A Globally Convergent Evolutionary Strategy for Stochastic Constrained Optimization with Applications to Reinforcement Learning

Youssef Diouane∗
Department of Mathematics and Industrial Engineering
Polytechnique Montréal
youssef.diouane@polymtl.ca

Aurelien Lucchi∗
Department of Mathematics and Computer Science
University of Basel
aurelien.lucchi@unibas.ch

Vihang Patil∗
Institute for Machine Learning
Johannes Kepler University Linz
patil@ml.jku.at

Abstract

Evolutionary strategies have recently been shown to achieve competing levels of performance for complex optimization problems in reinforcement learning. In such problems, one often needs to optimize an objective function subject to a set of constraints, including for instance constraints on the entropy of a policy or to restrict the possible set of actions or states accessible to an agent. Convergence guarantees for evolutionary strategies to optimize stochastic constrained problems are however lacking in the literature. In this work, we address this problem by designing a novel optimization algorithm with a sufficient decrease mechanism that ensures convergence and that is based only on estimates of the functions. We demonstrate the applicability of this algorithm on two types of experiments: i) a control task for maximizing rewards and ii) maximizing rewards subject to a non-relaxable set of constraints.

1 INTRODUCTION

Gradient-based optimization methods are pervasive in many areas of machine learning. This includes deep reinforcement learning (RL) which is notoriously known to be a challenging task due to the size of the search space as well as the problem of delayed rewards. The optimization landscape is also known to have lots of irregularities, where gradients can be extremely small in magnitude (Agarwal et al., 2019), which can severely hinder the progress of gradient-based methods. In order to overcome such difficulties, one needs to be able to efficiently explore the search space of parameters, which partially explains the recent success of a class of global optimization methods known as evolutionary strategies (ES) in reinforcement learning (Maheswaranathan et al., 2018). These methods belong to the class of randomized search that directly search the space of parameters without having to explicitly compute any derivative. Starting from an initial parameter vector $x_0$, the algorithm samples a set of offsprings near $x_0$. Based on the objective function values, the best offsprings are selected to update the parameter $x_0$. Multiple variants of ES methods have been proposed in the literature, including for instance the covariance matrix adaptation evolution strategy (CMA-ES) (Hansen and Ostermeier, 2001) as well as natural evolutionary strategies (Wierstra et al., 2008).

Given the recent attention given to evolutionary strategies in reinforcement learning, the question of global convergence of these methods seems of both theoretical and practical interest. By global convergence, we mean convergence to a first-order stationary point independently of the starting point. One approach for proving global convergence is to modify the traditional ES algorithm by accepting new iterates based on a forcing function that requires a sufficient amount of decrease at each step of the optimization process (Diouane et al., 2015a). A similar paradigm can be adapted to constrained problems (Diouane et al., 2015b, Diouane, 2021). More recently, for a simple instance of ES where recombination is not considered, Glasmachers (2020) showed a form of the global convergence can be achieved without imposing a sufficient decrease conditions on the population. The guarantees provided by these methods are however not applicable to typical practical problems in machine learning where the objective

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151. Copyright 2022 by the author(s). * All authors contributed equally to this work.
function (and potentially the constraints) can not be evaluated exactly, either for computational reasons, or because of the existence of inherent noise. We address this problem taking as a core motivation the problem of reinforcement learning where an agent learns to act by a process of trial and error which over time allows it to improve its performance at a given task. While early work in reinforcement learning allowed the agent to freely explore actions, more recent work, e.g. Achiam et al. (2017), has advocated for the use of constrained policies. As pointed out in Achiam et al. (2017), this is critical in certain environments such as robot automation for industrial or medical applications. Another typical example of constraints that are commonly found in RL are for maximizing the entropy of a policy (Haarnoja et al., 2018). Without such constraints, one might converge to a local solution that is far away from any global optimum. More examples of constrained problems for safety purposes can be found in (Ray et al., 2019b). Motivated by RL applications, the proposed approach in this paper extends the works (Dionane et al., 2015a;b) to the setting where only stochastic estimates of the objective and the constraints are available.

Our main goal is to design a variant of an evolutionary strategy with provable convergence guarantees in a constrained and stochastic setting. Broadly, the problem we consider can be cast as the following general stochastic optimization problem:

$$\min_x f(x) \quad \text{s.t.} \quad x \in \Omega,$$

where the objective function $f$ is assumed to be continuously differentiable. The feasible region $\Omega \subset \mathbb{R}^n$ will be assumed, in the context of this paper, to be of the form:

$$\Omega = \{x \in \mathbb{R}^n | \forall i \in \{1, \ldots, r\}, c_i(x) \leq 0\},$$

where each $c_i : \Omega \to \mathbb{R}$ is a given constraint function. We note that both linear and non-linear constraints can adequately been incorporated in $\Omega$.

In this paper, our main contributions are as follows:

- The design of a variant of an evolutionary strategy with provable convergence guarantees. While prior work in reinforcement learning such as Salimans et al. (2017); Choromanski et al. (2019) has demonstrated the good empirical performance of evolutionary strategies, it does not provide convergence guarantees.

- The theoretical guarantees we derive apply to unconstrained and constrained stochastic problems. While convergence guarantees for ES exist for optimization problems with **exact** function values (for the objective and the constraints), we are not aware of any prior work that handles stochastic problems where only estimates of the objective and the constraints are available.

- We test the empirical performance of our approach on a variety of standard RL problems and observe higher returns compared to common baselines. Importantly, our algorithm guarantees the feasibility of the constraints, which might be extremely important in some environments.

2 RELATED WORK

Evolutionary strategies in RL Classical techniques to solve RL problems include methods that use trajectory information such as policy gradients or Q-learning (Sutton and Barto, 2018). One alternative to these techniques is to use black-box optimization methods such as random search techniques. There has recently been a renewed interest in such methods, especially in the context of deep RL problems where they have been shown to be scalable to large problems (Mania et al., 2018; Salimans et al., 2017; Choromanski et al., 2019). For instance, Salimans et al. (2017) showed that ES can be scaled up using distributed systems while Maheswaranathan et al. (2018) suggested to use surrogate gradients to guide the random search in high-dimensional spaces. Another advantage ES methods have is that they are not affected by delayed rewards (Arjona-Medina et al., 2019; Patil et al., 2020). Because evolutionary methods learn from complete episodes, they tend to be less sample efficient than classical deep RL methods. This problem has been addressed in prior work, including e.g. (Pourchot and Sigaud, 2018) who suggested an approach named CEM-RL that combines an off-policy deep RL algorithm with a type of evolutionary search named Cross Entropy Method (CEM). The combination of these approaches makes CEM-RL able to trade-off between sample efficiency and scalability. Khadka and Tumer (2018) proposed another sample efficient hybrid algorithm where they utilize gradient information by adding an agent trained using off-policy RL into the evolving population at some fixed interval. Liu et al. (2019) improve the sample efficiency of ES by using a trust region approach that optimizes a surrogate loss, enabling to reuse data sample for multiple epochs of updates. Conti et al. (2017) improve the exploration qualities of ES for RL problems by utilizing a population of novelty seeking agents. Further, ES has also been used to evolve policies in model-based RL (Ha and Schmidhuber, 2018).

Constrained optimization in RL Achiam et al. (2017) designed an algorithm to optimize the return while satisfying a given set of deterministic constraints.
Their approach relies on a trust-region method which has been shown in (Salimans et al., 2017) to be practically outperformed by evolutionary strategies in various environments. A similar approach was proposed by Tessler et al. (2018a) but for a larger set of deterministic constraints. Further, Chow et al. (2019) use Lyapunov constraints to obtain feasible solutions on which the policy or the action is projected to guarantee the satisfaction of constraints. Another application of constrained optimization is to enforce safety rules in an RL environment. For instance, an agent exploring an environment might not want to visit certain states that are deemed unsafe. This problem has been formalized in (Altman, 1999) which will be discussed in more details in Sec. 5. In this paper, to handle constraints, we extend the unrelaxable constraints methodology, as in (Audet and Dennis Jr., 2006), to include uncertainties in the estimates of the objective function and the constraints. In particular, in our context, the constraints will be handled using an adjusted extreme barrier function (see Section 3).

**Maximum Entropy RL**  Entropy maximization in RL has been claimed to connect local regions in the optimization landscape, thereby making it smoother (Ahmed et al., 2018), which enables faster learning and also better exploration. Recent prior work includes Soft Actor-Critic (Haarnoja et al., 2018), Soft Q-learning (Haarnoja et al., 2017), and also better exploration. Recent prior work in- cludes Soft Actor-Critic (Haarnoja et al., 2018), Soft Q-learning (Haarnoja et al., 2017). To the best of our knowledge, none of the works discussed above provided convergence guarantees for an ES algorithm in a stochastic constrained setting. As we will see shortly, this will require a new Lyapunov function that is different from the one used for deterministic methods, e.g. (Diouane et al., 2015a).

3 **METHOD**

3.1 The proposed framework

**Provably convergent ES**  Evolution strategies iteratively sample candidate solutions from a distribution \( \mathcal{D}_k \) (scaled by a factor \( \sigma_k^{ES} > 0 \)) and select the best subset of candidates to create an update direction \( d_k \). The next iterate is then given by \( x_{k+1} = x_k + \sigma_k d_k \) where \( \sigma_k \) is a step-size parameter. A general technique (Diouane et al., 2015a) to ensure this approach globally converges is by imposing a sufficient decrease condition on the objective function value, which forces the step size \( \sigma_k \) to converge to zero. Constrained problems are discussed in (Diouane et al., 2015b), which starts with a feasible iterate \( x_0 \) and prevents stepping outside the feasible region by means of a barrier approach. In this context, the sufficient decrease condition is applied not to \( f \) but to the extreme barrier function \( f_{\Omega} \) associated to \( f \) with respect to the constraints set \( \Omega \) (Audet and Dennis Jr., 2006) (also known as death penalty function), which is defined by:

\[
 f_{\Omega}(x) = \begin{cases} f(x) & \text{if } c(x) \leq 0 \\ +\infty & \text{otherwise} \end{cases}
\]

(3) where \( c(x) \) is a constraint function as defined in Eq. 2.

**Inexact function values and constraints**  In this work, we consider the case where the function values cannot be accessed exactly and only some estimates of the objective function and the constraints are available. The definition of the barrier function evaluated at a point \( x_k \) is adjusted as follows:

\[
 f_k = \begin{cases} f_k & \text{if } c_k - \varepsilon \sigma_k \leq 0 \\ +\infty & \text{otherwise} \end{cases}
\]

(4)

where \( f_k \) and \( c_k \) are the estimate of \( f \) and \( c \) at the point \( x_k \), and \( \varepsilon > 0 \) a fixed tolerance on the constraints. The obtained method is thus given by Algorithm 1.

**Algorithm**  The first two steps sample a set of \( \lambda > 0 \) candidate directions and rank them according to their corresponding function values. In step 3, the algorithm combines the best subset of these directions (of size \( \lambda > 0 \)) using a linear mapping \( \Psi_k \) whose choice depends on the chosen ES strategy. For instance, using a CMA-ES strategy as proposed by Hansen and Ostermeier (2001), the mapping \( \Psi_k \) is a simple averaging function, i.e.

\[
 \Psi_k(d_k^1, \ldots, d_k^\lambda) = \sum_{i=1}^{\lambda} w_i^k d_i^k
\]

where the weights \( \{w_i^k\} \) belong to a simplex set. Another example of mapping function \( \Psi_k \) is the Guided ES (Maheswaranathan et al., 2018), where one typically has \( \lambda = 2\lambda' \) and \( \Psi_k \) is given by

\[
 \Psi_k(d_k^1, \ldots, d_k^{\lambda'}) = \frac{1}{\lambda'} \sum_{i=1}^{\lambda'} (f_{\lambda' k} + f_{\lambda' k + 1}^{\lambda'k}) / \sigma_k^{ES} d_i^k
\]

For further details, we refer the reader to the appendix. The direction computed by \( \Psi_k \) is denoted by \( d_k \). The algorithm steps in the direction \( d_k \) using a step size \( \sigma_k \), which is then adjusted in step 4 depending on whether the iteration decreases the function or not. We note that, for generality reasons, the updates of the ES parameters \( \sigma_k^{ES} \) and \( \mathcal{D}_{k+1} \) in step 5 are purposely left unspecified. In fact, our convergence analysis is independent of the choice of the sequences \( \{\sigma_k^{ES}\} \) and \( (\mathcal{D}_k) \). For the experimental results reported in Section 5 we use the same update rule for \( \sigma_k^{ES} \) as \( \sigma_k \).

**Remark 1** (Extreme barrier vs projection). In some applications, the feasible set is formed with linear constraints or simple bounds. In such cases where a projection to the feasible domain is computationally affordable, the use of the barrier function given by (4) in Algorithm 1 can be replaced by a projection. As long as the sufficient decrease condition is enforced, our convergence theory applies. Using exact estimates, Diouane et al. (2015b) showed that the analysis for both an extreme barrier approach and a projection approach to handle constraints are equivalent.
Algorithm 1: A class of ES using estimates

1: Choose positive integers \( \lambda \) and \( \lambda' \) such that \( \lambda \geq \lambda' \). Choose initial step lengths \( \sigma_0, \sigma^\text{ES}_0 > 0 \) and the constants \( \gamma, \ d_{\text{max}} \) such that \( \gamma \geq 1 \) and \( d_{\text{max}} > 0 \). Select two positive constants \( \epsilon_c > 0 \) and \( \kappa > 0 \). Select an initial \( x_0 \in \mathbb{R}^n \) such that \( \epsilon_c \sigma_0 \) and evaluate \( f_0 < \infty \) the estimation of \( f \) at \( x_0 \).
2: for \( k = 0,1, \ldots \) do
3:  Step 1: compute new sample points \( Y_{k+1} = \{y^1_{k+1}, \ldots, y^\lambda_{k+1}\} \) such that, for all \( i = 1, \ldots, \lambda \), one has \( y^i_{k+1} = x_k + \sigma^\text{ES}_k d_k^i \) where the directions \( d_k^i \)'s are drawn from a distribution \( D_k \).
4:  Step 2: compute \( \{f^i_{k+1}\}_{i=1}^{\lambda} \) the estimates of \( f \) at the set point \( Y_{k+1} \) and re-order the offspring points of \( Y_{k+1} \) into \( \tilde{Y}_{k+1} = \{\tilde{y}^1_{k+1}, \ldots, \tilde{y}^\lambda_{k+1}\} \) by increasing order: \( \tilde{f}^1_{k+1} \leq \cdots \leq \tilde{f}^\lambda_{k+1} \) where \( \tilde{f}^i_{k+1} \) is the estimation of \( f \) at the sample point \( \tilde{y}^i_{k+1} \).
5:  Step 3: select the new parents as the best \( \lambda' \) offspring sample points \( \{\tilde{y}^1_{k+1}, \ldots, \tilde{y}^\lambda'_{k+1}\} \) and let \( \{\tilde{d}^1_k, \ldots, \tilde{d}^\lambda'_{k}\} \) be the associated directions. Set \( d_k = \Psi_k(d^1_k, \ldots, d^\lambda'_{k}) \) where \( \Psi_k \) is a linear mapping related to the chosen ES strategy (such that \( \|d_k\| \leq d_{\text{max}} \)). Let \( x_{\text{trial}} = x_k + \sigma_k d_k \) (5)
and \( \tilde{f}_{\text{trial}} \) be the estimate of the barrier function at \( x_{\text{trial}} \) using \( \tilde{d}_{\text{trial}} \).  
6:  Step 4:  
7:  if \( \tilde{f}_{\text{trial}} \leq f_k - \frac{\gamma}{2} \sigma_k^2 \) then  
8:  Consider the iteration successful, set \( x_{k+1} = x_{\text{trial}}, f_{k+1} = \tilde{f}_{\text{trial}}, \) and \( \sigma_{k+1} = \gamma^{-1} \sigma_k \).  
9:  else  
10: Consider the iteration unsuccessful, set \( x_{k+1} = x_k, f_{k+1} = f_k \) and \( \sigma_{k+1} = \gamma^{-1} \sigma_k \).  
11: end if  
12:  Step 5: update the ES parameters (i.e., \( \sigma^\text{ES}_k \) and \( D_{k+1} \)).  
13: end for

Remark 2 (Analysis unconstrained case (new result)). Although Algorithm 1 is presented for constrained problems, the adaptation to the unconstrained case is straightforward. Indeed, it suffices to replace the barrier function estimates Eq. (4), computed at the offspring points and at the trial point \( x_{\text{trial}} \) by estimates of the objective function at the same points. The convergence analysis of the unconstrained framework can be deduced from the analysis we derived in the constrained case. We emphasize that, to the best of our knowledge, the analysis for the stochastic unconstrained case is also a new result in the literature.

3.2 Accuracy of the estimates

In order to obtain convergence guarantees for Algorithm 1, we require the estimates of \( f \) to be sufficiently accurate with a suitable probability. For practical reasons, we are interested in the case where the directions in Algorithm 1 are not defined deterministically but generated by a random process defined in a probability space \((\mathcal{E}, \mathcal{F}, \mathbb{P})\). Note that the randomness of the direction implies the randomness of the iterate \( X_k \), the direction \( D_k \), the parameters \( \Sigma_k \) and \( S_k = \Sigma_k D_k \). Given a sample \( \omega \in \mathcal{E} \), we denote by \( d_k = D_k(\omega), x_k = X_k(\omega), \sigma_k = \Sigma_k(\omega) \), and \( s_k = S_k(\omega) \) their respective realizations. Moreover, the objective function \( f \) and the constraints are supposed to be accessed only through stochastic estimators. Therefore, we define the realizations of the random variables \( F^0_k \) (i.e., the estimate of the objective function \( f \) at the iterate \( X_k \)) and \( F^1_k \) (i.e., the estimate of the objective function \( f \) at the iterate \( X_k + S_k \)) by \( f^0_k \) and \( f^1_k \) respectively. Similarly, we denote the realizations of the constraints \( C^0_k \) (i.e., the estimate of the constraints \( c \) at the iterate \( X_k \)) and \( C^1_k \) (i.e., the estimate of the constraints \( c \) at the iterate \( X_k + S_k \)) by \( c^0_k \) and \( c^1_k \). As mentioned earlier, we will require the random estimates to have a certain degree of accuracy during the application of the proposed framework. The accuracy of the objective functions estimates is formalized below.

Definition 1. Given constants \( \epsilon > 0 \), and \( p \in (0,1] \), the sequence of the random quantities \( F^0_k \) and \( F^1_k \) is called \( p \)-probabilistically \( \epsilon_f \)-accurate, for corresponding sequences \( \{X_k\}, \{\Sigma_k\} \), if the event

\[
T^f_k \triangleq \{ |F^0_k - f(X_k)| \leq \epsilon \Sigma_k \text{ and } |F^1_k - f(X_k + S_k)| \leq \epsilon \Sigma_k \}
\]

satisfies the condition \( \mathbb{P} \left( T^f_k \mid F_{k-1} \right) \geq p \), where \( F_{k-1} \) is the \( \sigma \)-algebra generated by \( F^0_0, F^1_0, \ldots, F^0_{k-1}, F^1_{k-1} \) and \( C^0_0, C^1_0, \ldots, C^0_{k-1}, C^1_{k-1} \).
Definition 2. Given a constant $\varepsilon_c > 0$, the sequence of the random quantities $C_k^0$ and $C_k^1$ is called almost-surely $\varepsilon_c$-accurate, for corresponding sequences $\{X_k\}$, $\{\Sigma_k\}$, if the event
\[
T_k^c \Delta \{ \|C_k^0 - c(X_k)\|_\infty \leq \varepsilon_c \Sigma_k \\
\text{and } \|C_k^1 - c(X_k + S_k)\|_\infty \leq \varepsilon_c \Sigma_k \}
\]
satisfies the condition $\mathbb{P}( T_k^c \mathcal{F}_{k-1} ) = 1$, where $\mathcal{F}_{k-1}$ is the $\sigma$-algebra generated by $F_0^0, F_0^1, \ldots, F_{k-1}^0, F_{k-1}^1$ and $C_0^0, C_1^0, \ldots, C_{k-1}^0, C_{k-1}^1$.

In Definition 1, the accuracy of the function estimation gap is of order $\Sigma_k^2$, which is a common assumption in the literature, see e.g., Blanchet et al. (2019). For the constraints, our analysis will require only to have estimates that converge to the exact value as $\Sigma_k \to 0$. For simplicity reasons, we make the choice of using only $\Sigma_k$ in Eq. (4) and Definition 2 to measure the accuracy level of the constraints. That can be generalized to take the form $\|C_k^0 - c(X_k)\|_\infty \to 0$ and $\|C_k^1 - c(X_k + S_k)\|_\infty \to 0$ as $\Sigma_k \to 0$.

3.3 Global convergence

We derive a convergence analysis of Algorithm 1 under the following assumptions.

Assumption 1. $f$ is continuously differentiable on an open set containing the level set $\mathcal{L}(x_0) = \{x \in \mathbb{R}^n \mid f(x) \leq f(x_0)\}$, with Lipschitz continuous gradient, of Lipschitz constant $L$.

Assumption 2. $f$ is bounded from below by $f_{\text{low}}$.

Assumption 3. The sequence of random objective function estimates $\{F_k^0, F_k^1\}_k$ generated by Algorithm 1 satisfies the two following conditions:

1. The sequence $\{F_k^0, F_k^1\}_k$ is $p$-probabilistically $\varepsilon_f$-accurate for some $p \in \left(\frac{1}{2}, 1\right]$, $\varepsilon_f \in (0, \frac{4}{3})$ where $\kappa$ is a constant used in Algorithm 2.

2. There exists $\varepsilon_v > 0$ such that the sequence of estimates $\{F_k^0, F_k^1\}_k$ satisfies the following $\varepsilon_v$-variance condition for all $k \geq 0$,
\[
\mathbb{E}\left(\|F_k^0 - f(X_k)\|^2 | \mathcal{F}_{k-1}\right) \leq \varepsilon_v^2 \Sigma_k^4 \\
\text{and } \mathbb{E}\left(\|F_k^1 - f(X_k + S_k)\|^2 | \mathcal{F}_{k-1}\right) \leq \varepsilon_v^2 \Sigma_k^4.
\]

Assumption 4. For all $k$, the sequence of random constraints estimates $\{C_k^0, C_k^1\}_k$ generated by Algorithm 1 is $\varepsilon_c$-accurate almost surely, for a given constant $\varepsilon_c > 0$.

Existence of a converging subsequence For the sake of our analysis, we introduce the following (random) Lyapunov function
\[
\Phi_k = \nu f(X_k) + (1 - \nu) \Sigma_k^2,
\]
where $\nu \in (0, 1)$. Consider a realization of Algorithm 1 and let $\Phi_k$ be the corresponding realization of $\Phi_k$. The next theorem shows that, under Assumption 3, the imposed decrease condition, in Algorithm 1 leads to an expected decrease on the Lyapunov function $\Phi_k$.

Theorem 1. Let Assumption 2 hold. Suppose that Assumption 3 is also satisfied with probability $p$ such that
\[
\frac{p}{(1-p)^{1/2}} \geq \left(\frac{1}{1-\nu}\right)^{1/(1-\gamma^2)}
\]
and $\nu \in (0, 1)$ satisfies
\[
\frac{\nu}{1-\nu} \geq 4(\kappa^2-1) \frac{1}{\kappa}.
\]
Then, there exists an $\alpha > 0$ such that, for all $k$,
\[
\mathbb{E}(\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}) \leq -\alpha \Sigma_k^2.
\]

Hence, the true value of the objective function $f$ may not decrease at each individual iteration but Theorem 1 ensures that the Lyapunov function decreases over iterations in expectation as far as the accuracy probability of the estimates of $f$ are high enough. Using such result, one can guarantee that the sequence of step sizes $\{\sigma_k\}$ will converge to zero almost surely. In particular, this will ensure the existence of a subsequence $K$ of iterates driving the step size to zero almost surely. Then, assuming boundedness of the iterates, it will be possible to prove the existence of a convergent subsequence.

Corollary 2. Let Assumption 2 hold. Suppose that the working assumptions of Theorem 1 are also satisfied. Then, the sequence $\{\Sigma_k\}_k$ goes to 0 almost surely. Moreover, if the sequence of iterates $\{X_k\}$ is bounded, then there exists a subsequence $K$ and $X_*$ such that $\{X_k\}_{k \in K}$ goes to zero and $\{X_k\}_{k \in K}$ converges almost surely to $X_*$.

Now that we have established the existence of a converging subsequence, a natural question is to study the properties of its limit point. In our case, we are interested in showing that this limit point satisfies the desired optimality condition for constrained problems, which we briefly review next.

Optimality conditions for constrained problems In optimization, first-order optimality conditions for constrained problems can be defined by using the concept of tangent cones. A known result is that the gradient at optimality belongs to the tangent cone (see, e.g., Thm 5.18 Rockafellar and Wets (1998)). In order to prove that Algorithm 1 satisfies the desired first-order optimality condition, we will require that for iterates $x \in \Omega$ arbitrarily close to $x_*$, the updated point $x + td$ (for $t > 0$ and a fixed direction $d$) also belongs to the constraint set $\Omega$. This can simply be guaranteed by ensuring that the set of the directions $d$ is hyper-tangent to $\Omega$ at $x_*$ (Audet and Dennis Jr. 2006). For readers who are not familiar with constrained optimiza-
tion, we give an overview of the required concepts and definitions in Section A.2 in the appendix.

**Main Convergence Theorem** We now state the main global convergence result for Algorithm 1. A formal variant of this theorem is available in appendix.

**Theorem 3** (Informal). Assume Assumption\textsuperscript{4} and the working assumptions of Corollary\textsuperscript{2} hold. Then, almost surely, the limit point of the subsequence of iterates \( \{X_k(\omega)\}_K \) satisfies the desired optimality condition for constrained problem and \( \lim_{k \in K} \Sigma_k(\omega) = 0 \).

Theorem 8 (appendix) gives the formal statement of Theorem 3. Theorem 8 states that, almost surely, the gradient at a limit point (of the algorithm iterate points) satisfies the desired optimality condition, i.e., it makes an acute angle with the tangent cone of the constraints. We therefore have shown convergence of Algorithm 1 to a point that is guaranteed to satisfy the desired constraints under a set of assumptions that, as discussed below, can be achieved in practice.

**Remark about satisfiability of the assumptions:** We note that the differentiability and boundedness assumptions (Assumptions 1 and 2) are common in machine learning. While the satisfaction of the accuracy required in Assumptions 3 and 4 might appear less trivial, one can in fact easily derive practical bounds in the context of finite sum minimization problems, e.g., Lemma 4.2 by Bergou et al. 2022 (to appear).

For instance, one can perform multiple function evaluations and average them out. We therefore get an estimate \( f_2 = \frac{1}{N} \sum_{i=1}^{N} f_i(x) \), where the set \( \{f_1, \ldots, f_N\} \) correspond to independent samples. Assuming bounded variance, i.e., \( \text{var}(f(x)) \leq v \), known concentration results guarantee that we can obtain \( p \)-probabilistically \( \epsilon_f \)-accurate estimates for \( N \geq \frac{16 v}{\epsilon_f^2} \log\left(\frac{2}{1-p}\right) \) number of evaluations. To also satisfy the variance assumption, we additionally require \( N \geq \frac{18}{\max_{i=1}^{N}u_i^2} \). We also note that one could still violate Assumption 3 to some degree and obtain convergence to a neighborhood around the optimum.

In the presence of constraints, Assumption 4 allows the use of inaccurate estimations of the constraints, in particular during the early stage of the optimization process (when \( \sigma_k \) may be large). In practice, our framework can also be seen suitable to handle hidden constraints (e.g., code failure) as far as one assumed that the set of such constraints is of \( P \)-measure zero. A second practical scenario is related to equality constraints where the feasible domain can be hard to reach using an evolution strategy. Using Assumption 4 allows us to relax the feasible domain. For instance, an equality constraint of the form \( c(x) = 0 \), by using Assumption 4, will be handled as \( |c(x)| \leq \epsilon_k \sigma_k \). This relaxation can gradually help the ES to explore the feasible domain and improve the objective function. The empirical validity of the assumptions on our test cases is given in the appendix, see Figure 7 and 8.

**Remark about novelty of our analysis:** Compared to prior works, we are the first to propose a class of globally convergent ES methods to handle noisy estimates of the objective function \( f(x) \). The extension of the proposed framework to handle constraints in a stochastic setting, as described by Eq. 4 is a second contribution of this work. The convergence analysis in particular requires designing a new Lyapunov function (as given by Eq. 9) that depends on both the function values and the level of noise on the objective function. Detailed proofs are provided in the appendix.

### 4 IMPLEMENTATION AND TEST CASES

We tested an adaptation of our Algorithm 1 that used the guided search approach introduced in Maheswaranathan et al. (2018). The Guided-Evolution Strategy (GES) defines a search distribution from a subspace spanned by a set of surrogate gradients. Importantly, this modification is also covered by the convergence guarantees derived in Theorem 3 (note that Maheswaranathan et al. 2018 did not provide convergence guarantees and was not applied to the constrained setting). We refer the reader to Section B in the appendix for further details. From here we denote our implementation of Algorithm 1 as PCCES (Provably Convergent Constrained ES). In what comes next, we present two different reinforcement learning applications.

**Constrained entropy maximization:** We consider the standard formalization of reinforcement learning (RL) as a finite time Markov Decision Process (MDP). At each time step \( t \), an RL agent receives a state \( s_t \) based on which it selects an action \( a_t \) using a policy denoted by \( \pi \). The environment then provides a reward \( r_t \) and a new state to the agent. We optimize the policy such that it learns to output the optimal sequence of actions that maximizes the cumulative reward over all steps. Formally, we consider a trajectory \( \tau := (s_0, a_0, r_0, \ldots, s_T, a_T, r_T) \) as a sequence of state-action-reward triples which is distributed according to \( \pi \). The goal of the RL agent is to maximize the objective \( R(\tau) = \sum_{t=0}^{T} r_t \delta^t \), where \( \delta \in [0,1) \) is the discount factor. We consider stochastic policies \( \pi_x(a|s) \) which are parameterised by \( x \), the parameters of a neural network. Then, we define the expected

\[ \text{Expected} \]
return as a function of trajectories generated by a policy, i.e., \( \mathbb{E}_{\tau \sim \pi}[R(\tau)] \), where the return \( R(\tau) \) is the discounted sum of rewards from the trajectory \( \tau \). The problem of finding the optimal policy \( \pi^* \) is thus given by

\[
\pi^* := \text{argmax}_{\pi} \mathbb{E}_{\tau \sim \pi}[R(\tau)].
\]

In this paper, we change the latter unconstrained optimization problem by adding constraints while maximizing the entropy of the learned policy. This has been shown to improve the exploration abilities of the agent and as a result yield higher return policies (Mnih et al., 2016; Ahmed et al., 2018; Haarnoja et al., 2018). In evolution strategies, the iterations are episodic in nature and not over time steps. Thus, we define the entropy of a policy over complete trajectories. We define the entropy of a policy \( \pi \) over a trajectory as the sum of the entropy over states in the trajectory, i.e.

\[
H_\pi(\tau) = \sum_{t=1}^{T} h_\pi(s_t),
\]

where \( \tau \) is the trajectory and \( h_\pi(s_t) \) is the entropy of the policy distribution at time step \( t \). We then define the constraint set \( \Omega \) as constraints that determine an acceptable interval for the entropy \( H_\pi(\tau) \). We obtain the following constrained entropy maximization problem:

\[
\max_{\pi \in \Omega} \mathbb{E}_{\tau \sim \pi}[R(\tau)] + \mu \mathbb{E}_{\tau \sim \pi}[H_\pi(\tau)],
\]

where \( \Omega = \{ \pi : h_l \leq \mathbb{E}_{\tau \sim \pi}[H_\pi(\tau)] \leq h_u \} \), \( h_u \) and \( h_l \) are fixed bounds for the entropy and \( \mu > 0 \) is weights the importance of the entropy term.

Policy optimization with constraints: In this application, we optimize for policies which maximize reward while including non-relaxable conditions over the MDP. Constrained Markov Decision Processes (CMDP) (Altman 1999) is a framework for representing systems with such conditions. Similar to the standard MDP framework, we can then obtain the optimal policy by maximizing the return, over the set of policies which satisfy the constraints. We assume that we are given a set of constraint cost functions \( g^i_\pi(\tau) \), \( i = 1, \ldots, r \) which depend on the policy \( \pi \) used in a specific application. Hence \( \Omega_2 := \{ \pi : \mathbb{E}_{\tau \sim \pi}[g^i_\pi(\tau)] \leq t_i, i = 1, \ldots, r \} \), where each \( \{t_i\}_{i=1}^r \) are chosen threshold values. Then, \( \mu \) is a penalty parameter, we target to solve the optimization problem

\[
\max_{\pi \in \Omega_2} \mathbb{E}_{\tau \sim \pi}[R(\tau)] + \mu \mathbb{E}_{\tau \sim \pi}[\sum_{i=1}^{r} g^i_\pi(\tau)].
\]
PPO for 300 epochs (30000 timesteps in each epoch) for CartSafe and MountainCarContinuousSafe. We increase the iterations/epochs to 500 for safety-gym tasks. PCCES does only one update per iteration, while the rest do multiple updates per epoch. This suggests a better scaling of PCCES with less wall-clock time per iteration. We average the runs over 10 different random seeds and plot the cost and return.

**Analysis:** Figure 1 (and Figures 3-6 in appendix) shows that PCCES always finds a better or equivalent solution and also does not stagnate or diverge in the tasks we tried. The sufficient increase condition makes PCCES more robust against divergence and entropy maximization pushes the algorithm to explore the state space, as a result PCCES eventually reaches better returns for a majority of tasks. Ant is a challenging task due to noisy estimates of its return and the need for exploration. The importance of entropy maximization and the sufficient increase condition can be most clearly seen in the Ant task, for which the performance of OpenES and ASEBO stagnates. Similarly, OpenES and ASEBO performance stagnates in Walker and Hopper,
while PCCES consistently keeps improving. Figure 9 (appendix) shows that PCCES consistently ensures the entropy constraints are also satisfied.

Figure 2 shows the performance of PCCES compared to various baselines (CPO, RCPO, PPO) for the constrained tasks. PCCES consistently returns a solution which has a cost lower than the threshold and improves the episodic return for all the environments tested. CPO on the other hand does not keep the cost below threshold, while RCPO and PPO keep the cost low but achieve low returns compared to PCCES for certain environments (CartSafe, PointGoal, PointButton).

6 CONCLUSION

We proposed a class of evolutionary method to solve stochastic constrained problems, with applications to reinforcement learning. One feature that distinguishes our approach from prior work is its global convergence guarantee for stochastic setting. We note that our proof technique does not exploit any specific information about the constraints being related to the entropy or the safety of the policy. Our algorithm could therefore be applied to different types of constraints and also problems outside the area of reinforcement learning. Empirically, we have seen that PCCES consistently achieves higher returns compared to other baselines but might require more iterations to do so. One potential direction to address this problem would be to anneal the constraints. Another future direction to pursue would be to benchmark PCCES in a distributed setting, especially given that prior work by Salimans et al. (2017) demonstrated that the strength of ES lies in its ability to scale over large clusters with less communication between actors. Other directions of interest would be to develop convergence rates as well as second-order guarantees (Gratton et al., 2016; Lucchi et al., 2021). Finally, our approach could also be extended to other problems in machine learning, such as min-max optimization (Anagnostidis et al., 2021).

7 ACKNOWLEDGEMENT

Part of this work was performed while Aurelien Lucchi and Vihang Patil were at ETH Zürich.

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org. Software available from tensorflow.org.

J. Achiam, D. Held, A. Tamar, and P. Abbeel. Constrained policy optimization. In Proceedings of the 34th International Conference on Machine Learning—Volume 70, pages 22–31. JMLR. org, 2017.

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. arXiv preprint arXiv:1908.00261, 2019.

Z. Ahmed, N. L. Roux, M. Norouzi, and D. Schuurmans. Understanding the impact of entropy on policy optimization, 2018.

E. Altman. Constrained markov decision processes. 1999.

S.-K. Anagnostidis, A. Lucchi, and Y. Diouane. Direct-search for a class of stochastic min-max problems. In International Conference on Artificial Intelligence and Statistics, pages 3772–3780. PMLR, 2021.

J. A. Arjona-Medina, M. Gillhofer, M. Widrich, T. Unterthiner, J. Brandstetter, and S. Hochreiter. Rudder: Return decomposition for delayed rewards, 2019.

C. Audet and J. E. Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim., 17:188–217, 2006.

C. Audet, K. J. Dzahini, M. Kokkolaras, and S. L. Digabel. Stochastic mesh adaptive direct search for blackbox optimization using probabilistic estimates. Comput. Optim. Appl., 19:1–34, 2021.

E. Bergou, Y. Diouane, V. Kunc, V. Kungurtsev, and C. W. Royer. A subsampling line-search method with second-order results. INFORMS J. Optim., 0:0–0, 2022 (to appear).a.

E. Bergou, Y. Diouane, V. Kungurtsev, and C. W. Royer. A stochastic levenberg-marquardt method using random models with complexity results. SIAM/ASA J. Uncertainty Quantification, 0:0–0, 2022 (to appear).b.

J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg. Convergence Rate Analysis of a Stochastic Trust Region Method via Supermartingales. INFORMS J. Optim., 2019.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

R. Chen, M. Menickelly, and K. Scheinberg. Stochastic optimization using trust-region method and random models. Math. Program., 169:447–487, 2018.
K. M. Choromanski, A. Pacchiano, J. Parker-Holder, Y. Tang, and V. Sindhwani. From complexity to simplicity: Adaptive es-active subspaces for blackbox optimization. In Advances in Neural Information Processing Systems 32. 2019.

Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman, and M. Gavamzadeh. Lyapunov-based safe policy optimization for continuous control, 2019.

E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. O. Stanley, and J. Chune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents, 2017.

E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning, http://pybullet.org, 2016–2019.

Y. Diouane. A merit function approach for evolution strategies. EURO J. Comput. Optim., 9:100001, 2021.

Y. Diouane, S. Gratton, and L. N. Vicente. Globally convergent evolution strategies. Math. Program., 152:467–490, 2015a.

Y. Diouane, S. Gratton, and L. N. Vicente. Globally convergent evolution strategies for constrained optimization. Comput. Optim. Appl., 62:323–346, 2015b.

R. Durrett. Probability: Theory and Examples. Cambridge Series in Statistical and Probabilistic Mathematics. fourth edition. Cambridge University Press, Cambridge, 2010.

T. Glasmachers. Global convergence of the (1+1) evolution strategy to a critical point. Evol. Comp., 28:27–53, 2020.

S. Gratton, C. W. Royer, and L. N. Vicente. A second-order globally convergent direct-search method and its worst-case complexity. Optimization, 65(6):1105–1128, 2016.

D. Ha and J. Schmidhuber. World models, 2018.

T. Haarnoja, H. Tang, P. Abbeel, and S. Levine. Reinforcement learning with deep energy-based policies, 2017.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. Evolutionary computation, 9(2):159–195, 2001.

C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. Nature, 558(7682):357–362, Sept. 2020. doi: 10.1038/s41586-020-2649-2.

J. D. Hunter. Matplotlib: A 2d graphics environment. Computing in Science & Engineering, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

S. Khadka and K. Tumer. Evolutionary reinforcement learning. CoRR, abs/1805.07917, 2018.

E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica. Rllib: Abstractions for distributed reinforcement learning, 2018.

G. Liu, L. Zhao, F. Yang, J. Bian, T. Qin, N. Yu, and T.-Y. Liu. Trust region evolution strategies. Proceedings of the AAAI Conference on Artificial Intelligence, 33:4352–4359, Jul 2019. doi: 10.1609/aaai.v33i01.33014352.

A. Lucchi, A. Orvieto, and A. Solomou. On the second-order convergence properties of random search methods. Advances in Neural Information Processing Systems, 34, 2021.

N. Maheswaranathan, L. Metz, G. Tucker, D. Choi, and J. Sohl-Dickstein. Guided evolutionary strategies: Augmenting random search with surrogate gradients. arXiv preprint arXiv:1806.10230, 2018.

H. Mania, A. Guy, and B. Recht. Simple random search of static linear policies is competitive for reinforcement learning. In Advances in Neural Information Processing Systems, pages 1800–1809, 2018.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019.

V. P. Patil, M. Hofmarcher, M. Dimu, M. Dorfer, P. M. Blies, J. Brandstetter, J. A. Arjona-Medina, and S. Hochreiter. Align-rudder: Learning from few demonstrations by reward redistribution. CoRR, abs/2009.14108, 2020. URL https://arxiv.org/abs/2009.14108.

A. Pourchot and O. Sigaud. CEM-RL: combining evolutionary and gradient-based methods for policy search. CoRR, abs/1810.01222, 2018.
A. Ray, J. Achiam, and D. Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning. 2019a.

A. Ray, J. Achiam, and D. Amodei. Benchmarking safe exploration in deep reinforcement learning. 2019b.

R. T. Rockafellar and R. J.-B. Wets. Variational Analysis. Springer-Verlag Berlin Heidelberg, 1998.

T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. arXiv preprint arXiv:1703.03864, 2017.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017.

R. Sutton and A. G. Barto. Reinforcement Learning: An Introduction. second edition. MIT Press, 2018.

C. Tessler, D. J. Mankowitz, and S. Mannor. Reward constrained policy optimization. arXiv preprint arXiv:1805.11074, 2018a.

C. Tessler, D. J. Mankowitz, and S. Mannor. Reward constrained policy optimization. CoRR, abs/1805.11074, 2018b. URL http://arxiv.org/abs/1805.11074.

D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural evolution strategies. In 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pages 3381–3387. IEEE, 2008.
A Main analysis

A.1 Existence of a converging subsequence

Lemma 4. Let Assumptions 1 and 3 hold. Then, for every iteration $k$, one has

$$\mathbb{E}\left(1_{T^f_k} |F^0_k - f(X_k)| |F_{k-1}\right) \leq (1 - p^*_k)^{1/2} \varepsilon \Sigma_k^2$$

and

$$\mathbb{E}\left(1_{T^f_k} |F^1_k - f(X_k + S_k)| |F_{k-1}\right) \leq (1 - p^*_k)^{1/2} \varepsilon \Sigma_k^2$$

where $T^f_k$ denotes the event where the objective function estimates $(F^0_k, F^1_k)$ are inaccurate. $1_{T^f_k}$ is the indicator function of the event set $T^f_k$.

Proof. The proof is the same as in Lemma 1 [Audet et al. 2021].

Lemma 5. Let Assumptions 1 and 3 hold. Then for every iteration $k$, one has

$$\mathbb{E}\left(1_{T^f_k} (\Phi_{k+1} - \Phi_k) |F_{k-1}\right) \leq -(1 - \gamma)(1 - \gamma^{-2}) \nu \Sigma_k^2,$$

where $1_{T^f_k}$ is the indicator function of the event set $T^f_k$.

Proof. Consider a realization of a given iteration $k$ of Algorithm 1 for which the objective function estimates $\{f^0_k, f^1_k\}_k$ are $\varepsilon_f$-accurate. Then, if the iteration is unsuccessful $x_{k+1} = x_k$ and $\sigma_{k+1} = \gamma^{-1} \sigma_k$, this leads to

$$\phi_{k+1} - \phi_k = (1 - \nu)(\gamma^{-2} - 1) \sigma_k^2 := b_1 < 0. \quad (10)$$

Otherwise, if the iteration is successful, then one has $x_{k+1} = x_k + \sigma_k d_k$, where $\sigma_{k+1} \leq \gamma \sigma_k$. One thus has:

$$\phi_{k+1} - \phi_k \leq \nu (f(x_{k+1}) - f(x_k)) + (1 - \nu)(\gamma^2 - 1) \sigma_k^2$$

$$\leq \left(\nu (2\varepsilon_f - \frac{\kappa}{2}) + (1 - \nu) (\gamma^2 - 1)\right) \sigma_k^2$$

$$\leq \left(-\nu \frac{\kappa}{4} + (1 - \nu) (\gamma^2 - 1)\right) \sigma_k^2,$$

where the last inequality is obtained using the fact that $\varepsilon_f \leq \frac{\varepsilon}{4}$. Assuming that $\frac{\nu}{1 - \nu} \geq \frac{8(\gamma^2 - 1)}{\kappa}$, one deduces that

$$\phi_{k+1} - \phi_k \leq -\nu \frac{\kappa}{8} \sigma_k^2 := b_2 < 0.$$

We note also that, as $\frac{\nu}{1 - \nu} \geq \frac{8(\gamma^2 - 1)}{\kappa \gamma^*}$, one has $b_2 \leq b_1$. Hence, one deduces that for any iteration $k$ conditioned by this case, one has

$$\phi_{k+1} - \phi_k \leq b_1.$$
Hence, since the event $T_k$ occurs with a probability $p_k^*$, one gets
\[
\mathbb{E}\left(1_{T_k}(\Phi_{k+1} - \Phi_k) | \mathcal{F}_{k-1}\right) \leq -(1 - \nu)(1 - \gamma^{-2})p_k^* \Sigma_k^2.
\]

**Lemma 6.** Let Assumption 3 hold. Suppose that Assumption 4 is also satisfied with probability $\nu$. Then, for every iteration $k$, one has
\[
\mathbb{E}\left(1_{T_k}(\Phi_{k+1} - \Phi_k) | \mathcal{F}_{k-1}\right) \leq 2\nu(1 - p_k^*)^{1/2} \epsilon_v \Sigma_k^2.
\]

**Proof.** Conditioned by the event $\mathcal{F}_{k-1}$, assuming that the objective function estimates $\{F_k, F_k^1\}_k$ are inaccurate. One has, if the iteration $k$ is successful, $X_{k+1} = X_k + \Sigma_k D_k$, where $\Sigma_{k+1} \leq \gamma \Sigma_k$. Hence, using Lemma 4, we get
\[
\mathbb{E}\left(1_{T_k}(\Phi_{k+1} - \Phi_k) | \mathcal{F}_{k-1}\right) \leq \nu \mathbb{E}\left(1_{T_k}(f(X_{k+1}) - f(X_k)) | \mathcal{F}_{k-1}\right) + (1 - \nu)(1 - p_k^*) \gamma^2 \Sigma_k^2
\]
\[
\leq \nu \left(2(1 - p_k^*)^{1/2} \epsilon_v - \frac{c(1 - p_k^*)}{2}\right) \Sigma_k^2 + (1 - \nu)(1 - p_k^*) \gamma^2 \Sigma_k^2.
\]

Assuming that $\frac{\nu}{1 - \nu} \geq \frac{4(\gamma^2 - 1)}{\epsilon_v}$, one gets $-\nu(1 - p_k^*) \epsilon_v + (1 - \nu)(1 - p_k^*) \gamma^2 \leq 0$, hence
\[
\mathbb{E}\left(1_{T_k}(\Phi_{k+1} - \Phi_k) | \mathcal{F}_{k-1}\right) \leq 2\nu(1 - p_k^*)^{1/2} \epsilon_v \Sigma_k^2.
\]

If the iteration is unsuccessful $X_{k+1} = X_k$ and $\Sigma_{k+1} = \gamma^{-1} \Sigma_k$, this leads to
\[
\mathbb{E}\left(1_{T_k}(\Phi_{k+1} - \Phi_k) | \mathcal{F}_{k-1}\right) = (1 - p_k)(1 - \nu)(\gamma^{-2} - 1) \Sigma_k^2 \leq 2\nu(1 - p_k^*)^{1/2} \epsilon_v \Sigma_k^2.
\]

Hence, in both cases, one has
\[
\mathbb{E}\left(1_{T_k}(\Phi_{k+1} - \Phi_k) | \mathcal{F}_{k-1}\right) = (1 - \nu)(\gamma^{-2} - 1)(1 - p_k^*) \Sigma_k^2 < 2\nu(1 - p_k^*)^{1/2} \epsilon_v \Sigma_k^2.
\]

**Theorem 1.** Let Assumption 3 hold. Suppose that Assumption 4 is also satisfied with probability $p$ such that
\[
\frac{p}{(1 - p)^{1/2}} \geq \frac{4\nu \epsilon_v}{(1 - \nu)(1 - \gamma^{-2})} \quad \text{and} \quad \nu \in (0, 1) \quad \text{satisfies} \quad \frac{\nu}{1 - \nu} \geq \frac{4(\gamma^2 - 1)}{\epsilon_v}.
\]

Then, there exists an $\alpha > 0$ such that, for all $k$,
\[
\mathbb{E}(\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}) \leq -\alpha \Sigma_k^2. \tag{7}
\]

**Proof.** The proof is inspired from what is done in Bergou et al. (2022 (to appear); Chen et al. (2018); Blanchet et al. (2019); Audet et al. (2021).

Putting the results of the two Lemmas 5 and 6 together, we obtain the following
\[
\mathbb{E}(\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}) \leq -(1 - \nu)p_k^*(1 - \gamma^{-2}) \Sigma_k^2 + 2\nu(1 - p_k^*)^{1/2} \epsilon_v \Sigma_k^2.
\]

Hence, assuming that
\[
\frac{p_k^2}{(1 - p_k^*)^{1/2}} \geq \frac{p}{(1 - p)^{1/2}} \geq \frac{4\nu \epsilon_v}{(1 - \nu)(1 - \gamma^{-2})},
\]

it reduces to
\[
\mathbb{E}(\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}) \leq -\frac{1}{2}(1 - \nu)p_k^*(1 - \gamma^{-2}) \Sigma_k^2 \leq -\alpha \Sigma_k^2
\]

where $\alpha = \frac{1}{2}(1 - \nu)p(1 - \gamma^{-2}) \Sigma_k^2$. 

[\text{Youssef Diouane}, \text{Aurelien Lucchi}, \text{Vihang Patil}]


Corollary 2. Let Assumption 2 hold. Suppose that the working assumptions of Theorem 1 are also satisfied. Then, the sequence \( \{ \Sigma_k \} \) goes to 0 almost surely. Moreover, if the sequence of iterates \( \{ X_k \} \) is bounded, then there exists a subsequence \( \mathcal{K} \) and \( X_* \) such that \( \{ \Sigma_k \}_{k \in \mathcal{K}} \) goes to zero and \( \{ X_k \}_{k \in \mathcal{K}} \) converges almost surely to \( X_* \).

Proof. Indeed, by taking expectation on the result from Theorem 1 one gets

\[
E(\Phi_{n+1} - \Phi_0) = \sum_{k=0}^{n} E(\Phi_{k+1} - \Phi_k) = \sum_{k=0}^{n} E(E(\Phi_{k+1} - \Phi_k | F_{k-1})) \leq -\alpha E \left( \sum_{k=0}^{n} \Sigma_k^2 \right).
\]

Since \( \Phi_{n+1} \geq \nu f_{\text{low}} \), one deduces that by taking \( n \to \infty \)

\[
E \left( \sum_{k=0}^{\infty} \Sigma_k^2 \right) < \infty.
\]

Thus, we conclude that the probability of the random variable \( \Sigma_k \) to converge to zero is one. Moreover, assuming the boundedness of the sequence \( \{ X_k \} \), one deduces the existence of random vector \( X_* \) and a subsequence \( \mathcal{K} \subseteq \mathbb{N} \) such that \( \{ \Sigma_k \}_{k \in \mathcal{K}} \) goes to zero almost surely and \( \{ X_k \}_{k \in \mathcal{K}} \) converges almost surely to \( X_* \).

A.2 Optimality condition for constrained problems

We now turn to deriving a main global convergence result.

Review of required definitions In what comes next, we introduce the formal definition of a hypertangent cone which will be required to state our main convergence theorem. We will denote by \( B(x; \epsilon) \) the closed ball formed by all points at a distance of no more than \( \epsilon \) to \( x \).

Definition 3. A vector \( d \in \mathbb{R}^n \) is said to be a hypertangent vector to the set \( \Omega \subseteq \mathbb{R}^n \) at the point \( x \) in \( \Omega \) if there exists a scalar \( \epsilon > 0 \) such that

\[
y + tw \in \Omega, \quad \forall y \in \Omega \cap B(x; \epsilon), \quad w \in B(d; \epsilon), \quad \text{and} \quad 0 < t < \epsilon.
\]

The hypertangent cone to \( \Omega \) at \( x \), denoted by \( T^H_\Omega(x) \), is the set of all hypertangent vectors to \( \Omega \) at \( x \). Then, the Clarke tangent cone to \( \Omega \) at \( x \) (denoted by \( T^{\text{Cl}}_\Omega(x) \)) can be defined as the closure of the hypertangent cone \( T^H_\Omega(x) \).

Definition 4. A vector \( d \in \mathbb{R}^n \) is said to be a Clarke tangent vector to the set \( \Omega \subseteq \mathbb{R}^n \) at the point \( x \) in the closure of \( \Omega \) if for every sequence \( \{ x_k \} \) of elements of \( \Omega \) that converges to \( x \) and for every sequence of positive real numbers \( \{ t_k \} \) converging to zero, there exists a sequence of vectors \( \{ d_k \} \) converging to \( d \) such that \( x_k + t_k d_k \in \Omega \).

Auxiliary result We state an auxiliary result from the literature that will be useful for the analysis (see Theorem 5.3.1 [Durrett 2010] and Exercise 5.3.1 [Durrett 2010]).

Lemma 7. Assume that, for all \( k \), \( G_k \) is a supermartingale with respect to \( \mathcal{F}_{k-1} \) (a \( \sigma \)-algebra generated by \( G_0, \ldots, G_{k-1} \)). Assume further that there exists \( M > 0 \) such that \( |G_k - G_{k-1}| \leq M < \infty \), for all \( k \). Consider the random events \( C = \{ \lim_{k \to \infty} G_k \text{ exists and is finite} \} \) and \( D = \{ \limsup_{k \to \infty} G_k = \infty \} \). Then \( P(C \cap D) = 1 \).

Main convergence result

Theorem 8 (Formal version of Theorem 3). Assume Assumption 3 and the working assumptions of Corollary 2 hold. Then, there exists an almost surely event \( \mathcal{A} \) such that for all \( \omega \in \mathcal{A} \), \( X_*(\omega) \in \Omega \) is a limit point of the sequence of iterates \( \{ X_k(\omega) \} \) and \( \lim_{k \to \infty} \Sigma_k(\omega) = 0 \). In this case, if \( d \in T^H_\Omega(X_*(\omega)) \) is a limit point associated with \( \{ D_k(\omega) \} \), then \( \nabla f(X_*(\omega))^T d \geq 0 \).

Proof. From Corollary 2 and Assumption 4 it follows that the event

\[
A = \{ \omega \in \mathcal{E} : \exists \mathcal{K} \subset \mathbb{N} \text{ such that } \{ \Sigma_k(\omega) \}_{k \in \mathcal{K}} \to 0 \text{ and } \{ X_k(\omega) \}_{k \in \mathcal{K}} \to X_*(\omega) \} \cap \{ \cap_{k=0}^{\infty} T^H_k \}
\]
We start by showing that 

deduces that

For any \( \sigma \) has \( k \) for \( A \). Thus, there exists \( k \) such that for \( \sigma \) and \( k \) always holds. Hence, there exists \( k \) such that for \( k \) goes to \( \infty \). Thus, there exists \( k \) such that for \( k \) and \( k \geq k_2 \), one has

\[
\sigma_k \leq b_k := \frac{2\epsilon}{\kappa + 4\varepsilon f}.
\]

For any \( k \) such that \( k \geq k_0 = \max\{k_1, k_2\} \), we note that since \( x_k + \sigma_k d_k \in \Omega \) and Assumption \( \ref{assumption:non-decreasing} \) holds, one deduces that \( c_k^0 - \varepsilon_c \sigma_k \leq 0 \) and \( c_k^1 - \varepsilon_c \sigma_k \leq 0 \), meaning that \( \bar{f}_{\text{trial}} = f_k^1 \). Two cases then occur. First if \( \mathbf{1}_{\bar{t}_f} = 1 \), then

\[
\bar{f}_{\text{trial}} - f_k \leq 2\varepsilon f \sigma_k^2 - \epsilon \sigma_k \leq -\frac{\kappa}{2} \sigma_k^2.
\]

Hence, the iteration \( k \) of Algorithm \( \ref{algorithm:gradient-descent} \) is successful and the stepsize \( \sigma_k \) is updated as \( \sigma_{k+1} = \gamma \sigma_k \).

Let now \( B_k \) be the random variable whose realization is \( b_k = \log_{\gamma} \left( \frac{\sum \sigma_k}{b_k} \right) \). Clearly, if \( \mathbf{1}_{\bar{t}_f} = 1 \), one has \( b_{k+1} = b_k + 1 \). Otherwise, if \( \mathbf{1}_{\bar{t}_f} = 0 \), \( b_{k+1} \geq b_k - 1 \) since \( \sigma_{k+1} \geq \gamma^{-1} \sigma_k \) always holds. Hence, \( B_k - B_{k_0} \geq W_k - W_{k_0} \), and from \( \mathbb{P}(\limsup_k W_k = \infty) = 1 \), one obtains \( \mathbb{P}(\limsup_k B_k = \infty) = 1 \). This leads to a contradiction with the fact that \( B_k < 0 \) for any \( k \) such that \( k \geq k_0 \).
B Guided-evolution strategy

As an efficient implementation of Algorithm 1, we tested the GES approach introduced in Maheswaranathan et al. (2018). The GES technique defines a search distribution from a subspace spanned by a set of surrogate gradients denoted by \( S_k \). At each iteration \( k \), the set \( S_k = [G_{k-1-m}, \ldots, G_{k-1}] \) consists of the last \( m \) surrogate gradients computed from iterations \( \{k-1-m, \ldots, k-1\} \). The set \( S_k \) is used to compute an orthogonal basis \( U_k \) of the subspace formed by the vectors in \( S_k \). This is done using a QR decomposition as specified in Maheswaranathan et al. (2018).

Samples are then drawn around the mean vector \( x_k \) according to the distribution \( \mathcal{N}(0, (\sigma_k^2)^2 C_k) \), where the covariance matrix is given by \( C_k = \frac{\alpha}{n} I_n + \frac{1-\alpha}{m} U_k U_k^T \), where \( I_n \) is the identity matrix and \( \alpha \in [0,1] \) is a hyperparameter that trade-offs the influence of the smaller subspace \( \mathbb{R}^m \) over the entire space \( \mathbb{R}^n \). A small value of \( \alpha \) enforces the search to be conducted in the smaller subspace while larger values give the smaller subspace less importance. In practice the directions \( d_k^j \) can efficiently be sampled as follows,

\[
d_k^j = \sigma_k^\text{ES} \sqrt{\frac{\alpha}{n}} + \sigma_k^\text{ES} \sqrt{1-\frac{\alpha}{m}} U_k d'
\]

where, \( d \sim \mathcal{N}(0, I_n) \), \( d' \sim \mathcal{N}(0, I_m) \) and \( \sigma_k^\text{ES} \) is the standard deviation of the distribution from which the direction's are sampled.

The surrogate gradient can be computed in various manners. Update directions in previous iterations can also be used to compute surrogate gradients. We compute the surrogate gradients required to compute \( U_k \) using an Actor-Critic (Sutton and Barto, 2018; Mnih et al., 2016) formulation of policy gradient, where every member of the population \( i \) computes an approximate gradient as

\[
G_{\pi(y_k)} = A_{\pi(y_k)}(s_t, a_t) \nabla \log \pi_{\theta_k}(a_t|s_t)
\]

where \( \pi_{\theta_k} \) is parameterized by \( \theta_k \), \( A_{\pi(y_k)}(s_t, a_t) = R_t - V_{\theta}(s_t) \) is the advantage function and, \( s_t, a_t \) are the state and actions sampled at step \( t \), \( R_t \) is the return from step \( t \) to the last time step \( T \) and \( V_{\theta} \) is the value function parameterized by parameters \( \theta \). Then, the surrogate gradient is averaged to obtain the surrogate gradient for iteration \( k \)

\[
G_k = \frac{1}{\lambda} \sum_{i=1}^{\lambda} G_{\pi(y_k)}
\]

where, \( \lambda \) is the size of the population.

Before we start optimizing the policy, we compute \( m \) surrogate gradients. We note that the guided-search strategy requires storing \( m \) surrogate gradients in memory. If we already have \( m \) surrogate gradients, then we discard the oldest surrogate gradient and replace it by the newer one. Once we have enough surrogate gradients, we sample points around the \( x_k \) using mirrored sampling (Salimans et al., 2017). These sample points are then evaluated on the environment, to compute the return and the entropy. Let \( \lambda' = \lambda/2 \), for a given \( i = 1, \ldots, \lambda' \), we compute the estimation of the objective function for each of the sample point \( y_k^{i} := x_k + d_k^i \) as

\[
f_k^i = -R_{\tau \sim \pi(y_k)}(\tau) - \mu \sum_{c=1}^{r} g_{\tau \sim \pi(y_k)}(\tau)
\]

and its mirror point \( y_k^{i+\lambda'} := x_k - d_k^i \) as

\[
f_k^{i+\lambda'} = -R_{\tau \sim \pi(y_k)}(\tau) - \mu \sum_{c=1}^{r} g_{\tau \sim \pi(y_k)}(\tau)
\]

We then obtain the trial point (Eq. 5) in Algorithm 1 using the following update rule,

\[
x_{k+1}^\text{trial} = x_k - \sigma_k \frac{\beta}{\sigma_k^\text{ES} \lambda} \sum_{i=1}^{\lambda'} \left( f_{k+1}^i - f_{k+1}^{i+\lambda'} \right) d_k^i
\]

A surrogate gradient is defined as an biased or corrupted gradient, which has correlation with the true gradient.
where $d_k^i$ is the direction used to obtain $f_{k+1}^i$, $\sigma_k$ is the stepsize, and $\beta$ is a hyperparameter used for scaling.

Further, we compute the barrier function at the new trial point $x_{k+1}^{\text{trial}}$ as, in the case of entropy maximization by

$$
\tilde{f}_{k+1}^{\text{trial}} = \begin{cases} 
-R(\pi_{x_{k+1}^{\text{trial}}}) - \mu H(\pi_{x_{k+1}^{\text{trial}}}) & \text{if } h_l \leq H(\pi_{x_{k+1}^{\text{trial}}}) \leq h_u \\
+\infty & \text{otherwise}
\end{cases}
$$

Or, in the case of constrained policy optimization by

$$
\tilde{f}_{k+1}^{\text{trial}} = \begin{cases} 
-R_{\tau \sim \pi_{x_{k+1}^{\text{trial}}}}(\tau) - \mu \sum_{c=1}^{r} g_{\tau \sim \pi_{x_{k+1}^{\text{trial}}}}(\tau) & \text{if } [g_{\tau \sim \pi_{x_{k+1}^{\text{trial}}}}(\tau)]_c \leq t_c, c = 1, \ldots, r \\
+\infty & \text{otherwise}
\end{cases}
$$

For both cases, we accept the trial point (i.e. $x_{k+1} = x_{k+1}^{\text{trial}}$ and $f_{k+1} = \tilde{f}_{k+1}^{\text{trial}}$) if the following condition is satisfied,

$$
\tilde{f}_{k+1}^{\text{trial}} \leq f_k - \frac{\kappa}{2} \sigma_k^2
$$

where $\kappa > 0$ is a hyperparameter. We increase the $\sigma_k$ if the iteration is successful and decrease it if it is unsuccessful.
C Additional Experimental Results

Figure 3: Acrobot-v1: PCCES and OpenES runs with 40, 80 and 240 samples per iteration

Figure 4: CartPole-v0: PCCES and OpenES runs with 40, 80 and 240 samples per iteration
Youssef Diouane*, Aurelien Lucchi*, Vihang Patil*

Figure 5: Pendulum-v0: PCCES and OpenES runs with 40, 80 and 240 samples per iteration

Figure 6: MountainCarContinuous-v0: PCCES and OpenES runs with 40, 80 and 240 samples per iteration
Figure 7: Variance of function estimates vs number of samples for CartSafe-v0
Figure 8: Variance of function estimates vs number of samples for MountainCarContinuousSafe-v0

Figure 9: Entropy vs upper threshold of the entropy constraint during training for constrained entropy maximization. PCCES consistently returns a solution which satisfies the constraints.
C.1 Sensitivity to Threshold

We intend to include an ablation study for different thresholds. As a preliminary result, the following table reports the performance of PCCES on two environments with different threshold values after 300 updates and averaged over 5 seeds. This shows that the algorithm is not very sensitive of the choice of threshold.

| Threshold | MountainCarContinuousSafe-v0 | CartSafe-v0 |
|-----------|-----------------------------|-------------|
|           | Threshold                   |             |
|           | 5                           | 15          | 20          |
| Cost      | 2.9 ± 1.1                   | 3.8 ± 1.2   | 3.1 ± 2.09  |
| Return    | 94.9 ± 0.4                  | 95.1 ± 0.46 | 94.6 ± 0.4  |
|           | 15                          | 20          | 35          |
| Cost      | 7.8 ± 6.43                  | 7.1 ± 7.2   | 14.1 ± 13.8 |
| Return    | 227.1 ± 17.9                | 213.3 ± 10.3| 229.7 ± 18.5|

C.2 Performance of CPO

| Clip Ratio | 0.05 | 0.1  | 0.2  |
|------------|------|------|------|
| Cost       | 70.4 ± 4.2 | 76.1 ± 5.3 | 61.3 ± 13.1 |
| Return     | 18.6 ± 2.6  | 19.7 ± 0.8  | 21.4 ± 1.2  |
| Step size  | 1e-4 | 1e-5 | 3e-5 |
| Cost       | 75.2 ± 7.1  | 83.2 ± 6.8  | 68.6 ± 4.1  |
| Return     | 15.36 ± 3.1 | 17.2 ± 0.1  | 19.2 ± 1.9  |

The above table reports the performance of CPO on Safexp-PointButton1-v0 after 300 epochs over 5 seeds for different clip ratios and step sizes. CPO fails to satisfy constraints, which is consistent with results in Safety Benchmarks [Ray et al., 2019a] (pg.18-19).

C.3 Sensitivity to μ

We observed changing μ by small amounts (0.0001 to 0.0002) had no significant effect on results. Though, large changes in μ (0.0001 to 0.1) could lead to a substantial difference in behavior.

Figure 10: Entropy μ = 0: The above plot shows the performance of PCCES with μ = 0 (no constraint on entropy) vs PCCES with constraints on entropy for Ant-v0. These runs were conducted for 1000 updates and averaged over 5 seeds. This result clearly shows the benefit of μ > 0.
D Experiment Details

D.1 Hyperparameter Selection and Compute

For the step size, we conducted a grid search over the following values [0.1, 0.01, 0.001, 0.05]. We conducted 3 runs for each environment with a different seed and selected the step size with the overall best performance. For $\alpha$ and $\beta$ we used the default values mentioned in Maheswaranathan et al. (2018). We fixed the initial standard deviation of sampling distribution to 1.0 for all experiments. The remaining hyperparameters are given in table 1, 2 and 3.

All experiments for PCCES, OpenES, ASEBO were conducted on servers with only CPU’s, with number of CPU’s per experiment varied from 5 - 30 depending on the availability of the compute. For CPO, RCPO and PPO, We distributed all runs across 4 CPUs per run and 1 GPU (various GPUs including GTX 1080 Ti, TITAN X, and TITAN V.)

| Hyperparameters | PCCES | OpenES |
|-----------------|-------|--------|
| L-2 coefficient | 0.0001 | 0.0001 |
| $\lambda$       | 40, 80, 240 | 40, 80, 240 |
| $\sigma_F$      | 1.0   | 1.0    |
| $\sigma_0$      | 0.1   | 0.1    |
| decrease $\sigma$ rate | 0.99 | - |
| increase $\sigma$ rate | 1.01 | - |
| min $\sigma$    | 0.001 | -      |
| max $\sigma$    | 0.1   | -      |
| $\alpha$        | 0.5   | -      |
| $\beta$         | 5.0   | -      |
| discount factor  | 0.99  | -      |
| $\mu$           | 0.0001 | -     |
| $m$             | 20    | -      |
| $\kappa$        | 0.005 | -      |

Table 1: Hyper parameters for PCCES and OpenES for control tasks

| Environment               | Entropy Low | Entropy High |
|---------------------------|-------------|--------------|
| CartPole                  | 0           | 1000         |
| Acrobat                   | 0           | 1000         |
| MountainCarContinuous     | -1000       | 1000         |
| MountainCar               | 0           | 1000         |
| Pendulum                  | 0           | 1000         |
| InvertedPendulum          | -1000       | 3000         |
| InvertedDoublePendulum    | 0           | 1000         |
| Ant                       | 0           | 5000         |
| Hopper                    | 0           | 1000         |
| Walker                    | 0           | 2000         |
| CartPoleSafeDelayed-v0    | 0           | 2000         |
| MountainCarSafeDelayed-v0 | 0           | 2000         |
| Safexp-PointGoal1-v0      | -1000       | 5000         |
| Safexp-PointButton1-v0    | -1000       | 5000         |
| Safexp-CarGoal1-v0        | -1000       | 5000         |

Table 2: Entropy lower and upper bounds for all the envs

D.2 Software Libraries

We thank the developers of Tensorflow (Abadi et al., 2015), PyTorch (Paszke et al., 2019), OpenAI Gym (Brockman et al., 2016), Numpy (Harris et al., 2020), RLLib (Liang et al., 2018) and Matplotlib (Hunter, 2007).
Table 3: Thresholds for the cost penalty in the constrained environments

| Environment                      | Threshold |
|----------------------------------|-----------|
| CartPoleSafeDelayed-v0           | 30        |
| MountainCarSafeDelayed-v0        | 10        |
| Safexp-PointGoal1-v0             | 25        |
| Safexp-PointButton1-v0           | 25        |
| Safexp-CarGoal1-v0               | 25        |