \end{bmatrix}^\top M \begin{bmatrix} \bar{x}_j \\ \bar{x}_j \\ \bar{g}_j \\ \bar{g}_j \end{bmatrix}, \quad (34)$$

647 where $i, j \in \{0, 1, \star\}$ and M is defined in [47, Eq. (11)].

648 **Interpolation condition for the compression.** We need to introduce a new interpolation condition,
 649 encoding the fact that we are using a contractive compressor. This can be written as

$$C^{\text{EF}} = \begin{bmatrix} \eta^2(1 - \epsilon) & -\eta & \eta(1 - \epsilon) \\ -\eta & 1 & -\eta \\ \eta(1 - \epsilon) & -\eta & 1 - \epsilon \end{bmatrix}, \quad (35)$$

650 where we use the notation

$$C_i^{\text{EF}} := \begin{bmatrix} \bar{g}_i \\ \bar{c}_i \end{bmatrix}^\top C^{\text{EF}} \begin{bmatrix} \bar{g}_i \\ \bar{c}_i \end{bmatrix}. \quad (36)$$

651 Overall, following the same approach as described in Step 1 and Step 2 in appendix A.1, and
 652 leveraging Eqs. (32), (34), (36) and (45) We can now state the feasibility problem to find Lyapunov
 653 functions for Algorithm 2 as

$$\text{feasible} \begin{cases} 0 \succ \Delta V_P(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} M_{ij} + \sum_{i \in \{0,1\}} \gamma_i \cdot C_i^{\text{EF}} \\ 0 \geq \Delta v_p(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} m_{ij} \\ 0 \preceq P \\ 0 \leq p \\ 1 = \text{Tr}(P) + p \end{cases} \quad (\text{EF-SDP})$$

$P \in \mathbb{S}^4,$
 $p \in \mathbb{R},$
 $\nu_{ij} \geq 0,$
 $\gamma_i \geq 0$

654 where the constraint $\text{Tr}(P) + p = 1$ ensures normalization as any valid nontrivial (i.e., avoiding
 655 $(P, p) = 0$) Lyapunov function can be normalized to satisfy this property.

656

□

657 A.3 Feasibility problem for EF²¹

658 As in the previous section, one can now adapt the same ideas as in Appendix A.1 and Appendix A.2
 659 to EF²¹. This section states the feasibility problem we solve to identify Lyapunov functions in this
 660 context.

661 We follow a parallel line of derivation. By Eq. (14), we have

$$\xi_k^{\text{EF}^{21}} = \begin{bmatrix} x_k \\ \nabla f(x_k) \\ d_k \end{bmatrix}, \quad (37)$$

662 The state space thus has dimension 3, let

$$\mathbb{V}_3 = \{(P, p) \in \mathbb{S}_+^3 \times \mathbb{R}^+ : \text{Tr}(P) + p = 1\}.$$

663 For any $(P, p) \in \mathbb{V}_3$ we thus consider $\mathcal{V}_{(P,p)}$ Lyapunov functions of the form:

$$\mathcal{V}_{(P,p)}(\xi^{\text{EF}^{21}}, x; f) = (\xi^{\text{EF}^{21}} - \xi_\star^{\text{EF}^{21}})^\top (P \otimes I_d)(\xi^{\text{EF}^{21}} - \xi_\star^{\text{EF}^{21}}) + p(f(x) - f_\star). \quad (38)$$

664 (where we impose $\text{Tr}(P) + p = 1$ to avoid the trivial solution $(P, p) = 0$).

665 Similarly, to Eqs. (18) and (26), we define here

$$\sigma_\rho^{\text{EF}^{21}}(\xi_1^{\text{EF}^{21}}, \xi_0^{\text{EF}^{21}}; (P, p); f) := \mathcal{V}_{(P,p)}(\xi_1^{\text{EF}^{21}}, x_1; f) - \rho \mathcal{V}_{(P,p)}(\xi_0^{\text{EF}^{21}}, x_0; f). \quad (39)$$

666 Again, we say that for $(P, p) \in \mathbb{V}_4$, a Lyapunov $\mathcal{V}_{(P,p)}$ satisfies rate ρ for the iterates of EF²¹, if we
 667 have for $\sigma_\rho^{\text{EF}^{21}}$ given by (39), that

$$\begin{aligned} 0 \geq \sup_{\substack{d \in \mathbb{N} \\ f \in \mathcal{F}_{\mu,L} \\ x_0, x_\star \in \mathbb{R}^d}} \sigma_\rho^{\text{EF}^{21}}(\xi_1^{\text{EF}^{21}}, \xi_0^{\text{EF}^{21}}; (P, p); f) \\ \text{s.t. } (x_1, \xi_1^{\text{EF}^{21}}) = \text{EF}^{21}(x_0, \xi_0^{\text{EF}^{21}}; f) \\ \nabla f(x_\star) = 0 \end{aligned} \quad (40)$$

668 Formally, we prove the following lemma.

669 **Lemma 2.** For a given $\rho > 0$, the existence of a (nontrivial) Lyapunov $\mathcal{V}_{(P,p)}$ (see Eq. (38)) satisfying
 670 rate ρ , for a $(P, p) \in \mathbb{V}_\ell$, for EF²¹ algorithm, can be written as

$$\text{feasible} \left\{ \begin{array}{l} 0 \succ \Delta V_P(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} M_{ij} + \sum_{i \in \{0,1\}} \gamma_i \cdot C_i^{EF} \\ 0 \geq \Delta v_p(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} m_{ij} \\ 0 \preceq P \\ 0 \leq p \\ 1 = \text{Tr}(P) + p \end{array} \right. \quad (\text{EF}^{21}\text{-SDP})$$

$P \in \mathbb{S}^3, p \in \mathbb{R}, \nu_{ij} \geq 0, \gamma_i \geq 0$

671 for matrices $(M_{ij})_{i,j \in \{0,1,\star\}}$ given above in Eq. (34), $(C_i)_{i \in \{0,1\}}$ given below in Eq. (44), and
 672 $\Delta V_P(\rho), \Delta v_p(\rho)$ given below in Eqs. (45) and (46).

673 We begin by changing our basis vectors to

$$\bar{x}_i := \mathbf{e}_{i+1}^\top, \bar{g}_i := \mathbf{e}_{i+3}^\top, \bar{c}_i := \mathbf{e}_{i+5}^\top, \bar{d}_i := \mathbf{e}_{i+7}^\top, \bar{f}_i := \mathbf{e}_i^\top, \quad i \in \{0, 1\} \quad (41)$$

674 where $\bar{x}_i, \bar{g}_i, \bar{c}_i, \bar{d}_i \in \mathbb{R}^8$, and $\bar{f}_i \in \mathbb{R}^2$. Similarly, we define the row vectors corresponding to the
 675 fixed-point as

$$\bar{x}_\star := \mathbf{0}_8^\top, \bar{g}_\star := \mathbf{0}_8^\top, \bar{c}_\star := \mathbf{0}_8^\top, \bar{d}_\star := \mathbf{0}_8^\top, \bar{f}_\star := \mathbf{0}_2^\top, \quad (42)$$

676 We define our method using these basis vectors as

$$\begin{aligned} \bar{x}_1 &= \bar{x}_0 + \eta \cdot \bar{d}_0, \\ \bar{d}_1 &= \bar{d}_0 + \bar{c}_0. \end{aligned} \quad (43)$$

677 The only difference in the interpolation conditions is that we are compressing a different quantity
 678 now, which we can encode using

$$C^{\text{EF}^{21}} := \begin{bmatrix} \bar{g} \\ \bar{c} \\ \bar{d} \end{bmatrix}^\top \begin{bmatrix} \eta^2(1-\epsilon) & -1 & \epsilon-1 \\ -1 & 1 & 1 \\ \epsilon-1 & 1 & 1-\epsilon \end{bmatrix} \begin{bmatrix} \bar{g} \\ \bar{c} \\ \bar{d} \end{bmatrix}. \quad (44)$$

679 where we only need a single matrix because the compression operator acts on a mixture of the current
 680 and next state.

681 Next, we encode the linear and quadratic terms in exactly the same way as we did for EF, except
 682 with the state variables

$$\Delta V_P(\rho) := \begin{bmatrix} \bar{x}_1 - \bar{x}_\star \\ \bar{g}_1 - \bar{g}_\star \\ \bar{d}_1 - \bar{d}_\star \end{bmatrix}^\top P \begin{bmatrix} \bar{x}_1 - \bar{x}_\star \\ \bar{g}_1 - \bar{g}_\star \\ \bar{d}_1 - \bar{d}_\star \end{bmatrix} - \rho \begin{bmatrix} \bar{x}_0 - \bar{x}_\star \\ \bar{g}_0 - \bar{g}_\star \\ \bar{d}_0 - \bar{d}_\star \end{bmatrix}^\top P \begin{bmatrix} \bar{x}_0 - \bar{x}_\star \\ \bar{g}_0 - \bar{g}_\star \\ \bar{d}_0 - \bar{d}_\star \end{bmatrix}. \quad (45)$$

683

$$\Delta v_p(\rho) := p(\bar{f}_1 - \bar{f}_\star) - \rho \cdot p(\bar{f}_0 - \bar{f}_\star), \quad (46)$$

684 Finally, we end up with the following SDP:

$$\text{feasible} \left\{ \begin{array}{l} 0 \succ \Delta V_P(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} M_{ij} + \gamma \cdot C^{\text{EF}^{21}} \\ 0 \geq \Delta v_p(\rho) + \sum_{i,j \in \{0,1,\star\}} \nu_{ij} m_{ij} \\ 0 \preceq P \\ 0 \leq p \\ 1 = \text{Tr}(P) + p \end{array} \right. \quad (\text{EF}^{21}\text{-SDP})$$

$P \in \mathbb{S}^3, p \in \mathbb{R}, \nu_{ij} \geq 0, \gamma \geq 0$

B Missing proofs

This section contains the proofs of Theorems 1 and 2. Those proofs were obtained from numerical inspiration using Lyapunov search procedure presented in Appendices A.2 and A.3. The resulting proofs are remarkably compact, and the main technical step consists of verifying an algebraic reformulation by hand. For ease of verification, we provide the reader with symbolic computation notebooks formally performing those reformulations.

B.1 Proof of Theorem 1

Theorem 1. Consider running Algorithm 2, i.e., EF, with a compression operator \mathcal{C} satisfying Assumption 1 for some $\epsilon \in [0, 1]$ on any function satisfying Assumptions 2, and 3. Let the step size be given by

$$\eta^* = \left(\frac{2}{L + \mu} \right) \cdot \left(\frac{1 - \sqrt{\epsilon}}{1 + \sqrt{\epsilon}} \right). \quad (8)$$

Then, we have that

$$\rho^*(\text{EF}_{\eta^*}) = \sqrt{\epsilon} + \frac{1}{4}(1 + \sqrt{\epsilon})(L - \mu)\lambda, \quad (9)$$

where

$$\lambda := \frac{\eta^*}{L + \mu} \left[(1 - \sqrt{\epsilon})(L - \mu) + (1 + \sqrt{\epsilon})\sqrt{(L - \mu)^2 + 16L\mu\frac{\sqrt{\epsilon}}{(1 + \sqrt{\epsilon})^2}} \right]. \quad (10)$$

A Lyapunov function achieving the rate in (9), with ξ^{EF} defined in (14), is given by

$$\mathcal{V}(\xi^{\text{EF}}, x; f) := \|x - x_\star\|^2 - 2(x - x_\star)^\top e + \left(1 + \frac{1}{\sqrt{\epsilon}}\right) \cdot \|e\|^2 = \|x - x_\star - e\|^2 + \frac{1}{\sqrt{\epsilon}}\|e\|^2, \quad (11)$$

Finally, the step size in (8) is worst-case optimal for EF: $\forall \eta \geq 0$, we have $\rho^*(\text{EF}_\eta) \geq \rho^*(\text{EF}_{\eta^*})$.

Proof. We begin by proving the rate given in (9) for our Lyapunov function. Consider the following inequalities, and associated with each of them the assigned multiplier (which corresponds to “symbolic” values for the variables of Eq. (EF-SDP))

$$\begin{aligned} I_{\mathcal{F}_{\mu,L}}^{(1)} &:= f(x_k) - f_\star - \nabla f(x_k)^\top (x_k - x_\star) + \frac{1}{2L}\|\nabla f(x_k)\|^2 && : \lambda \\ &+ \frac{\mu}{2(1 - \mu/L)}\|x_k - x_\star - \frac{1}{L}\nabla f(x_k)\|^2 \leq 0, && \\ I_{\mathcal{F}_{\mu,L}}^{(2)} &:= f_\star - f(x_k) + \frac{1}{2L}\|\nabla f(x_k)\|^2 + \frac{\mu}{2(1 - \mu/L)}\|x_k - x_\star - \frac{1}{L}\nabla f(x_k)\|^2 \leq 0, && : \lambda \\ I_C &:= \|e_{k+1}\|^2 - \epsilon\|e_k + \eta\nabla f(x_k)\|^2 \leq 0, && : \nu \end{aligned}$$

where λ is defined in (10), and $\nu := \frac{1}{\sqrt{\epsilon}}$.

Summing these inequalities with their multipliers, plugging in the update rules for x_{k+1} and e_{k+1} , and using ρ to denote the contraction factor we got in (9), we can rewrite the resulting inequality as:

$$\rho \cdot \mathcal{V}(x_k, e_k) \geq \mathcal{V}(x_{k+1}, e_{k+1}) + a \cdot \|e_k\| - \frac{\rho - 1}{a}(x_k - x_\star) + \frac{2(\sqrt{\epsilon} - 1)}{a(L + \mu)}g_k\|^2, \quad (47)$$

where

$$a := (\rho - \sqrt{\epsilon}) \cdot \left(\frac{1 + \sqrt{\epsilon}}{\sqrt{\epsilon}} \right). \quad (48)$$

The statement now follows from the simple inequality $\rho > \sqrt{\epsilon}$.

We now prove that the announced rate is tight. Consider the one-dimensional quadratic function

$$f_\mu(x) = \frac{\mu}{2}x^2. \quad (49)$$

The proof strategy used here is to show that the contraction for our Lyapunov function asymptotically matches the convergence rate announced in Theorem 1. We begin by fully exploiting Assumption 1 and set

$$c_k := \mathcal{C}(e_k + \eta\nabla f(x_k)) = (1 + \sqrt{\epsilon}) \cdot (\eta\nabla f(x_k) + e_k) \quad (50)$$

711 We can now rewrite the update rule for x_{k+1} and x_{k+2} to get an expression for e_k and e_{k+1} respec-
712 tively:

$$e_k = \frac{1 - \eta\mu(1 + \sqrt{\epsilon})}{1 + \sqrt{\epsilon}} x_k - \frac{x_{k+1}}{1 + \sqrt{\epsilon}}, \quad e_{k+1} = \frac{1 - \eta\mu(1 + \sqrt{\epsilon})}{1 + \sqrt{\epsilon}} x_{k+1} - \frac{x_{k+2}}{1 + \sqrt{\epsilon}}, \quad (51)$$

713 after which we use the update rule for e_{k+1} of Algorithm 2 to get a second-order recurrence relation
714 for the sequence $\{x_k\}_{k=1}^\infty$:

$$\sqrt{\epsilon}x_k = x_{k+2} - (1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu)) \cdot x_{k+1} \quad (52)$$

715 The solution to this recurrence relation is given by the roots of the characteristic equation, and after
716 plugging in the initial conditions, we get

$$x_k = \frac{1}{T} \cdot (1 - \eta\mu + \sqrt{\epsilon}(1 - \eta\mu) + T)(1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu) + T)^k - \frac{1}{T} \cdot (1 - \eta\mu + \sqrt{\epsilon}(1 - \eta\mu) - T)(1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu) - T)^k, \quad (53)$$

717 where $T := \sqrt{4\sqrt{\epsilon} + (1 - \eta\mu + \sqrt{\epsilon}(1 + \eta\mu))^2}$. Note that for

$$\eta < \left(\frac{1}{\mu}\right) \cdot \left(\frac{1 - \sqrt{\epsilon}}{1 + \sqrt{\epsilon}}\right), \quad (54)$$

718 which is strictly larger than the step size given in (8), the above expression is dominated by the first
719 term in the limit $k \rightarrow \infty$. If we plug in the resulting asymptotic expression for x_k into the definition
720 of e_k , and plug the resulting points into our Lyapunov function we get

$$\frac{\mathcal{V}(x_{k+1}, e_{k+1})}{\mathcal{V}(x_k, e_k)} \xrightarrow{k \rightarrow \infty} \frac{1}{4}(1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu) + T)^2 \quad (55)$$

721 which, after plugging in the step size given in (8), is exactly the convergence rate announced in
722 Theorem 1. The fact that our Lyapunov function is tight now follows from the remark made in
723 Section 3.4.

724 Finally, we prove that the step size given in (8) is the optimal step size for our Lyapunov function.
725 Note that the contraction factor

$$\rho(\eta) := \frac{1}{4}(1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu) + T)^2, \quad (56)$$

726 that becomes the dominant term in the limit $k \rightarrow \infty$ of (55) is strictly decreasing in η . This is
727 immediate from inspecting the sign of the derivative of the expression with respect to η :

$$\frac{d\rho(\eta)}{d\eta} = -\mu(1 + \sqrt{\epsilon}) \frac{(1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu) + T)^2}{2T}. \quad (57)$$

728 The rest of the proof now follows from instead considering the quadratic given by

$$f_L(x) := \frac{L}{2}x^2, \quad (58)$$

729 and repeating all the arguments stated above, except we instead consider step sizes

$$\eta > \left(\frac{1}{L}\right) \cdot \left(\frac{1 - \sqrt{\epsilon}}{1 + \sqrt{\epsilon}}\right), \quad (59)$$

730 and show that the contraction for our Lyapunov function is strictly decreasing for these step sizes.
731 The argument now follows from the fact that the contraction factor on both of these quadratics are the
732 same for the step size given in (8). \square

733 B.2 Proof of Theorem 2

734 **Theorem 2.** Consider running Algorithm 3 with a compression operator satisfying Assumption 1
735 for some $\epsilon \in [0, 1]$ on any function satisfying Assumptions 2, and 3. Let the step size be given by η^*
736 in (8). Then, we have that

$$\rho^*(\text{EF}^{21}_{\eta^*}) = \left[\sqrt{\epsilon} + \frac{1}{4}(1 + \sqrt{\epsilon})(L - \mu)\lambda \right], \quad (12)$$

737 where λ is given by (10). A Lyapunov function achieving the rate in (12) is given by

$$\mathcal{V}(\xi^{\text{EF}^{21}}, x; f) := (1 + \sqrt{\epsilon}) \cdot \|g\|^2 - 2g^\top d + \|d\|^2 = \|g - d\|^2 + \sqrt{\epsilon} \cdot \|d\|^2. \quad (13)$$

738 Finally, the step size η^* is worst-case optimal for this algorithm.

739 *Proof.* We begin by proving the rate given in (12) for our Lyapunov function. Consider the following
740 inequalities, and associated with each of them the assigned multiplier:

$$\begin{aligned} I_{\mathcal{F}_{\mu,L}}^{(1)} &:= f(x_k) - f(x_{k+1}) + \frac{\|\nabla f(x_{k+1}) - \nabla f(x_k)\|^2}{2L} + \nabla f(x_k)^\top (x_{k+1} - x_k) && : \lambda \\ &\quad + \frac{\mu}{2(1 - \mu/L)} \|x_k - x_{k+1} - \frac{1}{L}(\nabla f(x_k) - \nabla f(x_{k+1}))\|^2 \leq 0, \\ I_{\mathcal{F}_{\mu,L}}^{(2)} &:= f(x_{k+1}) - f(x_k) + \frac{\|\nabla f(x_k) - \nabla f(x_{k+1})\|^2}{2L} + \nabla f(x_{k+1})^\top (x_k - x_{k+1}) && : \lambda \\ &\quad + \frac{\mu}{2(1 - \mu/L)} \|x_{k+1} - x_k - \frac{1}{L}(\nabla f(x_{k+1}) - \nabla f(x_k))\|^2 \leq 0, \\ I_{\mathcal{C}} &:= \|\nabla f(x_{k+1}) - d_k - \mathcal{C}(\nabla f(x_{k+1}) - d_k)\|^2 - \epsilon \|\nabla f(x_{k+1}) - d_k\|^2 \leq 0, && : \nu \end{aligned}$$

741 where λ is defined in (10), and $\nu := 1$.

742 Summing these inequalities with their multipliers, plugging in the update rules for x_{k+1} and d_{k+1} ,
743 and using ρ to denote the contraction factor we got in (12), we can rewrite the resulting inequality as:

$$\rho \cdot \mathcal{V}(g_k, d_k) \geq \mathcal{V}(g_{k+1}, d_{k+1}) + a \cdot \|d_k + \frac{1}{a}((\epsilon + b)g_{k+1} - (\rho + b)g_k)\|^2, \quad (60)$$

744 where

$$b := \frac{\lambda}{L - \mu} \cdot \left(\frac{1 - \sqrt{\epsilon}}{1 + \sqrt{\epsilon}} \right). \quad (61)$$

745 and

$$a := \rho - \epsilon + 2\eta\lambda(1 - \sqrt{\epsilon}) \frac{L\mu}{L - \mu}. \quad (62)$$

746 The statement now follows from plugging in the value of our multipliers and checking the sign.

747 We now prove that the announced rate is tight. Consider the one-dimensional quadratic function

$$f_\mu(x) = \frac{\mu}{2}x^2. \quad (63)$$

748 The proof strategy used here is to show that the contraction for our Lyapunov function asymptotically
749 matches the convergence rate announced in Theorem 2. We begin by fully exploiting Assumption 1
750 and set

$$c_k := \mathcal{C}(\nabla f(x_{k+1}) - d_k) = (1 + \sqrt{\epsilon}) \cdot (\nabla f(x_{k+1}) - d_k) \quad (64)$$

751 We can now rewrite the update rule for x_{k+1} and x_{k+2} to get an expression for d_k and d_{k+1}
752 respectively:

$$d_k = \frac{x_k - x_{k+1}}{\eta}, \quad d_{k+1} = \frac{x_{k+1} - x_{k+2}}{\eta}, \quad (65)$$

753 after which we use the update rule for d_{k+1} of Algorithm 3 to get a second-order recurrence relation
754 for the sequence $\{x_k\}_{k=1}^\infty$:

$$\sqrt{\epsilon}x_k = x_{k+2} - (1 - \eta\mu - \sqrt{\epsilon}(1 + \eta\mu)) \cdot x_{k+1} \quad (66)$$

755 Note that this is the exact same recurrence relation as in (9), which means we can reuse the expression
756 in (53) and the argument that follows. The proof now follows from the definition of our Lyapunov
757 function in (13). The optimality of our Lyapunov function similarly follows from the same argument
758 given in the remark made in Section 4.

759 Lastly, the proof of optimality of our step size follows directly from the same argument given in the
760 proof of Theorem 1. \square

C Additional numerical results

All subsections include details on how the corresponding results were computed. This section is organized as follows:

- Appendix C.1 presents additional performance plots for all methods and explains how the plots in the main paper were generated.
- Appendix C.2 provides illustrations demonstrating that our Lyapunov functions remain tight over multiple iterations.
- Appendix C.3 includes additional tables further confirming the tightness of our Lyapunov functions.
- Appendix C.4 shows plots illustrating the optimality of our step size.

All experiments were run on a MacBook Pro with an M4 Max processor. While none of the experiments are computationally intensive by today’s standards, they can be scaled up by increasing the resolution of the grid of values for η and ϵ to generate finer plots.

C.1 Performance plots

This section presents the worst-case performance of all methods studied in this work, plotted as a function of the step size η and the compression parameter ϵ .

All contour plots were evaluated over a grid with $\epsilon \in [0.01, 0.99]$, and $\eta \in [0.01, \frac{2}{L+\mu_\star}]$, where μ_\star is the smallest μ specified in the caption of each figure (except for Figure 4, where it is set to 0.1). Each axis was discretized with a resolution of 100 points.

To generate each non-cyclic point, we used the following procedure:

1. For each method, we computed the optimal Lyapunov function (without additional constraints) via bisection on the contraction factor ρ , up to a precision of 10^{-6} .
2. Using the resulting Lyapunov function, we then computed the worst-case contraction factor using PEPit and the MOSEK solver [56].

We adopt this two-step approach because the feasibility problems used to compute the Lyapunov functions suffer from numerical instability. By evaluating the contraction factor separately using PEPit, we can ensure that the reported value is an upper bound on the true contraction factor, up to solver tolerance.

To identify the area of non-convergence in the plots, we check whether a cycle exists for each pair of η and ϵ . This is done by following the procedure outlined in Goujaud et al. [63]: we compute the worst-case performance of the metric $-\|x_k - x_0\|^2$ for CGD, $-\|x_k - x_0\|^2 - \|e_k - e_0\|^2$ for EF, and $-\|x_k - x_0\|^2 - \|d_k - d_0\|^2$ for EF²¹. If this value falls below a threshold (set to 10^{-3}), we conclude that a cycle is present. In our experiments, successfully identified cycles of length 2 for all methods, and these matched precisely with the regions of the contour plots where $\rho > 1$.

C.2 Multi-step Lyapunov analysis

In this section, we show that our simple Lyapunov functions achieve the claimed convergence rate over multiple iterations. Specifically, we use PEPit to compute the contraction factor achieved by the Lyapunov function after k iterations and compare it to the theoretical rate ρ^k , where ρ is the single-step contraction factor. The exact match between these quantities confirms that our single-step analysis accurately characterizes the worst-case performance over multiple iterations on these Lyapunov functions.

C.3 Lyapunov function class tightness

In this section, we show that for various conditioning numbers κ , our Lyapunov functions for EF and EF²¹ are tight with respect to our class of Lyapunov functions, when using optimal step sizes. We remark that our Lyapunov functions are actually tight for many step size settings, but notably not step sizes which are larger than the optimal step size.

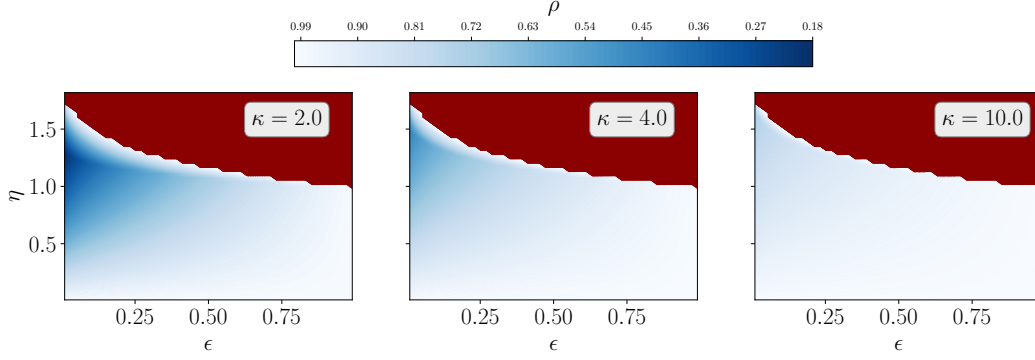


Figure 3: Contour plot showing the performance of CGD as a function of step size η and compression parameter ϵ , with regions of non-convergence marked in red. The regions of non-convergence were identified using PEPit by finding cycles of length 2. Each column corresponds to $\mu = 0.5, 0.25, 0.1$, with $L = 1.0$ fixed across all plots.

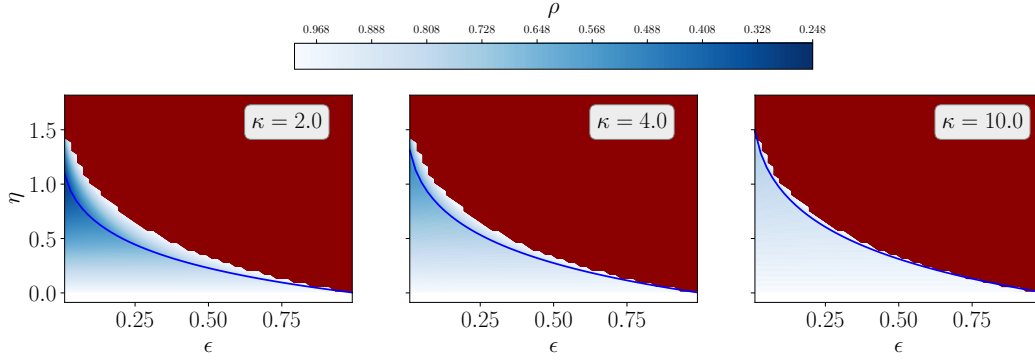


Figure 4: Contour plot showing the performance of EF as a function of step size η and compression parameter ϵ , with regions of non-convergence marked in red. The regions of non-convergence were identified using PEPit by finding cycles of length 2. Each column corresponds to $\mu = 0.5, 0.25, 0.1$, with $L = 1.0$ fixed across all plots. The optimal step size setting for a given ϵ is marked in blue.

807 The tables were generated by evaluating a variety of ϵ and η values (as specified in the captions),
808 and computing the maximum absolute difference between the contraction factor achieved by our
809 Lyapunov function and that of the optimal Lyapunov function in our class. All contraction factors
810 were computed using PEPit, and the procedure for the unconstrained Lyapunov functions follows the
811 procedure outlined in Appendix C.1. Points where either Lyapunov function yields a contraction
812 factor greater than 1 were excluded from the computation of the maximum absolute difference.

	$\kappa = 2.0$	$\kappa = 4.0$	$\kappa = 10.0$
Absolute error	1.17e-07	3.96e-08	4.05e-05

Table 3: Maximum absolute difference in contraction factor for EF when comparing the general Lyapunov function—constructed using any combination of state terms specified in Subsection 3.1—to the simplified Lyapunov function defined in Theorem 1. The results are computed over a line with $\epsilon \in [0.01, 0.99]$ and with η set to the optimal step size for $L = 1$, and $\mu = 0.5, 0.25, 0.1$.

813 C.4 Step size comparison

814 In this section, we compare the theoretically optimal step sizes we propose for our methods with
815 empirically optimal step sizes determined through numerical experiments. To compute the empirical

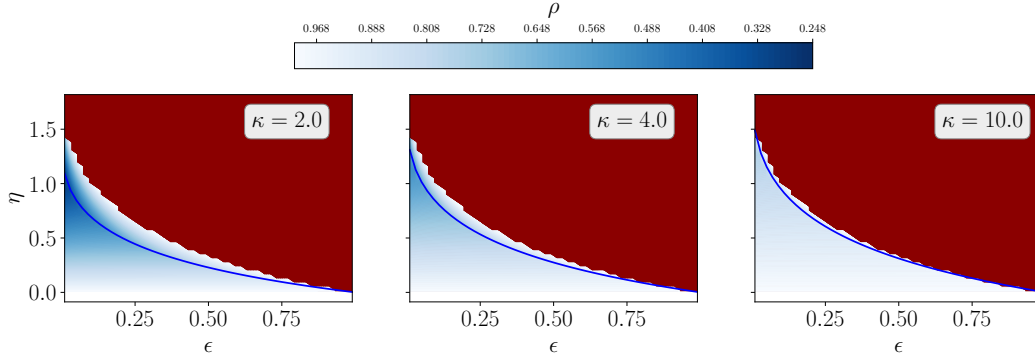


Figure 5: Contour plot showing the performance of EF^{21} as a function of step size η and compression parameter ϵ , with regions of non-convergence marked in red. The regions of non-convergence were identified using PEPit by finding cycles of length 2. Each column corresponds to $\mu = 0.5, 0.25, 0.1$, with $L = 1.0$ fixed across all plots. The optimal step size setting for a given ϵ is marked in blue.

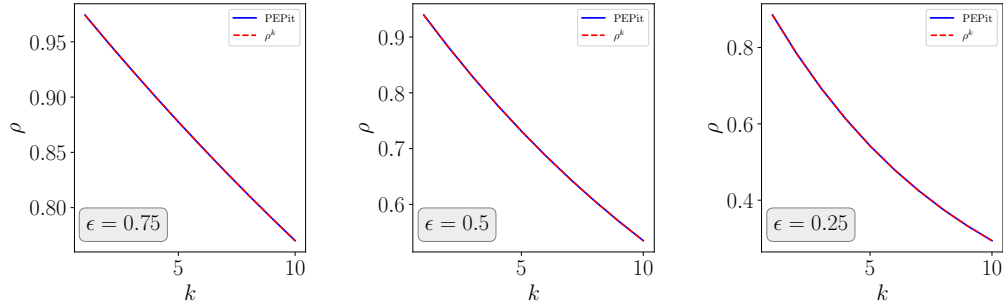


Figure 6: Multi-step Lyapunov analysis for EF, computed using PEPit. The blue line shows the contraction factor achieved by the Lyapunov function after k iterations, while the red line represents the theoretical rate ρ^k , where ρ is the single-step contraction factor. Each column corresponds to a different value of $\epsilon = 0.75, 0.5, 0.25$, with $L = 1.0$ and $\mu = 0.1$ fixed across all plots.

816 optima, we evaluate a grid of η and ϵ values and select the step size that minimizes the contraction
817 factor achieved by our simplified Lyapunov functions.

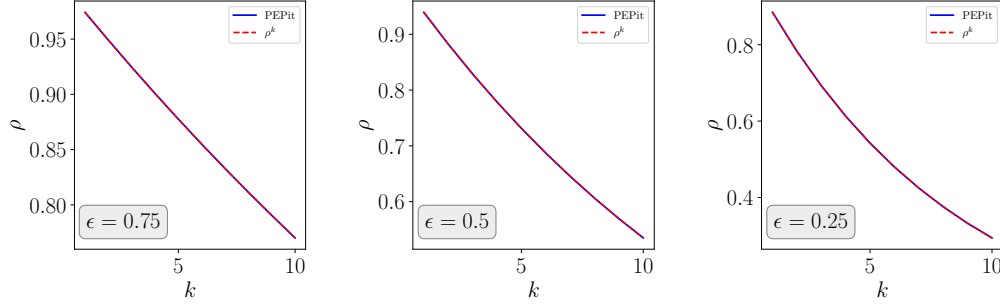


Figure 7: Multi-step Lyapunov analysis for EF^{21} , computed using PEPit. The blue line shows the contraction factor achieved by the Lyapunov function after k iterations, while the red line represents the theoretical rate ρ^k , where ρ is the single-step contraction factor. Each column corresponds to a different value of $\epsilon = 0.75, 0.5, 0.25$, with $L = 1.0$ and $\mu = 0.1$ fixed across all plots.

	$\kappa = 2.0$	$\kappa = 4.0$	$\kappa = 10.0$
Absolute error	1.69e-07	2.16e-04	2.94e-04

Table 4: Maximum absolute difference in contraction factor for EF^{21} when comparing the general Lyapunov function—constructed using any combination of state terms specified in Subsection 3.1—to the simplified Lyapunov function defined in Theorem 2. The results are computed over a line with $\epsilon \in [0.01, 0.99]$ and with η set to the optimal step size for $L = 1$, and $\mu = 0.5, 0.25, 0.1$.

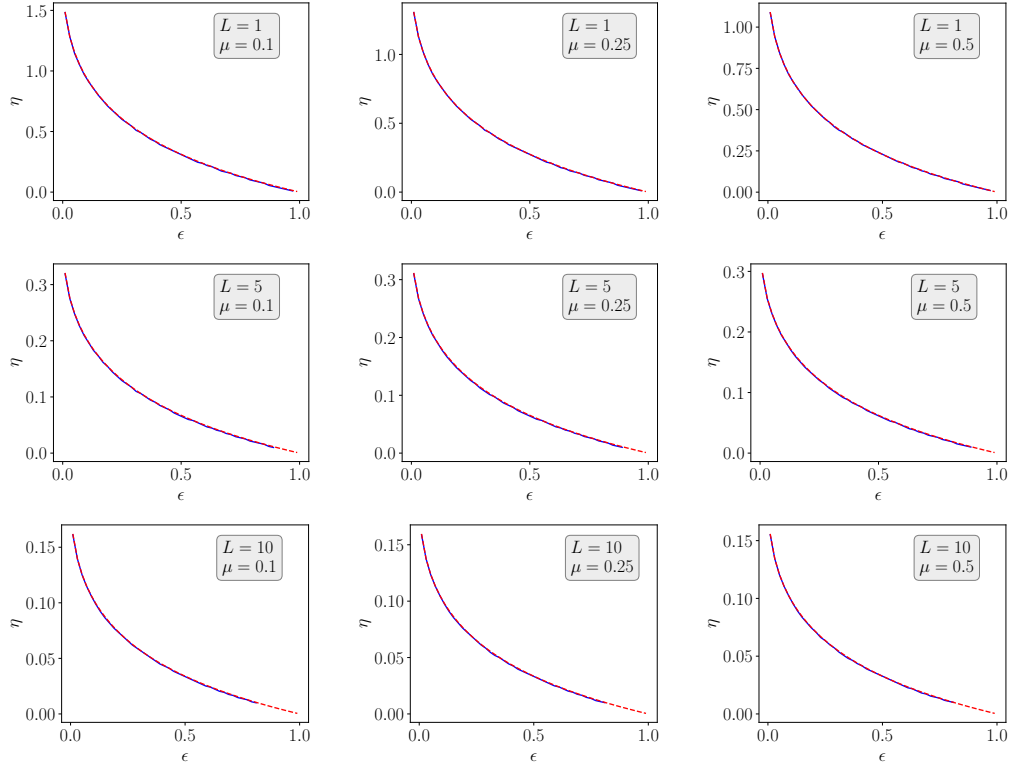


Figure 8: Empirically observed optimal step sizes (blue) in comparison with our setting (red) as a function of ϵ for different values of μ and L . The end of some plots is cut off due to a problem with numerical instability when using larger values of L .

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [NA].
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes]" is generally preferable to "[No]", it is perfectly acceptable to answer "[No]" provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No]" or "[NA]" is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading “NeurIPS Paper Checklist”,**
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers.**

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [Yes]

Justification: Every claim is proven in the paper, and we make the setting and assumptions clear in the paper.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We discuss the fact that we are restricted to the single-agent setting in the discussion. We also make it quite clear that we are working with deterministic compression operators and smooth strongly convex functions in the background section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: All assumptions and full theorem statements are in the paper, and the proofs are in the appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: Full details about how the plots were generated are given in the appendix. We also provide the code as part of the submission for the reviewers, and intend on making it open source afterwards.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code will be made open source after the review process. In the meantime, the reviewers will get access to the code in an anonymized form.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Necessary details to understand the plots are given in the captions. Details for how they were generated are given in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The paper does include experiments, but they are not random and so this question is not applicable.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [No]

Justification: The experiments can be scaled arbitrarily to a fine or course grid of points, and the time to finish is highly dependent on this. Nevertheless, all the experiments were run on an M4 Max.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines?>

Answer: [Yes]

Justification: The authors have not violated the code of ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: The paper only provides tight analysis of already existing methods. The only impact it will have is on the community of researchers working in distributed optimization.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: No data or models are released in this paper.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Citations for the software packages PEPit and PySR are present in paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The only new asset introduced is the code implementing the feasibility problems themselves, and the utility functions surrounding them. The documentation for these are given in the form of instructions for reviewers to replicate our findings in the supplementary material, and in the comments of the code.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No crowdsourcing or human subjects are involved in this paper.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No research with human subjects is involved in this paper.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The paper does not use LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.