DISTRIBUTED BLACK-BOX OPTIMIZATION OF NONCONVEX FUNCTIONS

Sergio Valcarcel Macua *Santiago Zazo* *Javier Zazo*

ABSTRACT

We combine model-based methods and distributed stochastic approximation to propose a fully distributed algorithm for nonconvex optimization, with good empirical performance and convergence guarantees. Neither the expression of the objective nor its gradient are known. Instead, the objective is like a "black-box", in which the agents input candidate solutions and evaluate the output. Without central coordination, the distributed algorithm naturally balances the computational load among the agents. This is especially relevant when many samples are needed (e.g., for high-dimensional objectives) or when evaluating each sample is costly. Numerical experiments over a difficult benchmark show that the networked agents match the performance of a centralized architecture, being able to approach the global optimum, while none of the individual noncooperative agents could by itself.

Index Terms— adaptive networks, cross-entropy, diffusion strategies, global optimization, stochastic approximation.

1. INTRODUCTION

Consider the problem in which a network of autonomous agents cooperate to optimize a nonconvex, possibly nondifferentiable, deterministic objective function $J : \mathbb{X} \to \Re$:

$$\underset{x \in \mathbb{X}}{\text{maximize}} \quad J(x) \tag{1}$$

where x is a vector of length M and $\mathbb{X} \subset \Re^M$ is a non-empty compact set of solutions. The agents do not know the analytical expression of the objective, rather, the objective function is like a "black-box", in which the agents input candidate solutions and evaluate the outputs.

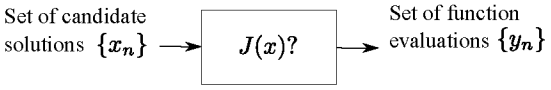

Fig. 1. Black-box optimization.

Although finding the global optimum of an unknown function with multiple local extrema is a difficult task, recent results on global optimization [1, 2, 3, 4, 5] show that a new class of randomized search algorithms, named *model-based* methods, can perform well even when the solution lies in a multidimensional space. Model-based methods learn a probabilistic model that concentrates its probability density around the set of optimal solutions, and sample candidate solutions from this model. Learning over the space of distributions enhances the exploration capabilities of the algorithms with respect to *instance-based* methods (like simulated annealing [6]), which optimize directly over the solution space.

One important example of model-based algorithms is the cross-entropy (CE) method [7, 8, 9]. We propose the first fully distributed implementation of CE, for which we only assume local communication between neighbors over a connected graph. Neither central node fusing information, nor load balancing server are required. All nodes have the same role and run the same algorithm to cooperatively learn the parameters of the probabilistic model. The algorithm uses diffusion strategies [10, 11, 12]. Diffusion have less computational complexity than incremental algorithms [13, 14], and have shown enhanced stability over consensus strategies [11, 12, 15]. Together with the robustness, reduced bias and variance, increased stability against node failure and intrinsic privacy (due to never exchanging local samples) of the diffusion-like distributed algorithms, an important extra feature highlighted by our method is that the computational load is naturally divided among the agents.

Many algorithms have been proposed for distributed *convex* optimization under different scenarios (noisy links, changing topologies, asynchronous iterations...) [16, 17, 18, 19, 20, 21]. Nevertheless, there are much fewer studies on *nonconvex* optimization. Although there are some useful theoretical works [22, 23] analyzing the convergence of diffusion and consensus *first-order* methods over nonconvex cost functions (indeed, we use results from [22] in this paper), first-order methods are not effective for optimizing black-box multidimensional multi-extrema objectives.

2. MODEL-BASED AND CROSS-ENTROPY METHODS

The aim of model-based methods is to find a probabilistic distribution g that assigns most of its probability density to the set of optimal solutions of (1), so that we obtain a new problem:

$$\underset{g \in \mathcal{D}}{\text{maximize}} \quad \int_{\mathbb{X}} J(x) g(x) dx \tag{2}$$

where $\mathcal{D}$ is the space of distributions. The problem of learning such optimal density can be formulated into the Bayesian filtering framework [24]. We can see the optimal solution as a latent variable, and the function evaluation of candidate solutions as noisy measurements of the unknown latent variable that we want to estimate. At every iteration, the estimated density represents our belief of where the optimal solution lies, given the candidate solutions that we have evaluated. More formally, consider the following state-space model

$$X_i = X_{i-1} \tag{3}$$

$$Y_i | (X_i = x_i) \sim h(y_i | x_i) \tag{4}$$

where subscript i denotes iteration counter, X is the latent optimal solution we want to estimate, Y denotes the outcome of the function evaluation and $h(y_i|x_i)$ is the likelihood or observation model. Let

g_{i-1} denote our belief of the set of optimal solutions at time $i-1$:

$$g_{i-1}(x) = \mathbb{P}(x|y_{0:i-1}) \tag{5}$$

where $y_{0:i-1}$ denotes the set of output values obtained until time $i-1$. Using Bayes rule, the belief density is updated as

$$g_i(x) = \mathbb{P}(x|y_{0:i-1}, y_i) = \frac{\mathbb{P}(y_i|x)g_{i-1}(x)}{\mathbb{P}(y_i|y_{0:i-1})} \tag{6}$$

At every iteration of a model-based algorithm, we draw independent and identically distributed (i.i.d.) samples $\{X_i = x_i\}$ from our latest belief g_{i-1}, obtaining output values $\{Y_i = y_i\}$ that are conditionally independent given the samples. Hence, the belief update can be expressed as

$$g_i(x) = \frac{h(y_i|x)g_{i-1}(x)}{\int h(y_i|x)g_{i-1}(x)dx} \tag{7}$$

Reference [24] (see also [25, Ch. 9.9]) shows that if the likelihood $h(y|x)$ is an increasing function of y, we obtain the following important monotonicity property of this approach:

$$\mathbb{E}_{g_i}[J(X)] = \frac{\mathbb{E}_{g_{i-1}}[J(X)h(y_i|X)]}{\mathbb{E}_{g_{i-1}}[h(y_i|X)]} \geq \mathbb{E}_{g_{i-1}}[J(X)] \tag{8}$$

Nevertheless, in order to build practical algorithms, we have to face two main challenges. First, it is difficult to sample from an arbitrary belief density (7) over a high-dimensional solution space. Second, although an increasing likelihood $h(y|x)$ is reasonable—because the higher the output value y, the higher the likelihood that such value has been generated by the optimal solution x^o—we have to design the specific form of h in such a way that the algorithm converges quickly, without getting trapped in a local optimum.

The cross-entropy method (CE) [7, 8] is one model-based algorithm that aims to surmount these two issues. CE can be seen as a particle filter implementation of (7), but, instead of sampling from an arbitrary belief density g_i, CE draws particles (i.e., resampling step) from a surrogate density. The surrogate sampling density is obtained by projecting the belief onto a parametrized family of densities $\{f_w(\cdot), w \in \mathbb{W}\}$, where w is the parameter, and $\mathbb{W} \subset \Re^M$ is the parameter space. The projection of the current belief is obtained by minimizing the Kullback-Leibler (KL) divergence (also known as *cross-entropy*) with the parametrized density. In order to guide the search to the most promising regions (i.e., exploitation), CE uses a narrow likelihood h that allocates most of the likelihood to a few set of elite samples with highest output values. Example of an increasing likelihood function is the smooth indicator function given by [4], which we denote $\mathcal{I}(y, \gamma)$:

$$h(y|x) \triangleq \mathcal{I}(y, \gamma) \propto \frac{1}{1 + e^{-\epsilon(y-\gamma)}} \tag{9}$$

where γ is the threshold that distinguishes the elite samples, and ϵ is a large positve parameter. Note that γ is critical to the performance of the algorithm. In our search for a global optimum, we want to exploit the best samples, but, at the same time, we have to continue exploring those areas that seem less promising. Reference [8] proposed a simple smoothing rule for trading exploration-exploitation, while [2] introduced a more efficient approach. In the following section, we introduce a fully distributed implementation of CE, which generalizes the single-agent algorithm proposed by [2] to a network of cooperative agents.

3. DISTRIBUTED CROSS-ENTROPY ALGORITHM

Consider a network of N agents. Each agent k updates its own belief, denoted $g_{k,i}$, and projects the result onto the family of parametrized densities f_w. Thus, the belief update (7) becomes:

$$g_{k,i}(x) = \frac{\mathcal{I}(y_{k,i}, \gamma_{k,i}) f_{w_{k,i-1}}(x)}{\mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X), \gamma_{k,i})]} \tag{10}$$

where $\gamma_{k,i}$ denotes a high quantile, estimated by agent k from its own set of output values, and $w_{k,i-1}$ is the parameter of its sampling density in the earliest iteration. There are reasons (e.g., privacy and communication overhead) to prevent agents from exchanging their data set of candidate solution and output value pairs. Therefore, we impose (10) to be performed locally by each agent.

We choose f_w to belong to the natural exponential family (NEF) of distributions (see, e.g., [9]), which can be expressed as

$$f_w(x) \triangleq e^{w^\top \phi(x) - Z(w)} \tag{11}$$

where $Z(w) \triangleq \log \int e^{w^\top \phi(x)} dx$ ensures normalization, and $\phi(x) : \mathbb{X} \to \Re^{M+M^2}$ is known as the natural sufficient statistic of x and contains all the first and second order moments:

$$\phi(x) \triangleq \left(x, \text{vec}\left[xx^\top\right]\right) \tag{12}$$

We wish the sampling distribution to approach the belief update. This can be achieved by minimizing their KL divergence, given by

$$\mathcal{D}(g_{k,i}, f_w) \triangleq \mathbb{E}_{g_{k,i}}\left[\log \frac{g_{k,i}(X)}{f_w(X)}\right] \tag{13}$$

In order for every agent to benefit from the learning process of the whole network, instead of locally minimizing (13), we propose a *collaborative smoothed projection* step, in which the agents minimize the sum of the KL divergences across the network:

$$\underset{w \in \mathbb{W}}{\text{minimize}} \;\; S_i(w) \triangleq \sum_{k=1}^{N} \mathcal{D}(g_{k,i}, f_w) \tag{14}$$

Introduce the optimal parameter that minimizes (14):

$$w_i^o \triangleq \arg\min_{w \in \mathbb{W}} S_i(w) \tag{15}$$

such that $\nabla S_i(w_i^o) = 0$. When f_w belongs to the NEF, it is well known that $\nabla Z(w) = \mathbb{E}_w[\phi(X)]$. Then, from (14), we have:

$$\begin{aligned}\nabla S_i(w_i^o) &= \nabla\left[\sum_{k=1}^{N} \mathbb{E}_{g_{k,i}}[\log g_{k,i}(X)] + Z(w_i^o) - \mathbb{E}_{g_{k,i}}^\top[\phi(X)]w_i^o\right] \\ &= \sum_{k=1}^{N} \mathbb{E}_{w_i^o}[\phi(X)] - \sum_{k=1}^{N} \mathbb{E}_{g_{k,i}}[\phi(X)] = 0\end{aligned} \tag{16}$$

Hence, the optimal sufficient statistics of the sampling distribution that solves (14) is given by

$$\mathbb{E}_{w_i^o}[\phi(X)] = \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}_{g_{k,i}}[\phi(X)] \tag{17}$$

Since the agents do not know the terms $\mathbb{E}_{g_{k,i}}[\phi(X)]$, they have to rely on Monte Carlo estimates. At every iteration i, each agent draws a local set of samples from its local model $f_{w_{k,i-1}}$. These samples are evaluated—in a black-box fashion—so that we obtain a set of

candidate solution and output value pairs $\{(x,y) \in \Lambda_{k,i}\}$. The elite-threshold $\gamma_{k,i}$ is estimated as the the q-th highest output value, which can be seen as a Monte Carlo estimate of the $q/|\Lambda_{k,i}|$-th quantile of the distribution of output values. From (10), the term $\mathbb{E}_{g_{k,i}}[\phi(X)]$ is given by:

$$\mathbb{E}_{g_{k,i}}[\phi(X)] = \frac{\mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})\phi(X)]}{\mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})]} = \frac{\sum_{(x,y)\in\Lambda_{k,i}} \mathcal{I}(y,\gamma_{k,i})\phi(x)}{\sum_{(x,y)\in\Lambda_{k,i}} \mathcal{I}(y,\gamma_{k,i})} + \xi_{k,i} \quad (18)$$

where $\xi_{k,i}$ is the Monte Carlo noise. The main benefit from cooperation is that the agents can combine their local estimators to reduce the approximation noise in (18).

Note that if the agents choose a high $\gamma_{k,i}$, their belief (10) will move towards the most promising areas of the solution space so quickly that they may be trapped by a local maximum of (1). Therefore, in order to enhance exploration, we extend the ideas in [2] and propose the agents to execute a *diffusion-based distributed stochastic approximation* [26] to approach (17) with little steps:

$$\widehat{\theta}_{k,i} = \theta_{k,i-1} - \alpha_i \left(\theta_{k,i-1} - \frac{\sum_{(x,y)\in\Lambda_{k,i}} \mathcal{I}(y,\gamma_{k,i})\phi(x)}{\sum_{(x,y)\in\Lambda_{k,i}} \mathcal{I}(y,\gamma_{k,i})} \right)$$
$$\theta_{k,i} = \sum_{l\in\mathbb{N}_k} b_{lk}\widehat{\theta}_{l,i} \quad (19)$$

where α_i is the step-size, $\mathbb{N}_k$ denotes the neighborhood of agent k (i.e., the set of nodes that can share information with node k, including itself), and b_{lk} denotes the weight given by agent k to information coming from its neighbor l. Note that (19) performs a stochastic approximation in the space of sufficient statistics. If the sampling distribution is initialized with high variance, the objective in (2) will be almost flat. In succeeding iterations, by assigning a small step-size α_i to the filter update, the real shape of the original objective in (1) will be slowly revealed, giving time to the stochastic approximation to ascend through its envelope, avoiding local peaks.

In the following section, we show conditions under which every node executing (19) will asymptotically approach (17).

4. CONVERGENCE ANALYSIS

First, using earlier results on distributed stochastic approximation due to [22], we show that the sequence obtained from (19) is related to a mean ordinary differential equation (ODE). Then, we show that this ODE is similar to the one analyzed by [2] for the single-agent CE algorithm. We require the following assumptions:

Assumption 1 (Combination weights)**.** *Let $B = [b_{lk}]$ be the $N \times N$ matrix including the set of combination coefficients, then:*

$$b_{lk} \geq 0, \;\; b_{lk} = 0 \;\; \textit{if } l \notin \mathcal{N}_k, \;\; B^\top \mathbb{1}_N = \mathbb{1}_N, \;\; B\mathbb{1}_N = \mathbb{1}_N \quad (20)$$
$$\rho\left(B\left(I_N - \mathbb{1}_N\mathbb{1}_N^\top/N\right)B^\top\right) < 1 \quad (21)$$

Assumption 2 (Smoothing parameters)**.** *The sequence $\{\alpha_i\}$ satisfies: $\alpha_i > 0 \; \forall i$, $\lim_{i\to\infty} \alpha_i = 0$ and $\sum_{i=0}^{\infty} \alpha_i = \infty$.*

Assumption 3 (Continuity)**.** *$\nabla \log \mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})]$ is continuous with a unique integral curve.*

Assumption 4 (Number of samples)**.** *The number of samples per node at time i satisfies $|\Lambda_{k,i}| \geq i^\beta/N$, for some $\beta > 1$.*

Assumption 5 (Unique quantile)**.** *The quantile $q \in (0,1)$ of $\{J(X),\, X \sim f_w(X)\}$ is unique for each $w \in \mathbb{W}$.*

The first condition in (20) is imposed by the network topology and enforces the agents to only communicate with their neighbors (i.e., multihop communications are not allowed). The other conditions in (20) ensure that B is doubly-stochastic. Condition (21) is easily satisfied in practice (e.g., by setting Metropoli-Hasting weights [27] in a connected graph). Assumption 2 is standard for the analysis of stochastic approximations with the ODE method. Assumption 3 ensures that the related ODE (see (31) below) is well posed. Assumptions 4 and 5 are used by [2][Lemma 3.1] to prove asymptotically diminishing Monte Carlo noise.

Introduce the network vectors of length MN:

$$\theta_i \triangleq \operatorname{col}\{\theta_{1,i},\ldots,\theta_{N,i}\} \quad (22)$$
$$\pi_i \triangleq \operatorname{col}\{\mathbb{E}_{g_{1,i}}[\phi(X)],\ldots,\mathbb{E}_{g_{N,i}}[\phi(X)]\} \quad (23)$$
$$\xi_i \triangleq \operatorname{col}\{\xi_{1,i},\ldots,\xi_{N,i}\} \quad (24)$$

Collect the updates (19) for all agents into a network recursion:

$$\theta_i = (B^\top \otimes I_M)(\theta_{i-1} - \alpha_i(\theta_{i-1} - \pi_i + \xi_i)) \quad (25)$$

From (20), note that $(\mathbb{1}_N^\top \otimes I_M)(B^\top \otimes I_M) = (\mathbb{1}_N^\top B^\top \otimes I_M) = (\mathbb{1}_N^\top \otimes I_M)$. Introduce the network average operator $\langle\cdot\rangle : \Re^{MN} \to \Re^M$, such that $\langle u\rangle = (\mathbb{1}^\top \otimes I_M)u = 1/N\sum_{k=1}^N u_k$, for vectors $\{u_k \in \Re^M\}_{k=1}^N$ and $u = \operatorname{col}\{u_1,\ldots,u_N\} \in \Re^{MN}$. By applying this operator to (25), we obtain:

$$\langle\theta_i\rangle = \langle\theta_{i-1}\rangle - \alpha_i\left(\langle\theta_{i-1} - \pi_i\rangle + \langle\xi_i\rangle\right) \quad (26)$$

In order to expand the term $\langle\theta_{i-1} - \pi_i\rangle$, note that, for the NEF distributions (11), we can derive the following equivalence:

$$\theta_{k,i-1} - \mathbb{E}_{g_{k,i}}[\phi(X)] = \frac{\mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})(\theta_{k,i-1} - \phi(X))]}{\mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})]} = \nabla \log \mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})] \quad (27)$$

Introduce the shorthand

$$G(\theta_{k,i-1}) \triangleq \log \mathbb{E}_{w_{k,i-1}}[\mathcal{I}(J(X),\gamma_{k,i})] \quad (28)$$

Define

$$r_i \triangleq \nabla G(\langle\theta_{i-1}\rangle) - 1/N\sum_{k=1}^N \nabla G(\theta_{k,i-1}) \quad (29)$$

Hence, from (26), we obtain the following mean network recursion:

$$\langle\theta_i\rangle = \langle\theta_{i-1}\rangle + \alpha_i(\nabla G(\langle\theta_{i-1}\rangle) - r_i - \langle\xi_i\rangle) \quad (30)$$

Thus, we can see $\langle\theta_i\rangle$ as a noisy discrete approximation of the ODE

$$\dot{\theta} = \nabla G(\theta) \quad (31)$$

We are ready to state the main results of this section:

Theorem 1 (Convergence of network recursion)**.** *If Assumptions 1–5 hold, the sequence $\{\langle\theta_i\rangle\}$ generated by* (30) *converges almost surely to the solution set of* (31)*.*

Proof. Under Assumptions 1–3, we can use [22][Proposition 1] to show that r_i converges almost surely towards zero. Therefore, (30) follows the same asymptotic behavior of the single-agent CE algorithm analyzed by [2][Theorem 3.1]. □

Corollary 1 (Convergence of the individual agents). *If Assumptions 1–5 hold, and the solution set of* (31) *are isolated equilibrium points, then the sequences of individual moments* $\{\theta_{k,i}\}$ *generated by* (19) *converge almost surely to the aggregated* $\mathbb{E}_{w_i^o}[\phi(X)]$ *in* (17).

Proof. Under Assumptions 1–2, Lemma 1 in [22] proves that the agents asymptotically reach an agreement on their estimate. Therefore, the asymptotic analysis of the individual estimates $\theta_{k,i}$ reduces to study their average $\langle\theta_i\rangle = 1/N\sum_{k=1}^{N}\theta_{k,i}$, which, by Theorem 1, we know that converges to one equilibrium $\langle\theta^o\rangle$, such that $\nabla G(\langle\theta^o\rangle) = 0$. As stated in Theorem 1, $\lim_{i\to\infty} r_i = 0$. Thus, from (29), $\lim_{i\to\infty}\left(\nabla G(\langle\theta_i\rangle) - 1/N\sum_{k=1}^{N}\nabla G(\theta_{k,i})\right) = 0$. From (27) and (17), it follows that every agent converges to

$$\begin{aligned}\langle\theta^o\rangle &= \lim_{i\to\infty} 1/N\sum_{k=1}^{N}\theta_{k,i} = \lim_{i\to\infty} 1/N\sum_{k=1}^{N}\mathbb{E}_{g_{k,i}}[\phi(X)]\\ &= \lim_{i\to\infty}\mathbb{E}_{w_i^o}[\phi(X)]\end{aligned}$$ □

5. NUMERICAL EXPERIMENTS

For the single agent case, model-based methods have already been tested in difficult benchmarks, achieving state-of-the art performance [2, 3, 4, 9]. Here, our purpose is to compare the results of the Distributed CE (DCE) algorithm given by (19) with the single-agent CE (SACE) presented in [2]. Emulating the numerical experiments in [2] for SACE, we particularize DCE for multivariate Normal distributions. From (19), by using the one-to-one relationship between the natural moments $\theta_{k,i}$ and the mean $\mu_{k,i}$ and covariance $\Sigma_{k,i}$, we obtain the following *distributed* iterations:

$$\widehat{\mu}_{k,i} = \mu_{k,i-1} - \alpha_{k,i}\left(\mu_{k,i-1} - \frac{\sum_{(x,y)\in\Lambda_{k,i}}\mathcal{I}(y,\gamma_{k,i})\cdot x}{\sum_{(x,y)\in\Lambda_{k,i}}\mathcal{I}(y,\gamma_{k,i})}\right)$$

$$\mu_{k,i} = \sum_{l\in\mathbb{N}_k} b_{lk}\widehat{\mu}_{l,i} \tag{32}$$

$$\begin{aligned}\widehat{\Sigma}_{k,i} = &(1-\alpha_{k,i})\left(\Sigma_{k,i-1} + (\mu_{k,i-1}-\mu_{k,i})(\mu_{k,i-1}-\mu_{k,i})^\top\right)\\ &+\alpha_{k,i}\left(\frac{\sum_{(x,y)\in\Lambda_{k,i}}\mathcal{I}(y,\gamma_{k,i})(x-\mu_{k,i})(x-\mu_{k,i})^\top}{\sum_{(x,y)\in\Lambda_{k,i}}\mathcal{I}(y,\gamma_{k,i})}\right)\end{aligned}$$

$$\Sigma_{k,i} = \sum_{l\in\mathbb{N}_k} b_{lk}\widehat{\Sigma}_{l,i} \tag{33}$$

We consider three scenarios, namely, distributed, noncooperative and centralized. The distributed setting consists of a network of $N = 10$ agents, with average number of neighbors $|\mathbb{N}_k| = 3$ and Metropoli-Hasting [27] combination weights b_{lk}, in which every agent runs DCE, evaluating $|\Lambda_{k,i}|$ samples per iteration. In the centralized case, a single-agent running SACE invests the same computational effort invested by the whole distributed network, evaluating $|\Lambda_i| = N\cdot|\Lambda_{k,i}|$ samples per iteration. The noncooperative scenario consists of a set of $N = 10$ isolated (single) agents running SACE, but each one evaluating only $|\Lambda_{k,i}|$ samples per iteration.

We test DCE with a benchmark consisting of 9 different nonconvex problems—including objective functions with multiple extrema, high dimensionality and bad scaling—for which we know the global solution. The list of functions is in Table 1 and their definitions are in [3] (see Fig. 2 for an example of the objectives used in this benchmark). The search space is a hypercube centered at the origin and whose sides cover the interval $[-100, 100]$ along every axis.

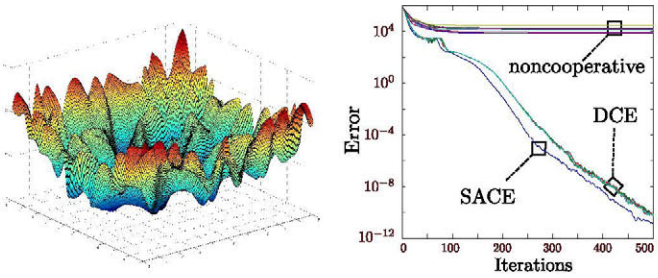

Fig. 2. (left) 2-D plot of the 20-D Pinter's function. (right) Distance to the global solution of the Pinter's function, achieved in a single experiment by centralized SACE, 10 collaborative nodes runing DCE and 10 noncooperative nodes running SACE.

Problem	M	SACE	DCE	Noncoop.
1. Dejong 5th	2	$6e-12$	$6e-12$	4.1
2. Shekel	4	$4e-6$	$4e-6$	1.4
3. Rosenbrock	20	$4e-10$	$4e-10$	458.9
4. Powel singular	20	$7e-14$	$6e-13$	$1e+7$
5. Trigonometric	20	$1e-14$	$1e-13$	936.0
6. Griewank	20	$2e-17$	$6e-16$	1.0
7. Pinter	20	1.5 (*)	$9e-11$	$2e+4$

Table 1. Distance to the global solution averaged over 50 independent experiments. For DCE and noncooperative scenarios, results are averaged over all agents.

One practical benefit ot SACE is that it requires little tuning. This feature remains in the proposed DCE. For all problems, we use the following parameters that aim to establish an effective exploration-exploitation tradeoff [2]. The smoothing parameter is $\alpha_i = 2/(i+100)^{0.501}$. The number of samples per iteration is $|\Lambda_{k,i}| = \max(50, i^{1.01})$ for DCE and noncooperative agents, and $|\Lambda_i| = 10\cdot|\Lambda_{k,i}|$ for the centralized SACE. The initial mean $\mu_{k,0}$ is drawn randomly from a uniform distribution over the whole search space, independently for all agents. The initial covariance is $\Sigma_{k,0} = 1000\cdot I_M$ equal for all agents.

Our performance criterion is the distance to the global solution. We run 50 independent experiments, with 500 iterations each. Table 1 shows that, in most of the cases, cooperative agents match closely the performance of the centralized algorithm but using N times less samples each. (*) We have observed that, when the agents have different starting point, DCE improves its exploration capabilities and avoid local maximum more efficiently than centralized SACE. For instance, for Pinter's problem, the centralized SACE achieved an average distance of $1e{-}11$ in all but one experiment (so the average error increases upto 1.5), while all DCE agents were able to find the global solution in all experiments.

6. CONCLUSIONS

We applied diffusion strategy for distributed implementation of the Cross-Entropy method proposed by [2], which works in any arbitrary connected topology with no central coordinator, allowing within neighborhood communication only, and without exchanging samples. We proved that agents running DCE converges to a local optimum almost surely. Numerical experiments show that the computational load is efficiently divided among the agents, achieving good results even in 20-dimensional problems, with just 50 samples per iteration in each of the DCE nodes.

7. REFERENCES

[1] J. Hu, Y. Wang, E. Zhou, M. C. Fu, and S. I. Marcus, "A survey of some model-based methods for global optimization," in *Optimization, Control, and Applications of Stochastic Systems*. Springer, 2012, pp. 157–179.

[2] J. Hu, P. Hu, and H. S. Chang, "A stochastic approximation framework for a class of randomized optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 165–178, January 2012.

[3] J. Hu, M. C. Fu, and S. I. Marcus, "A model reference adaptive search method for global optimization," *Operations Research*, vol. 55, no. 3, pp. 549–568, June 2007.

[4] E. Zhou and J. Hu, "Gradient-based adaptive stochastic search for non-differentiable optimization," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1818–1832, July 2014.

[5] X. Chen and E. Zhou, "Population model-based optimization with sequential monte carlo," in *Simulation Conference (WSC), 2013 Winter*, Dec 2013, pp. 1004–1015.

[6] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

[7] R. Rubinstein and D. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004.

[8] D. Kroese, S. Porotsky, and R. Rubinstein, "The cross-entropy method for continuous multi-extremal optimization," *Methodology and Computing in Applied Probability*, vol. 8, no. 3, pp. 383–407, 2006.

[9] D. P. Kroese, T. Taimre, and Z. I. Botev, *Handbook for Monte Carlo methods*. Wiley, 2011.

[10] A. H. Sayed, "Diffusion adaptation over networks," in *E-Reference Signal Processing*, R. Chellapa and S. Theodoridis, Eds. Elsevier, 2013. Also available as arXiv:1205.4220v1, May 2012.

[11] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 155–171, May 2013.

[12] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, April 2014.

[13] A. Nedic and D. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[14] M. Rabbat and R. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 798–808, April 2005.

[15] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6217–6234, 2012.

[16] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[17] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, January 2011.

[18] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[19] M. Zhu and S. Martinez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 151–164, Jan 2012.

[20] J. Chen and A. H. Sayed, "Distributed Pareto optimization via diffusion strategies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 205–220, Apr 2013.

[21] T.-H. Chang, A. Nedic, and A. Scaglione, "Distributed constrained optimization by consensus-based primal-dual perturbation method," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1524–1538, June 2014.

[22] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 391–405, Feb 2013.

[23] S. Ram, A. Nedic, and V. Veeravalli, "Asynchronous gossip algorithms for stochastic optimization," in *Int. Conf. Game Theory for Networks*, May 2009, pp. 80–81.

[24] E. Zhou, M. Fu, and S. Marcus, "Particle filtering framework for a class of randomized optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 1025–1030, April 2014.

[25] S. Ross, *Simulation*. Academic Press, 2013.

[26] J. Chen and A. H. Sayed, "On the limiting behavior of distributed optimization strategies," in *Proc. Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, October 2012, pp. 1535–1542.

[27] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems and Control Letters*, vol. 53, pp. 65–78, 2004.