Optimization Letters manuscript No.
(will be inserted by the editor)

Variable sample size method for equality constrained optimization problems

Nataša Krejić · Nataša Krklec Jerinkić · Andrea Rožnjik

Received: date / Accepted: date

Abstract An equality constrained optimization problem with a deterministic objective function and constraints in the form of mathematical expectation is considered. The constraints are transformed into the Sample Average Approximation form resulting in deterministic problem. A method which combines a variable sample size procedure with line search is applied to a penalty reformulation. The method generates a sequence that converges towards first-order critical points. The final stage of the optimization procedure employs the full sample and the SAA problem is eventually solved with significantly smaller cost. Preliminary numerical results show that the proposed method can produce significant savings compared to SAA method and some heuristic sample update counterparts while generating a solution of the same quality.

Keywords stochastic optimization · equality constraints · variable sample size · penalty method · line search

N. Krejić and N. Krklec Jerinkić are supported by Serbian Ministry of Education, Science and Technological Development, grant no. 174030.

Nataša Krejić
Department of Mathematics and Informatics, University of Novi Sad,
Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia
E-mail: natasak@uns.ac.rs

Nataša Krklec Jerinkić
Department of Mathematics and Informatics, University of Novi Sad,
Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia
E-mail: natasa.krklec@dmi.uns.ac.rs

Andrea Rožnjik
Faculty of Civil Engineering, University of Novi Sad,
Kozaračka 2a, 24000 Subotica, Serbia
E-mail: andrea@gf.uns.ac.rs

1 Introduction

We consider the following equality constrained optimization problem

$$\min_x f(x), \quad \text{subject to } h(x) = 0, \tag{1}$$

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is deterministic and the constraints are in the form of mathematical expectation, i.e. $h(x) = E(H(x,\xi))$ where $H : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m$, ξ is a random vector $\xi : \Omega \to \mathbb{R}^p$ and $(\Omega, \mathscr{F}, P)$ is a probability space.

In general it is difficult to compute the mathematical expectation and the common approach is to generate a sample of random vectors and replace the expectation function by the sample average function. In variable sample methods, [10], different samples are used along the optimization process. Another approach is to fix a sample (rather large in general) at the beginning of the optimization procedure, so the stochastic problem is converted into a deterministic problem. This approach is known as the Sample Average Approximation (SAA) or the sample path, details can be found for example in [17,18]. The obtained SAA problem can be solved by standard optimization techniques. Since the sample often needs to be large to ensure a good approximation of the mathematical expectation, the sample average function (and possibly its gradient) is expensive to evaluate and thus solving the SAA problem is expensive. One possible way to eliminate this drawback is to vary a sample size throughout the optimization process. Namely, when the current iteration point is far from the solution, a smaller sample size can give an iteration point which is good enough and therefore reduce a number of function evaluations. Some methods for controlling the sample size are presented in [4,10]. Roughly speaking the optimization method starts with a small size subsample and increases the subsample throughout the iterations. An alternative approach, which can be classified as adaptive, relays on the progress achieved in each iteration and thus allows sample size to oscillate until eventually working with the full sample, [1,2,12,13].

The approach we consider in this paper is based on penalty methods, [8,14,11,6]. Penalty methods are successfully applied in stochastic environment. In [16,20] an exact penalty method is used to solve a stochastic optimization problem with expected value objective function and deterministic constraints. Polak and Royset, in [16], considered the problem with inequality constraints and proposed an algorithm for solving the SAA reformulation for sufficiently large penalty parameter. Constraints in [20] are defined in form of equalities and the rule for varying the penalty parameter is defined through minimization of a subproblem.

We propose an algorithm for solving the SAA reformulation of the problem (1). As an optimization procedure we apply the quadratic penalty method combined with the variable sample size scheduling and line search technique. We show that if Linear Independence Constraint Qualification (LICQ) holds, then the proposed algorithm generates a sequence that converges to a Karush-Kuhn-Tucker (KKT) point of the SAA problem. The algorithm is implemented on a set of test problems from [9] with added noise, and the numerical results show that the proposed algorithm requires a significantly smaller number of function evaluation than the full sample SAA method as well as some heuristic procedure.

The rest of the paper is structured as follows. In the next section details of the observed problem and the algorithm are presented. The convergence results are stated in Section 3. Section 4 contains numerical results. Conclusions and some details that complete the paper are given in the last two sections.

Throughout this paper $\|\cdot\|$ denotes the Euclidian norm.

2 The Algorithm

From now on, we consider the SAA reformulation of problem (1), i.e.

$$\min_x f(x), \quad \text{subject to } \hat{h}_{N_{max}}(x) = 0, \tag{2}$$

where the Sample Average Approximation for any $N \in \mathbb{N}$, $N \leq N_{max}$ is

$$\hat{h}_N(x) = \frac{1}{N}\sum_{i=1}^{N} H(x,\xi_i)$$

and the full sample $\xi_1, \xi_2, \ldots, \xi_{N_{max}}$ is given in advance. Even if the problem (1) is feasible, (2) may be infeasible for a particular sample $\{\xi_1, \ldots, \xi_{N_{max}}\}$. Thus we must assume the existence of a solution. Although such assumption might appear strong, it is rather common to impose it (see Section 5.1 in [18] for instance). Moreover, the objective function is assumed to be bounded from bellow on the feasible set given by (2).

The method presented here aims to solve (2) by exploiting an efficient sample size update combined with an update of the penalty parameter μ of the penalty function

$$\phi(x;N;\mu) = f(x) + \mu\,\hat{\theta}_N(x), \quad \text{with} \quad \hat{\theta}_N(x) = \|\hat{h}_N(x)\|^2. \tag{3}$$

The penalty function (3) is defined for an arbitrary $N \leq N_{\max}$. Thus at each iteration k we are dealing with a sample size N_k and a penalty parameter μ_k, i.e. with the function $\phi(x,N_k,\mu_k)$. It is assumed here that we are taking the first N_k elements of the initially generated sample $\{\xi_1, \ldots, \xi_{N_{\max}}\}$, i.e., we are using cumulative sample, see [10]. While the sequence of penalty parameters μ_k is nondecreasing, the sample sizes N_k may oscillate depending on the progress of the algorithm. Roughly speaking, μ_k is increased when a stationary point of function $\phi(x;N_k;\mu_k)$ is approached. Notice that N_k changes through iterations and an additional scheduling procedure is needed. Thus we are using different penalty functions in each iteration instead of $\phi(x;N_{max};\mu_k)$. The sample size update is based on two error measures denoted by dm_k and $\varepsilon_\delta^{N_k}(x_k)$. Both of them have to be nonnegative and $\varepsilon_\delta^{N_k}(x_k)$ is assumed to be bounded away from zero. The first one, dm_k measures the distance from the stationary point of $\phi(x;N_k;\mu_k)$, while the second one approximates the distance from the target (full sample) problem, i.e., it estimates the error of the approximation $\hat{h}_{N_k} \approx \hat{h}_{N_{max}}$. We assume that the gradient $g_k := \nabla\phi(x_k;N_k;\mu_k)$ is available, thus we define $dm_k = dm_k(\alpha_k) = -\alpha_k g_k^T d_k$, where α_k is a step length and d_k is assumed to be a descent search direction for $\phi(x_k,N_k,\mu_k)$. There are other possibilities for choosing

dm_k, see [13] for instance. The estimator $\varepsilon_\delta^{N_k}(x_k)$ can be chosen in many ways, but the common choice is the following sample variance. For instance,

$$\varepsilon_\delta^{N_k}(x_k) = \hat{\sigma}_{N_k}(x_k)1.96/\sqrt{N_k}, \tag{4}$$

where $\hat{\sigma}^2_{N_k}(x_k) = 1/(N_k-1)\sum_{i=1}^{N_k}\|H(x_k,\xi_i) - \hat{h}_{N_k}(x_k)\|^2$.

The rule for changing the sample size is taken over from [13] with specified weighting parameter, i.e. the aim is to find a sample size N_{k+1} such that $dm_k \approx N_k/N_{k+1}\,\varepsilon_\delta^{N_{k+1}}(x_k)$ and $N_k^{min} \le N_{k+1} \le N_{max}$, where $\{N_k^{min}\}$ is a lower bound sequence. This sequence is updated as in [13], but instead of the objective function we observe the measure of infeasibility $\hat{\theta}_{N_{k+1}}$. These algorithms are stated in the Appendix for completeness, while here we give only a brief discussion.

The main idea behind the sample size updating is as follows. A relatively small value of dm_k suggest the proximity of a solution to the current approximate problem (i.e. a stationary point of $\phi(x,N_k,\mu_k)$) and thus the sample size is increased, to get a better approximation of (2). On the other hand, if dm_k is relatively large, the sample size is decreased (but not below N_k^{min}) in order to save computational efforts since we are probably still far away from the solution. That way, the algorithm copes with different kinds of approximations simultaneously.

Although the lower bound N_k^{min} update does not interfere within the main sample size update in most of the tested applications, it plays an important role in the convergence theory. Its main role is to prevent permanent oscillations of the sample size and to push it to the full sample, eventually. This is done by tracking different levels of precision determined by the sample size, or more precisely, by the function $\hat{\theta}_N$. If a sample size is increased to some precision, let us say N_k, and if there is not enough decrease in measure of infeasibility $\hat{\theta}_{N_k}$ since the last time that this same level of precision has been used, the lower bound is increased. Consequently, it pushes up the overall precision controlled by the sample size. The main algorithm is stated below. Note that, a sample $\xi_1,\xi_2,\ldots,\xi_{N_{max}}$ is generated at the beginning of the optimization procedure. Throughout all iterations with the sample size $N_k < N_{max}$, the first N_k elements of the whole sample are used. This way we are dealing with the so-called cumulative samples, see [10]. The line search is performed in Step 3 as the classical backtracking with the step β but other options, like interpolation and similar are possible as well.

Algorithm 1

Step 0 Input parameters: $N_{min} \in \mathbb{N}$, $x_0 \in \mathbb{R}^n$, $\beta,\eta,\nu_1 \in (0,1)$, $\mu_0 > 0$, $\gamma > 1$.

Step 1 Set $k = 0$, $N_k = N_{min}$, $x_k = x_0$, $\mu_k = \mu_0$, $l = 1$, $N_0^{min} = N_{min}$.

Step 2 Calculate a descent search direction d_k.

Step 3 Find the smallest nonnegative integer j such that $\alpha_k = \beta^j$ satisfies

$$\phi(x_k + \alpha_k d_k; N_k; \mu_k) \le \phi(x_k; N_k; \mu_k) - \eta dm_k(\alpha_k). \tag{5}$$

Set $x_{k+1} = x_k + \alpha_k d_k$ and $dm_k = dm_k(\alpha_k)$.

Step 4 If $dm_k \le \alpha_k/\mu_k^2$, set $z_t = x_k$ and $t = t+1$.

Step 5 Determine the sample size N_{k+1} using Algorithm 2.

Step 6 Determine the lower bound of the sample size N_{k+1}^{min} using Algorithm 3.

Step 7 Determine the penalty parameter μ_{k+1}:
If $N_k = N_{k+1} < N_{max}$ or $dm_k > \alpha_k/\mu_k^2$, then $\mu_{k+1} = \mu_k$, else $\mu_{k+1} = \gamma\mu_k$.
Step 8 Set $k = k+1$ and go to Step 2.

One additional comment regarding the Algorithm above is due here. Notice that in Step 2 we are taking an arbitrary descent direction d_k. Thus the Algorithm represents a general framework and some of it properties, like convergence rate for example, will be determined by a particular search direction used in actual implementation. The details of Algorithm 2 and 3 are available in the Appendix.

3 Convergence analysis

To show the convergence of Algorithm 1 we need the following standard assumption.

Assumption 1 *Function f is bounded from bellow on a feasible set given in (2). Moreover, $f, H(\cdot,\xi_i) \in C^1(\mathbb{R}^n)$ for every $i = 1,2,\ldots,N_{max}$ and the sequence $\{x_k\}_{k\in\mathbb{N}_0}$ generated by Algorithm 1 has at least one accumulation point.*

Assumption 1 provides continuity and differentiability of the function $\hat{h}_N$ for every $N \in \mathbb{N}$ and therefore ensures that the penalty function (3) is continuously differentiable. Moreover, the measure of infeasibility $\hat{\theta}_N$ is nonnegative so the penalty function (3) is also bounded from below whenever f is. Furthermore, notice that fixing the penalty parameter and the sample size to $\bar{\mu}$ and $\bar{N}$, respectively, yields a standard backtracking line search method applied on $\phi(x;\bar{N};\bar{\mu})$. Therefore, using the standard technique (see [12] for instance), we can prove the following lemma.

Lemma 1 *Suppose that the Assumption 1 holds and there exists $\bar{n} \in \mathbb{N}$ such that $\mu_k = \bar{\mu}$ and $N_k = \bar{N}$ for all $k \geq \bar{n}$. Then $\lim_{k\to\infty} dm_k = 0$.*

Proof Define $\phi(x) := \phi(x;\bar{N};\bar{\mu})$ and let x^* be an arbitrary accumulation point of the sequence $\{x_k\}_{k\in\mathbb{N}_0}$, i.e. $x^* = \lim_{j\to\infty} x_{k_j}$ for some subsequence $\{x_{k_j}\}_{j\in\mathbb{N}_0} \subseteq \{x_k\}_{k\in\mathbb{N}_0}$. Without loss of generality we can assume that $k_j \geq \bar{n}$ for every j. The convergence of $\{x_{k_j}\}_{j\in\mathbb{N}_0}$ and the boundedness from below, Assumption 1, implies the existence of a constant M such that $\phi(x_{k_j}) \geq M$ for every j. Furthermore, using the line search rule (Step 3, Algorithm 1) we obtain the following inequality which holds for every j

$$M \leq \phi(x_{k_j}) \leq \phi(x_{k_0}) - \eta \sum_{i=k_0}^{k_j-1} dm_i.$$

Letting j tend to infinity we obtain the result. □

The proof of the following lemma leans on the proof of Lemma 4.1 in [12]. Nevertheless, we provide the proof for completeness.

Assumption 2 *There are $\kappa > 0$ and $n_1 \in \mathbb{N}$ such that $\varepsilon_\delta^{N_k}(x_k) \geq \kappa$ for every $k \geq n_1$.*

Lemma 2 *Suppose that the Assumptions 1-2 hold. Then there exists $q \in \mathbb{N}$ such that $N_k = N_{max}$ for every $k \geq q$.*

Proof Suppose that there exists $\bar{n} > n_1$ such that for all $k \geq \bar{n}$ we have $N_k = N^1 < N_{max}$. Then the updating rule for the penalty parameter (Step 7, Algorithm 1) implies that the penalty parameter also remains fixed and Lemma 1 implies

$$\lim_{k\to\infty} dm_k = 0.$$

Consequently, $dm_k < \nu_1\kappa$ holds for all k large enough where parameters $\nu_1 \in (0,1)$ and $\kappa > 0$ are given in Algorithm 2 and Assumption 2, respectively. Since $\varepsilon_\delta^{N_k}(x_k) \geq \kappa$, it holds that $dm_k < \nu_1\varepsilon_\delta^{N_k}(x_k)$, and Step 1 3) ii) of Algorithm 2 implies that the sample size will be increased, which is clearly a contradiction.

On the other hand, assume that the sample size permanently oscillates. In that case we know that the lower bound N_k^{min} is smaller that N_{max} as otherwise we would have $N_k \geq N_k^{min} = N_{max}$ for all k sufficiently large. Therefore, the lower bound remains constant after a finite number of iterations, i.e., $N_{k+1}^{min} = N_k^{min}$ for every k large enough. Denote the maximal sample size that is used infinitely many times by $\bar{N}$ and notice that there are infinitely many iterations $k_1, k_2, \ldots$ in which the sample size is increased on $\bar{N}$. (There are only finitely many iterations in which the sample size is decreased on $\bar{N}$ since otherwise $\bar{N}$ would not be the largest sample size that is used infinitely many times. Therefore, we may assume that the decrease on $\bar{N}$ does not happen after k_1.) More precisely, there exists a subsequence $k_1, k_2, \ldots$ such that $N_{k_i} < N_{k_i+1} = \bar{N}$ and the lower bound remains unchanged, i.e., $N_{k_i+1}^{min} = N_{k_i}^{min}$ for all $i = 1, 2, \ldots$. Considering Algorithm 3, this corresponds to Step 2), case i) or ii). Obviously, if we exclude the first member of the subsequence, we eliminate case i) which leads us to a conclusion that the big enough decrease of function $\hat{\theta}_{\bar{N}}$ occurs in $k_2, k_3, \ldots$ Also, notice that $l(k_i) = k_{i-1} + 1$. Thus, for every $i = 2, 3, \ldots$ there holds

$$\hat{\theta}_{\bar{N}}(x_{k_{(i-1)}+1}) - \hat{\theta}_{\bar{N}}(x_{k_i+1}) \geq \frac{\bar{N}}{N_{max}} \left(k_i - k_{(i-1)}\right) \varepsilon_\delta^{\bar{N}}(x_{k_i+1}).$$

Hence, due to Assumption 2, we obtain

$$\hat{\theta}_{\bar{N}}(x_{k_{(i-1)}+1}) - \hat{\theta}_{\bar{N}}(x_{k_i+1}) \geq \frac{\bar{N}}{N_{max}} \kappa > 0.$$

However, such subsequence can not exist since the function $\hat{\theta}$ is bounded from below – more precisely, $\hat{\theta}_{\bar{N}}(x) \geq 0$. Therefore, once again we obtain the contradiction and conclude that the sample size can not oscillate permanently. Finally, from everything above, we conclude that the statement of this theorem is true. □

In order to prove the main result, we need an additional assumption. Notice that the following implication is obviously satisfied for the negative gradient.

Assumption 3 *The search directions d_k are descent, bounded and the implication $\lim_{k\in K} g_k^T d_k = 0 \implies \lim_{k\in K} g_k = 0$ holds for any subsequence $K \subseteq \mathbb{N}$.*

Theorem 1 *Suppose that the Assumptions 1-3 hold. Then $\lim_{k\to\infty} \mu_k = \infty$.*

Proof Due to Lemma 2 there exists $q \in \mathbb{N}$ such that $N_k = N_{max}$ for every $k \geq q$. Therefore, by Step 7 of Algorithm 1 there are two possibilities for $k \geq q$: μ_k is increased if $dm_k \leq \alpha_k/\mu_k^2$, otherwise it remains unchanged. If the increase happens infinitely many times, the result holds. Therefore, let us consider the opposite case, i.e. suppose that there exists an iteration $q_1 > q$ such that $dm_k > \alpha_k/\mu_k^2$ for every $k > q_1$. In that case $\mu_k = \mu_{q_1}$ for every $k > q_1$ and the previous inequality is equivalent to $-g_k^T d_k > \mu_{q_1}^{-2}$. Therefore, the sequence $\{-g_k^T d_k\}_{k>q_1}$ is bounded from below. On the other hand, since the sample size and the penalty parameter are fixed for every $k > q_1$, Lemma 1 implies

$$0 = \lim_{k\to\infty} dm_k = \lim_{k\to\infty} \left(-\alpha_k g_k^T d_k\right).$$

Consequently,

$$\lim_{k\to\infty} \alpha_k = 0, \tag{6}$$

which implies the existence of $q_2 > q_1$ such that for every $k > q_2$ the step size α_k is smaller than 1. This further implies that for all $k > q_2$ there exists α_k' such that $\alpha_k = \beta\alpha_k'$ and α_k' does not satisfy the inequality (5), i.e.,

$$\phi(x_k + \alpha_k' d_k; N_{max}; \mu_{q_1}) > \phi(x_k; N_{max}; \mu_{q_1}) + \eta\,\alpha_k' g_k^T d_k.$$

Defining $\phi(x_k) := \phi(x_k; N_{max}; \mu_{q_1})$, the previous inequality is equivalent to

$$\frac{\phi(x_k + \alpha_k' d_k) - \phi(x_k)}{\alpha_k'} > \eta\, d_k^T \nabla_x \phi(x_k).$$

Using the Mean value theorem we obtain that there exists $t_k \in (0,1)$ such that

$$d_k^T \nabla_x \phi(x_k + t_k \alpha_k' d_k) > \eta\, d_k^T \nabla_x \phi(x_k). \tag{7}$$

Let x^* be an arbitrary accumulation point of the sequence $\{x_k\}_{k\in\mathbb{N}}$ and denote by K the corresponding subset such that $\lim_{k\in K}(x_k, d_k) = (x^*, d^*)$. Notice here that d^* exists since the sequence of search directions is bounded. Furthermore, due to $\alpha_k = \beta\alpha_k'$ and (6) there holds $\lim_{k\to\infty} \alpha_k' = 0$. This equality, together with $t_k \in (0,1)$ for all $k > q_2$ and the boundedness of $\{d_k\}$ imply that $\lim_{k\in K}(x_k + t_k\alpha_k' d_k) = x^*$. Letting $k \in K$, $k \to \infty$ in the inequality (7), we obtain $(d^*)^T \nabla\phi(x^*) \geq \eta (d^*)^T \nabla\phi(x^*)$ or equivalently, $(d^*)^T \nabla_x \phi(x^*)(1-\eta) \geq 0$. The condition $\eta \in (0,1)$ and the previous inequality imply

$$(d^*)^T \nabla\phi(x^*) \geq 0.$$

On the other hand, the search direction is assumed to be descent, so $(d^*)^T \nabla_x \phi(x^*) \leq 0$, which further implies

$$(d^*)^T \nabla\phi(x^*) = 0.$$

This is in contradiction with

$$-g_k^T d_k > \mu_{q_1}^{-2}$$

and the statement is proved. □

Notice that Theorem 1 implies the existence of an infinite sequence $\{z_t\}$ defined by Step 4 of Algorithm 1. Finally we prove the global convergence result.

Theorem 2 *Suppose that the Assumptions 1-3 hold. Then every accumulation point x^* of $\{z_t\}_{t\in\mathbb{N}}$ is stationary for $\hat{\theta}_{N_{max}}$. Moreover, if LICQ holds then x^* is a KKT point of the problem* (2)*.*

Proof Let x^* be an arbitrary accumulation point of the sequence $\{z_t\}$. Since $\{z_t\} \subseteq \{x_k\}$, there is a set $K \subseteq \mathbb{N}$ such that $\lim_{k\in K} x_k = x^*$ and $dm_k \leq \alpha_k/\mu_k^2$, or equivalently $0 < -g_k^T d_k \leq \mu_k^{-2}$, for all $k \in K$. Due to Lemma 2 there exists $q \in \mathbb{N}$ such that $N_k = N_{max}$ for every $k \geq q$. Also, Theorem 1 implies $\lim_{k\to\infty} \mu_k = \infty$. Without loss of generality, we can assume that $k \geq q$ for all $k \in K$. Therefore, it holds that $\lim_{k\in K} g_k^T d_k = 0$ and the Assumption 3 implies $\lim_{k\in K} g_k = 0$. Since $g_k = \nabla f(x_k) + \mu_k \nabla \hat{\theta}_{N_{max}}(x_k)$ we obtain

$$\|\nabla \hat{\theta}_{N_{max}}(x_k)\| \leq (\|\nabla f(x_k)\| + \|g_k\|)/\mu_k$$

and taking the limit we get

$$\nabla \hat{\theta}_{N_{max}}(x^*) = 0,$$

i.e. x^* is a stationary point of $\hat{\theta}_{N_{max}}$.

Besides, if LICQ holds then $\nabla \hat{h}_{N_{max}}(x^*)$ has a full rank. Since

$$\nabla \hat{\theta}_{N_{max}}(x^*) = 2\nabla \hat{h}_{N_{max}}(x^*)^T \hat{h}_{N_{max}}(x^*),$$

we conclude that $\hat{h}_{N_{max}}(x^*)$ must be zero, i.e. x^* is feasible. Moreover, it is a KKT point of problem (2) with the Lagrange multiplier

$$\lambda^* = -\left(\nabla \hat{h}_{N_{max}}(x^*) \nabla \hat{h}_{N_{max}}(x^*)^T\right)^{-1} \nabla \hat{h}_{N_{max}}(x^*) \nabla f(x^*). \tag{8}$$

To see this, define $\lambda_k := 2\mu_k \hat{h}_{N_{max}}(x_k)$. Then

$$g_k = \nabla f(x_k) + \nabla \hat{h}_{N_{max}}(x_k)^T \lambda_k. \tag{9}$$

Due to the continuity of $\nabla \hat{h}_{N_{max}}$, the matrix $\nabla \hat{h}_{N_{max}}(x_k)$ has a full rank for sufficiently large k. So, for sufficiently large k the matrix $\nabla \hat{h}_{N_{max}}(x_k) \nabla \hat{h}_{N_{max}}(x_k)^T$ is nonsingular. Therefore, after multiplying (9) by $\nabla \hat{h}_{N_{max}}(x_k)$ from the left side and rearranging, we obtain

$$\lambda_k = \left(\nabla \hat{h}_{N_{max}}(x_k) \nabla \hat{h}_{N_{max}}(x_k)^T\right)^{-1} \nabla \hat{h}_{N_{max}}(x_k) \left(g_k - \nabla f(x_k)\right).$$

Now, using the fact that $\lim_{k\in K} g_k = 0$ we conclude that $\lim_{k\in K} \lambda_k = \lambda^*$ given by (8) and

$$0 = \lim_{k\in K} g_k = \lim_{k\in K} \left(\nabla f(x_k) + \nabla^T \hat{h}_{N_{max}}(x_k) \lambda_k\right) = \nabla f(x^*) + \nabla \hat{h}_{N_{max}}(x^*)^T \lambda^*.$$

□

The statement of Theorem 2 is a general convergence result as we are dealing with an arbitrary search direction d_k, assuming only that d_k is a descent direction for $\phi(x_k, N_k, \mu_k)$. Any particular search direction, like the negative gradient direction or some second-order direction, will further imply additional properties like convergence rate and similar.

4 Numerical results

The test collection consists of 14 standard optimization problems with the unique solution and the objective function bounded from below on $\mathbb{R}^n$. The problems (6, 27, 28, 42, 46-52, 61, 77 and 79) are taken from Hock and Schittkowski [9] and transformed into SAA by $H(x,\xi) = c(\xi x)$ where ξ follows Normal distribution $\mathcal{N}(1,1)$ and $c(x)$ is the function defining constraints in [9]. For each of the problems, 10 different samples of size $N_{max} = 2000$ are generated and in total 140 different problems have been tested. The tests are carried out by implementing the proposed algorithm in Matlab 8.0. The samples are generated by the built-in solver *randn*. As already mentioned, the sample $\{\xi_1,\ldots,\xi_{N_{\max}}\}$ is generated at the beginning of the iterative procedure. Whenever the sample size is smaller than $N_{\max}$, i.e., in all iterations where $N_k < N_{max}$, we take the first N_k elements of the full sample.

The proposed procedure (VSS) is compared with two other sample scheduling schemes – the SAA where $N_k = N_{max}$ for each k, and the heuristic (HEUR) where $N_{k+1} = \lceil \min\{1.1N_k, N_{max}\} \rceil$ as in [7,13,15]. To make the comparison fair, all the remaining parameters are the same for all tests. The BFGS search direction with the safeguard that ensures the descent property of d_k and the gradient difference $y_k = \nabla\phi(x_{k+1};N_{k+1};\mu_k) - \nabla\phi(x_k;N_k;\mu_k)$ is used. Line search is performed with $\beta = 0.5$ and $\eta = 10^{-4}$. The initial penalty parameter is set to $\mu_0 = 1$ and the increase factor is $\gamma = 1.5$. The initial points are as in [9] and the initial (and minimal) sample size is $N_{min} = 3$. Furthermore, Algorithm 1 is applied with $\nu_1 = 1/\sqrt{N_{max}}$ and $\varepsilon_\delta^{N_k}$ defined by (4).

The comparison is based on number of function H evaluations (FEV) where each component of ∇H is counted as one FEV (see [19] for instance). The stopping criterion is

$$\| \left(\nabla_x \phi(x_k;N_{max};\mu_k), \hat{h}_{N_{max}}(x_k)\right) \| \leq 10^{-1}.$$

Notice that the algorithms terminate (successfully) only if $N_k = N_{max}$ and a KKT point is approximated. On the other hand, if the stopping criterion is not met within 10^8 FEVs, the run is considered unsuccessful.

The results are presented using the performance profile, introduced by Dolan and Moré [5] and shown in Figure 1. Roughly speaking, for every observed solver (VSS, HEUR, SAA) the performance profile defines cumulative distribution function of the so-called performance ratio comparing to all other observed solvers on the considered test set of problems (140 examples). Performance ratio of a solver at any problem is the ratio of obtained FEV for the considered problem by the considered solver and the minimal FEV of all observed solvers on the considered problems. That is, performance profile of a solver is the probability that its FEV is not greater than the factor $\tau \in \mathbb{R}$ of the FEV of the best tested solver.

The results of comparison of algorithms are presented in Figure 1. As we can see, the winning probabilities (results for $\tau = 1$) of VSS, HEUR and SAA are 0.52, 0.29 and 0.12 respectively. Differences in probabilities when performance ratios are two times larger than the best performance ratio (the case $\tau = 2$) are roughly of the same order – probabilities for $\tau = 2$ are 0.91, 0.67 and 0.47 respectively for VSS, HEUR and SAA. Therefore, the differences in number of function of evaluations in com-

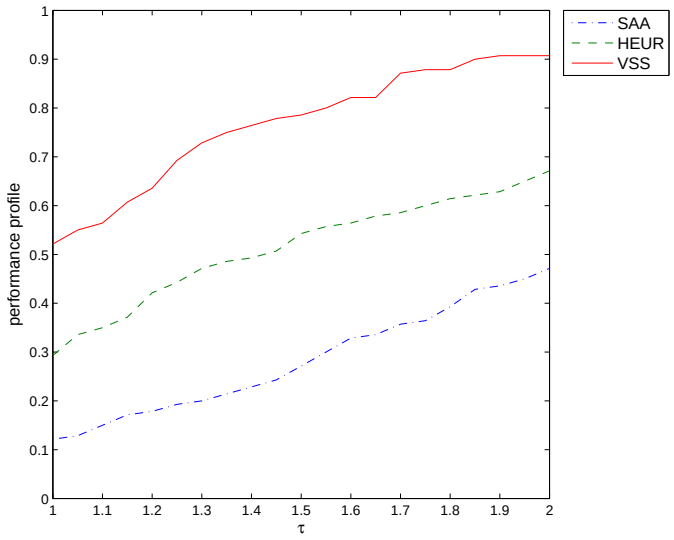

Fig. 1 Performance profile

pared methods are significant. We can conclude that strategies with varying sample size outperform SAA method significantly, at least at the considered test collection, and it is worth while to use smaller sample size at the beginning of the search and while we are far away from the solution. Furthermore, plot shows that Algorithm 1 outperforms the heuristic scheme significantly and fully justifies the idea of decreasing the sample size whenever we are far away from the solution.

We conclude this section by addressing the robustness of the methods. In the vast majority of cases, algorithms approached a KKT point. The exceptions are problems 46 and 77. In the case of 46, HEUR and VSS failed each one at one run, while the SAA was fully successful. On the other hand, in problem 77 all the tested methods failed except for VSS which managed to solve one run.

5 Conclusions

Difficulties of solving stochastic problems of the form considered in this paper are due to high cost of computing the mathematical expectation. This difficulty can be resolved by transforming the problem (1) into an SAA problem, with sufficiently large sample. However, solving the SAA problem with large sample which ensures a good approximation of the original problem, leads to computationally costly procedure with very high number of function evaluations. The algorithm proposed in this paper is such that the sample size is varying during optimization process. Under a set of standard conditions, the convergence of sequence generated by the proposed algorithm to a KKT point of the SAA problem is shown. The presented numerical results demonstrate that the algorithm requires significantly smaller number of function

evaluations than the SAA method and the heuristic procedure. The proposed method is also fairly robust.

6 Appendix

Algorithm 2
Step 0 Input parameters: dm_k, $\varepsilon_\delta^{N_k}(x_k)$, x_k, N_k, N_k^{min}, $\nu_1 \in (0,1)$.
Step 1 Determine N_{k+1}
1) If $dm_k = \varepsilon_\delta^{N_k}(x_k)$ set $N_{k+1} = N_k$.
2) If $dm_k > \varepsilon_\delta^{N_k}(x_k)$
Starting with $N = N_k$, while $dm_k > \frac{N_k}{N}\,\varepsilon_\delta^{N}(x_k)$ and $N > N_k^{min}$, decrease N by 1 and calculate $\varepsilon_\delta^{N}(x_k)$. Set $N_{k+1} = N$.
3) $dm_k < \varepsilon_\delta^{N_k}(x_k)$
i) If $dm_k \geq \nu_1 \varepsilon_\delta^{N_k}(x_k)$
Starting with $N = N_k$, while $dm_k < \frac{N_k}{N}\,\varepsilon_\delta^{N}(x_k)$ and $N < N_{max}$, increase N by 1 and calculate $\varepsilon_\delta^{N}(x_k)$. Set $N_{k+1} = N$.
ii) If $dm_k < \nu_1 \varepsilon_\delta^{N_k}(x_k)$ set $N_{k+1} = N_{max}$.

Algorithm 3 We say that we have not made big enough decrease of the function $\hat{\theta}_{N_{k+1}}$ if the following inequality is true

$$\frac{\hat{\theta}_{N_{k+1}}(x_{l(k)}) - \hat{\theta}_{N_{k+1}}(x_{k+1})}{k+1-l(k)} < \frac{N_{k+1}}{N_{max}}\,\varepsilon_\delta^{N_{k+1}}(x_{k+1}),$$

where $l(k)$ is the iteration at which we started to use the sample size N_{k+1} for the last time.

Step 0 Input parameters: N_k, N_{k+1}, N_k^{min}.
Step 1 Determine N_{k+1}^{min}:
1) If $N_{k+1} \leq N_k$ then $N_{k+1}^{min} = N_k^{min}$.
2) If $N_{k+1} > N_k$ and
i) if N_{k+1} is a sample size which has not been used so far then $N_{k+1}^{min} = N_k^{min}$.
ii) if N_{k+1} is a sample size which had been used and if we have made big enough decrease of the function $\hat{\theta}_{N_{k+1}}$ since the last time we used it, then $N_{k+1}^{min} = N_k^{min}$.
iii) if N_{k+1} is a sample size which had been used and if we have not made big enough decrease of the function $\hat{\theta}_{N_{k+1}}$ since the last time we used it, then $N_{k+1}^{min} = N_{k+1}$.

Acknowledgements. We are grateful to the Associate Editor and reviewers whose comments helped us to improve the paper.

References

1. Bastin, F.: Trust-Region Algorithms for Nonlinear Stochastic Programming and Mixed Logit Models, PhD thesis. University of Namur, Belgium (2004)
2. Bastin, F., Cirillo, C., Toint, P.L.: An adaptive Monte Carlo algorithm for computing mixed logit estimators, Comput. Manag. Sci. 3(1), 55-79 (2006)
3. Bastin, F., Cirillo, C., Toint, P.L.: Convergence theory for nonconvex stochastic programming with an application to mixed logit, Math. Program. 108(2-3), 207–234 (2006)
4. Deng, G., Ferris, M.C.: Variable-number sample path optimization, Math. Program. 117(12), 81-109 (2009)
5. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles, Math. Program. 91(2), 201-213 (2002)
6. Dolgopolik, M.V.: Smooth exact penalty functions: a general approach, Optim. Lett. 10, 635–648 (2016)
7. Friedlander, M.P., Schmidt, M.: Hybrid deterministic-stochastic methods for data fitting, SIAM. J. Sci. Comput. 34(3), 13801405 (2012)
8. Gill, P.E., Murray, W., Wright, M.H.: Practical optimization. Academic Press, London (1997)
9. Hock, W., Schittkowski, K.: Test Examples for Nonilnear Programming Codes, Lecture Notes in Economics and Mathematical Systems 187, Springer (1981)
10. Homem-de-Mello, T.: Variable-Sample Methods for Stochastic Optimization, ACM Trans. Model. Comput. Simul. 13(2), 108–133 (2003)
11. Huyer, W., Neumaier, A.: A new exact penalty function,SIAM J. Optim. 13(4), 1141–1158 (2003)
12. Krejić, N., Krklec, N.: Line search methods with variable sample size for unconstrained optimization, J. Comput. Appl. Math. 245, 213-231 (2013)
13. Krejić, N., Krklec Jerinkić, N.: Nonmonotone line search methods with variable sample size, Numer. Algorithms 68(4), 711-739 (2015)
14. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research, Springer Verlag New York Inc (1999)
15. Pasupathy, R.: On choosing parameters in retrospective-approximation algorithms for stochastic root finding and simulation optimization, Oper. Res. 58(4), 889-901 (2010)
16. Polak, E., Royset, J.O.: Eficient sample sizes in stochastic nonlinear programing, J. Comput. Appl. Math. 217(2), 301-310 (2008)
17. Shapiro, A.: Monte Carlo Sampling Methods. In: Stochastic programming, Handbook in Operations Research and Management Science 10, Elsevier, 353–425 (2003)
18. Shapiro, A., Dentcheva, D., Ruszczynski, A.: Lectures on Stochastic Programming: Modeling and Theory. MPS-SIAM Series on Optimization (2009)
19. Spall, J.C.: Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. Wiley-Interscience series in discrete mathematics, New Jersey (2003)
20. Wang, X., Ma, S., Yuan, Y.: Penalty Methods with Stochastic Approximation for Stochastic Nonlinear Programming, technical report, arXiv:1312.2690 [math.OC] (2015)