Structural and Multidisciplinary Optimization (2020) 61:2595–2611
https://doi.org/10.1007/s00158-020-02571-x

RESEARCH PAPER

CSG: A new stochastic gradient method for the efficient solution of structural optimization problems with infinitely many states

Lukas Pflug[1,2] · **Niklas Bernhardt**[1] · **Max Grieshammer**[1] · **Michael Stingl**[1]

Received: 14 September 2019 / Revised: 7 January 2020 / Accepted: 4 March 2020 / Published online: 31 May 2020

Abstract

This paper presents a novel method for the solution of a particular class of structural optimzation problems: the continuous stochastic gradient method (CSG). In the simplest case, we assume that the objective function is given as an integral of a desired property over a continuous parameter set. The application of a quadrature rule for the approximation of the integral can give rise to artificial and undesired local minima. However, the CSG method does not rely on an approximation of the integral, instead utilizing gradient approximations from previous iterations in an optimal way. Although the CSG method does not require more than the solution of one state problem (of infinitely many) per optimization iteration, it is possible to prove in a mathematically rigorous way that the function value as well as the full gradient of the objective can be approximated with arbitrary precision in the course of the optimization process. Moreover, numerical experiments for a linear elastic problem with infinitely many load cases are described. For the chosen example, the CSG method proves to be clearly superior compared to the classic stochastic gradient (SG) and the stochastic average gradient (SAG) method.

Keywords Stochastic gradient method · Infinitely many state problems · Robust structural optimization · Proof of convergence

1 Introduction and problem statement

In the following, we define the set of Lebesgue integrable functions mapping from the space X to space Y by $L^1(X;Y)$ and from the space X to the real numbers $\mathbb{R}$ by $L^1(X)$. The "dot" notation is used in the following way: $g(\cdot, y)$ denotes the mapping $x \mapsto g(x, y)$.

Our goal in this article is to develop a novel stochastic gradient method for the efficient solution of optimization problems of the general form:

$$\min_{u\in U_{ad}} F(u) = J(f(u,\cdot)). \tag{1.1}$$

Here, u is the design variable, which can be subject to constraints described by the set U_{ad}, and $F : U_{\text{ad}} \mapsto \mathbb{R}$ is given as a composition of a functional $J : L^1(V_{\text{ad}}) \mapsto \mathbb{R}$ and a function $f : U_{\text{ad}} \times V_{\text{ad}} \mapsto \mathbb{R}$, where V_{ad} is a continuous parameter set. Throughout this paper, we further assume that the evaluation of the function f for any $(u, v) \in U_{\text{ad}} \times V_{\text{ad}}$ requires the solution of an underlying state problem, i. e., f is given in the form as follows:

$$f(u, v) = j(u, y(u; v)),$$

where $y(u; v)$ denotes the solution of the state problem parameterized by the design u and the additional continuous *index variable* $v \in V_{\text{ad}}$. As a consequence of this construction, an evaluation of the function F at a given design u theoretically requires the solution of infinitely many state problems.

In order to demonstrate that problem (1.1) has a broad range of applications, we give two particular examples for the choice of the functional J. In our first example, J is simply an integral over V_{ad}, resulting in the problem as follows:

$$\min_{u\in U_{\text{ad}}} \int_{V_{\text{ad}}} f(u, v)\,\mathrm{d}v. \tag{1.2}$$

Responsible Editor: Mehmet Polat Saka

✉ Lukas Pflug
Lukas.Pflug@fau.de

[1] Chair of Applied Mathematics (Continuous Optimization), Department of Mathematics, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Cauerstr. 11, Erlangen, 91058, Germany

[2] Zentralinstitut for Scientific Computing, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Martensstr. 5a, 91058 Erlangen, Germany

The structure in (1.2) can arise in various settings. First, in an elastic setting, v could be a continuous load index and $f(\cdot, v)$ a compliance, displacement, or stress evaluation associated with the solution of the state problem with load index $v \in V_{\text{ad}}$. In this case, (1.2) would be a structural optimization problem with infinitely many load cases. For optimization problems with at least *many* load cases, we refer, e.g., Alvarez and Carrasco (2005) and Zhang et al. (2017), as well as the references therein. Furthermore, if for applications in acoustics (see, e.g., Dilgen et al. 2019) or optics (see, e.g., Jensen and Sigmund 2011), a state problem in time-harmonic form is considered, the parameter v can play the role of a frequency or wavelength. A prominent example for f in this context would be an L^2-tracking function, such that (1.2) would describe the design of a device with a prescribed behavior over a continuous frequency range, such as the range of visible light, see Semmler et al. (2015). Another application in optics could be the optimization of an anisotropic object with respect to arbitrary illumination directions, again see Semmler et al. (2015). While we have so far looked at all these examples from a deterministic point of view, another important class of applications arises if we interpret $F(u)$ as the expected value of a given property f associated with a design u. This immediately leads to the notion of reliability-based optimization problems (RBO), see, e.g., the overview article by Maute and Frangopol (2003), Conti et al. (2009) or De et al. (2020) and the references therein for a collection of more recent articles on this topic. In all these cases, the parameter v constitutes a realization of uncertainty (from a continuous uncertainty set V_{ad}), where the source of uncertainty can be, e.g., in loading, material properties, or stiffness.

Following this line of argumentation a little further, we come to the second instantiation of the generic problem (1.1), which is referred to as robust structural optimization problem according to De et al. (2020) and takes the form as follows:

$$\min_{u \in U_{\text{ad}}} \mathbb{E}[f(u, \cdot)] + \lambda \text{Var}(f(u, \cdot)). \tag{1.3}$$

Here, the expected value $\mathbb{E}[f(u, \cdot)])$ is computed by the integral in (1.2), and λ is a positive parameter that denotes the importance of variations.

Having said this, we would like emphasize out that our paper is not the first one suggesting the application of stochastic-gradient-type methods for the solution of structural optimization problems of the aforementioned type. Zhang et al. (2017) use the classical stochastic gradient (SG) method for the efficient solution of structural optimization with many (but finitely many) load cases. In De et al. (2020), robust optimization problems in the form of (1.3) are solved by the SG method and variants, as well as a stochastic version of the well-known GCMMA framework, see Svanberg (2002). However, in this paper, a discrete set of scenarios is also assumed from the very beginning. More applications of the SG method can be found in the closely related area of inverse problems, see Haber et al. (2012) and Roosta-Khorasani et al. (2014). These are structurally similar to the problems considered in this article, in the sense that each evaluation of the function f for a given scenario v, and also requires the potentially expensive solution of a discretized partial differential equation (PDE).

However, there are at least two substantial differences between all these references and the approach we suggest in this paper. Firstly, even though in some cases in the aforementioned articles a continuous index set V_{ad} is considered, an a priori selection of scenarios (or discretization of V_{ad}) is used. Secondly, even though in many applications the property function f depends on the index variable v with a certain regularity, this structure is ignored. In sharp contrast to this, we avoid an a priori discretization of integrals in this paper. This is principally due to the fact that a too coarse discretization can lead to artificial minima as will be demonstrated by means of a simple example in Section 2. Moreover, we would like to exploit the natural regularity of the property function f with respect to the index parameter v in order to design an efficient optimization algorithm in which the objective function F and its gradient are increasingly better approximated.

Beyond stochastic descent methods, robust problems of type (1.3) can also be approached by a combination of stochastic collocation methods combined with deterministic optimization solvers, see, e.g., Lazarov et al. (2012). However, also in this case, through the collocation, an approximation of the objective functional based on finitely many scenarios is chosen a priori.

To the best knowledge of the authors, the SG method itself is the only method which can, in principle, be applied to structural optimization problems with infinitely many state problems without relying on an a priori approximation of the objective functional. However, as it will be shown in the article, only the CSG method is able to successfully solve these problems.

Despite this, there are substantial structural similarities between our CSG method and the classic SG method and its relatives. Therefore, in the following, we briefly describe the basic SG idea.

The original SG method, see Robbins and Monro (1951), is a method frequently used to minimize functions of the form $F : \mathbb{R}^p \to \mathbb{R}$, with

$$F(x) := \frac{1}{n} \sum_{i=1}^{n} f_i(x)$$

and $f_i := \mathbb{R}^p \mapsto \mathbb{R}$ for all $i \in \{1, \dots, n\}$. Whereas conventional gradient methods calculate all n gradients ∇f_i in each iteration, the stochastic gradient method uses only a small random selection thereof. Hereby, for large n (as, e.g., in machine learning applications, see Bottou and Cun 2004), the computational effort can be drastically reduced in comparison to the classical gradient method. Bottou et al. (2018) and Schmidt et al. (2017) introduced an improved version of the SG algorithm, the so-called stochastic average gradient method (SAG). This benefits from previously calculated gradients, leading to a better approximation of the exact gradient. Thus, a better convergence behavior is typically observed. In this paper, the basic properties of the SG and the SAG method will be combined.

- Low computational effort per iteration
- Reuse of previously obtained information

We will integrate these two properties in our CSG algorithm and compare it with the SG and the SAG method by means of examples.

The remainder of this paper is structured as follows: We will first close this section with the formal statement of the problem. In Section 2, we introduce the proposed CSG algorithm by which we aim to solve the types of problems discussed. In Section 3, we then analytically study the convergence of the algorithm and state assumptions necessary for convergence. The main theoretical results are given in the two Theorems 19 and 20. Section 4 provides first numerical results of the CSG algorithm, as well as a comparison with the SG and SAG methods. Finally, in Section 5, we provide a summary and a brief outlook for further scientific work.

1.1 Formal statement of the problem

Generally, we look at the objective functionals as defined in (1.1), and we further assume the following:

Assumption 1 (**Objective functional**) *The reduced objective functional $F(u) : U_{ad} \mapsto \mathbb{R}$ is given by the composition of a mapping $J : L^1(V_{ad}; \mathbb{R}) \mapsto \mathbb{R}$ and $f : U_{ad} \times V_{ad} \mapsto \mathbb{R}$. The Fréchet derivative of J will be denoted by its associated function $D_J := L^\infty(V_{ad}; \mathbb{R}) \times V_{ad} \mapsto \mathbb{R}$. D_J and $\nabla_u f$ have to be Lipschitz continuous w.r.t. both their arguments in the respective topology.*

For a definition of the Fréchet differential, see for instance (Jahn 2007, Def. 3.11.). With F chosen as in (1.1) and the latter assumption, we can state its derivative as follows:

Remark 2 (**Derivative of** F) The derivative of F is then given by the following:

$$\nabla F(u) = \int_{V_{\text{ad}}} D_J(f(u, \cdot), v) \cdot \nabla_u f(u, v) \, \mathrm{d}v \tag{1.4}$$

for all $u \in U_{\text{ad}}$, with D_J being the Fréchet differential as defined in Assumption 1.

The Fréchet derivative mentioned is exemplified in two relevant cases:

Remark 3 (**Examples for Fréchet derivatives**)
If $J(f(u, \cdot))$ is the expected value of f w.r.t. the second argument, we obtain the following:

$$D_J(f(u, \cdot), v) = 1$$

and for $J(f(u, \cdot))$ being the variance of f with respect to the second argument, we obtain for all $v \in V_{\text{ad}}$, as follows:

$$D_J(f(u, \cdot), v) = 2(f(u, v) - \mathbb{E}[f(u, \cdot)]) + \int_{V_{\text{ad}}} 2(f(u, w) - \mathbb{E}[f(u, \cdot)]) \, \mathrm{d}w.$$

As already mentioned, in the case of structural optimization, for each parameter $v \in V_{\text{ad}}$, the evaluation of $f(\cdot, v)$ requires the solution of an associated state problem. Consequently, the (approximate) evaluation of F and its gradient given in (1.4) is computationally very expensive. In general, it can be stated that the algorithm introduced in Section 2 is especially attractive for problems in which F is numerically expensive to evaluate due to its functional dependency.

One advantage of our algorithm in comparison to the SG and the SAG method is that no (a priori) quadrature rule is used to approximate the integral in (1.4). In this way, we later gain convergence of a subsequence to a stationary point of the continuous problem (1.1). Moreover, this helps to avoid artificial local minima, as will be outlined in the next section.

2 Continuous stochastic gradient method

Before presenting the new optimization algorithm, we will briefly discuss how discretization of the objective functional can introduce local minima, looking at the following function as follows:

$$F(u) := \int_{-1}^{1} \frac{v^2}{(u - v)^2 + 10^{-3}} \, \mathrm{d}v. \tag{2.1}$$

By choosing $U_{\text{ad}} = [-\frac{1}{2}, \frac{1}{2}]$ and $V_{\text{ad}} = (-1, 1)$, this corresponds to problem (1.1).

In Fig. 1, the graph of the function F in (2.1) is shown along with numerical approximations based on equidistant discretizations of the integral. Although the original function is convex, local minima are introduced due to the discretization of the integral. It is noted that without much information on the function f, it is hard to choose a suitable discretization, which would avoid this effect. Nevertheless, a good algorithm should be able to prevent convergence to such an artificial local minimum. In fact, this is one of the key features of the CSG algorithm introduced in detail in the following section.

2.1 The CSG algorithm

With $\mathcal{P}_{U_{\text{ad}}}$ being the orthogonal projection onto U_{ad}, λ^{d_v} being the Lebesgue measure in the d_v-dimensional space, and

$$M_{i,j}^n := \left\{ v \in V_{\text{ad}} \,:\, \left\| \begin{pmatrix} u_n - u_i \\ v - \omega_i \end{pmatrix} \right\|_* < \left\| \begin{pmatrix} u_n - u_j \\ v - \omega_j \end{pmatrix} \right\|_* \right\}$$

being the set of points that are closer to (u_i, ω_i) than they are to (u_j, ω_j) in the $\|\cdot\|_*$-norm given in Definition 10, we can state the proposed Algorithm 1:

Algorithm 1 CSG: **C**ontinuous **S**tochastic **G**radient method.

u_0 given
for $n = 0, 1, \ldots$ **do**
  choose ω_n uniformly at random in V_{ad}
  $g_n := \nabla_u f(u_n, \omega_n)$
  $f_n := f(u_n, \omega_n)$
  for $i = 0, \ldots, n$ **do**
    $V_i^n := \bigcap_{k \neq i} M_{i,k}^n$
    $\hat{f}_n(u_n, w) := f_i$ iff $w \in V_i^n$
    $(D_J^h)_i^n := D_J(\hat{f}_n(u_n, \cdot), \omega_i)$
    $\alpha_i^n := \lambda^{d_v}(V_i^n)$
  end
  $\hat{G}_n := \sum_{i=0}^n \alpha_i^n (D_J^h)_i^n g_i$ (search direction)
  $\hat{F}_n := J(\hat{f}_n(u_n, \cdot))$
  choose suitable stepsize τ_n
  $u_{n+1} := \mathcal{P}_{U_{ad}}\left(u_n - \tau_n \hat{G}_n\right)$
  if *termination conditons satisfied* **then**
    break
  end
end

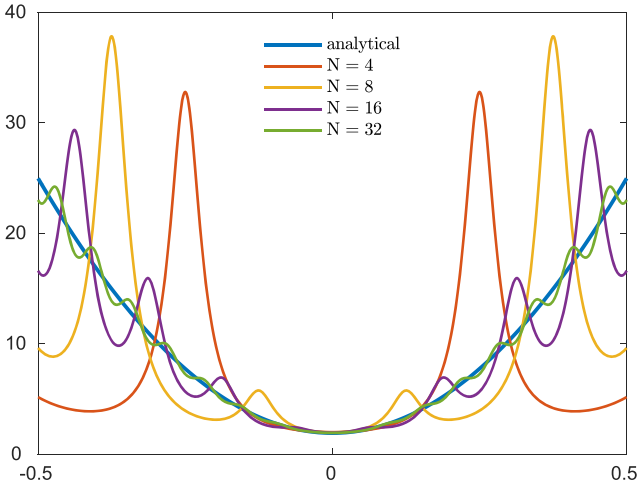

Fig. 1 The analytic function F (blue) and the function F discretized with 4 (red), 8 (yellow), 16 (green), and 32 (purple) equidistant discretization points. F is given in (2.1)

Note that $(V_i^n)_{i=1}^n$ defined in Algorithm 1 is a partition of V_{ad} for all $n \in \mathbb{N}$, i.e., $V_i^n \cap V_j^n = \emptyset$ for all $i, j \in \{0, \ldots, n\}$ with $i \neq j$ and $V_{\text{ad}} = \cup_{i=0}^n V_i^n$.

The CSG method as defined in Algorithm 1 is suitable for problems of the form (1.1) and is structured as most gradient descent methods. In each iteration n, a search direction $\hat{G}_n$ (an approximation of the gradient $\nabla F_n := \nabla F(u_n)$) is calculated, a step length $\tau_n > 0$ is chosen and a sequence $(u_n)_{n\in\mathbb{N}}$ is generated by the following:

$$u_{n+1} := \mathcal{P}_{U_{\text{ad}}}\left(u_n - \tau_n \hat{G}_n\right).$$

Note that the existence and uniqueness of $\mathcal{P}_{U_{\text{ad}}}$ is guaranteed by the projection theorem (see, e.g., Aubin (2000)) and for all $w \in \mathbb{R}^{d_u}$ defined by the following:

$$\mathcal{P}_{U_{\text{ad}}}(w) := \operatorname*{arg\,min}_{u \in U_{\text{ad}}} \|u - w\|.$$

In this contribution, we use the following abbreviations: $F_n := F(u_n)$ and $\|\cdot\|$ denote the euclidean norm in the respective dimensions. The distinctive feature of the algorithm lies in the calculation of the search direction $\hat{G}_n$. In each iteration n, the gradient $\nabla_u f(u_n, \cdot)$ is evaluated at a random position $\omega_n \in V_{\text{ad}}$ and stored for later iterations. The search direction $\hat{G}_n$ is in principle a linear combination of the former gradients $g_i := \nabla_u f(u_i, \omega_i)$, $i = 0, \ldots, n$ with weights α_i^n, $i = 0, \ldots, n$. To provide an idea how the weights are calculated, we refer to the sketch in Fig. 2. There, $\omega_0, \ldots, \omega_{10} \in V_{\text{ad}}$ are randomly sampled points and $g_0, \ldots, g_{10}$ the corresponding gradients. Then, the approximate gradient is given as $\hat{G}_{10} = \sum_{k=0}^{10} a_k g_k$, where $a_0^{10}, \ldots, a_{10}^{10}$ are the lengths of the line segments associated with the points $(\omega_0, u_0), \ldots, (\omega_{10}, u_{10})$. Here, the assignment of segments to points is indicated by the same color. The underlying structure is the Voronoi diagram

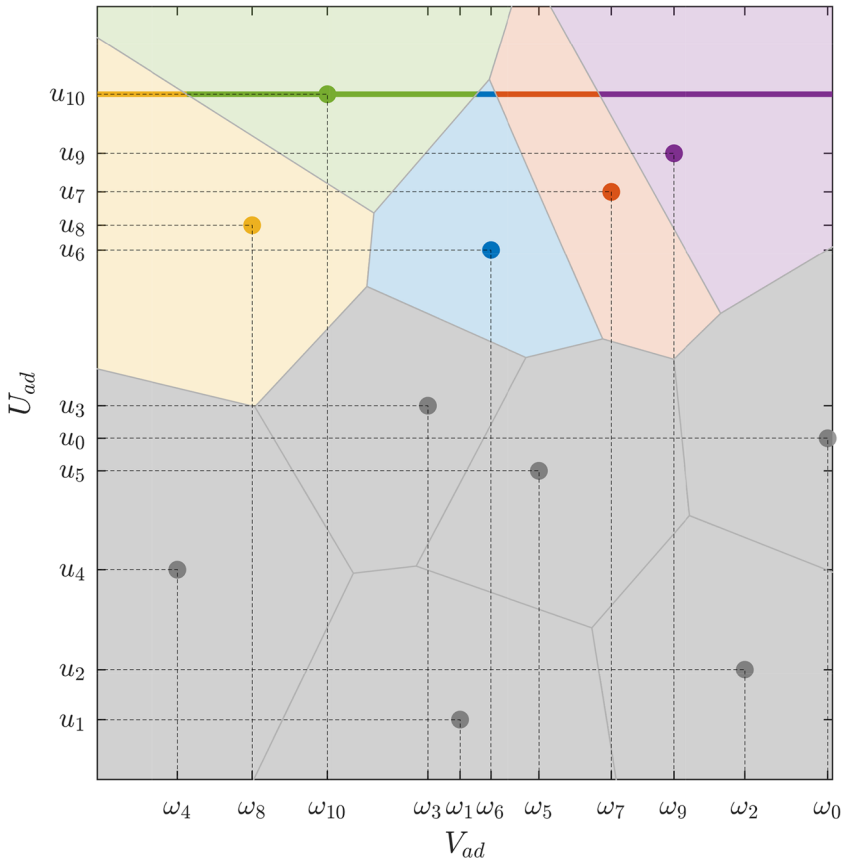

Fig. 2 Example for the calculation of the approximated gradient $\hat{G}_n$ in Algorithm 1 in the case of $V_{\text{ad}}, U_{\text{ad}} \subset \mathbb{R}$

of the points $(\omega_k, u_k)_{k\in\{1,\dots,10\}}$ (see, e.g., Fortune (1995)). More formally, the weights α_k^{10} can be defined for all $k \in \{0, \dots, 10\}$ as the d_v-dimensional measure of $\Omega_k \cap (u_{10} \times V_{\text{ad}})$ where $\Omega_k \subset V_{\text{ad}} \times U_{\text{ad}}$ is the Voronoi face associated with the point (ω_k, u_k).

We note that the computational complexity per iteration is given by the evaluation of the gradient of the function f and the calculation of the weights $\alpha_0^n, \dots, \alpha_n^n$. Up to the calculation of the weights, this is analogous to the SG and SAG method. It should also be noted that all the gradients $g_0, \dots, g_{n-1}$ of the previous iterations are included in the current iteration. In Section 3, we will show that the error $\|\hat{G}_n - \nabla F_n\|$ almost sure converges to zero. Hence, the algorithm does not become trapped in a local minimum of the discretized function. Therefore, the problem described in Fig. 1 will not arise, since the approximation of ∇F_n by $\hat{G}_n$ will be better and better.

3 Convergence analysis

In this section, we will study the convergence of the proposed algorithm. Due to the randomly chosen evaluation point within the algorithm, we will have to study probabilistic convergence behavior in terms of "almost sure convergence" and convergence in expectation. This notion of convergence as well as further assumptions and definitions are given in Section 3.1. In Section 3.2, we prove that the error in the gradient approximation goes to zero and finally apply this result in Section 3.3 to prove convergence of the CSG method.

3.1 Assumptions, definitions, and preliminary results

For the convergence analysis of Algorithm 1, the following three assumptions on the objective functional, the step length, and the sets U_{ad}, V_{ad} are an important ingredient. In the following, we will assume that these Assumptions are always satisfied without mentioning it explicitly.

Definition 4 (Lipschitz constants and maxima) We will denote the Lipschitz constants of D_J and $\nabla_u f$ with respect to both their arguments by $L^{(1)}_{D_J}, L^{(2)}_{D_J}$ and by $L^{(1)}_{\nabla_{uf}}, L^{(2)}_{\nabla_{uf}}$. Their maximal absolute function values will be defined as $C_{D_J}, C_{\nabla_{uf}}$.

For U_{ad} and V_{ad}, we assume the following:

Assumption 5 (Regularity of U_{ad} and V_{ad}) *The set $U_{ad} \subset \mathbb{R}^{d_u}$ is compact and convex. $V_{ad} \subset \mathbb{R}^{d_v}$ is*

open and bounded. In addition, there exists $c \in \mathbb{R}$ *s.t.* $\left|V_{ad} \setminus V_{ad}^r\right| \leq cr \ \forall r \in (0,1)$, *with* $V_{ad}^r := \{x \in V_{ad} : B_r(x) \subset V_{ad}\}$ *and where* $B_r(x) \subset \mathbb{R}^{d_v}$ *is an open ball centered in* $x \in \mathbb{R}^{d_v}$ *with radius* r.

The latter assumption is fulfilled for non pathological open sets with finite perimeter.

Assumption 6 (**Step length**) *The step length* $(\tau_n)_{n\in\mathbb{N}}$ *satisfies the following:* $\exists N \in \mathbb{N}$, $c_1, c_2 \in \mathbb{R}_{>0}$, *and* $\delta \in \left(0, \frac{1}{\max\{d_v,2\}}\right)$ *s.t.*

$$c_1 n^{-1} \leq \tau_n \leq c_2 n^{-1+\frac{1}{\max\{d_v,2\}}-\delta} \quad \forall n \in \mathbb{N}_{>N}.$$

These conditions on the step length satisfy the conditions stated in Robbins and Monro (1951, Eqns. (6) and (26)) as well as equivalently in Bottou et al. (2018, Eqn. (4.19) in the one-dimensional case, and can be seen as a higher dimensional equivalent.

Remark 7 (**Step length for** $d_v = 1$ **and** $d_v = 2$) In case of a one- or two-dimensional set V_{ad}, Assumption 6 is satisfied iff

$$\tau_n \in \mathcal{O}(n^{-1}) \cap o(n^{-\frac{1}{2}})$$

with the Big Oh and Little Oh notation as defined in Bürgisser and Cucker (2013). In other words, the null series $(\tau_n)_{n\in\mathbb{N}}$ must not tend faster to zero than $(n^{-1})_{n\in\mathbb{N}}$ but not as slow as $(n^{-\frac{1}{2}})_{n\in\mathbb{N}}$.

The lower bound for the stepsizes ensure that the accumulated stepsizes reach infinity and the algorithm does not get stuck due to their reduction. The upper bound ensures that the approximation of the search direction is appropriate. This is equivalent to the rate of convergence for empirical measures, see, e.g., Dudley (1969, Prop. 3.4.).

Despite these assumptions in the stepsize, in Theorem 20, a result will be stated for the case of a step length bounded away from zero.

To show convergence of the algorithm, we must first state the probability space setting.

Definition 8 (**Probability space setup**) The probability space $(\Omega, \mathcal{A}, \mathbb{P})$ is given by the following:

$$\Omega := V_{\text{ad}}^{\mathbb{N}}, \quad \mathbb{P} := \mu^{\otimes\mathbb{N}}$$
$$\mathcal{A} := \sigma\{A_1 \times \ldots \times A_n : A_i \in \mathcal{B}(V_{\text{ad}}), \forall i, n \in \mathbb{N}\},$$

where $\mu^{\otimes\mathbb{N}}(A_1 \times \ldots \times A_n) = \prod_{i=1}^n \frac{\mu(A_i)}{\mu(V_{\text{ad}})}$ is the product measure and $\mu = \lambda^{d_v}$ the Lebesgue measure in $\mathbb{R}^{d_v}$.

All the following random variables are defined in this setting. For the convergence of random variables, we use the following commonly used notation:

Definition 9 (**Stochastic convergence**) A sequence of random variables $(Z_n)_{n\in\mathbb{N}}$ converges *almost surely* to some random variable Z iff

$$\mathbb{P}\left(\{\omega \in \Omega : \lim_{n\to\infty} Z_n(\omega) = Z(\omega)\}\right) = 1,$$

which we denote by $Z_n \longrightarrow a.s.Z$.

In this document, we define the following norm on the product space $U_{\text{ad}} \times V_{\text{ad}}$.

Definition 10 (**Norm in** $U_{\text{ad}} \times V_{\text{ad}}$) For better readability, we define the following ℓ^1/ℓ^2-norm on the product space $U_{\text{ad}} \times V_{\text{ad}}$:

$$\left\|\begin{pmatrix} u \\ v \end{pmatrix}\right\|_* := \|u\|_2 + \|v\|_2.$$

Due to norm equivalence in finite dimensional spaces, the results presented later hold for all chosen norms in U_{ad} and V_{ad} and combinations thereof.

The orthogonal projection used in Algorithm 1 has some important properties:

Lemma 11 (**Orthogonal projection**) *Let* $S \subset \mathbb{R}^n$ *for* $n \in \mathbb{N}_{>0}$ *be closed and convex and let* $\mathcal{P}_S$ *be the orthogonal projection, then the following holds for all* $x, y \in \mathbb{R}^n$ *and* $z \in S$:

a) $(\mathcal{P}_S(x) - x)^T(\mathcal{P}_S(x) - z) \leq 0$,
b) $(\mathcal{P}_S(y) - \mathcal{P}_S(x))^T(y - x) \geq \|\mathcal{P}_S(y) - \mathcal{P}_S(x)\|^2 \geq 0$,
c) $\|\mathcal{P}_S(y) - \mathcal{P}_S(x))\| \leq \|y - x\|$.

Proof (a) is (ii) in Aubin (2000, Thm. 1.4.1) and (b), and (c) are (iii) and (ii) in Aubin (2000, Prop. 1.4.1). □

For $h \in C^1(U_{\text{ad}})$ and U_{ad} convex, the following sufficient conditions for first-order optimality are equivalent:

Corollary 12 (**Optimality conditions**) *For all* $u^* \in U_{ad}$, *the following items are equivalent:*

i) $-\nabla h(u^*)^T(u - u^*) \leq 0 \quad \forall u \in U_{ad}$,
ii) $\mathcal{P}(u^* - t\nabla h(u^*)) = u^* \quad \forall t \geq 0$.

We call u^* *satisfying one of the above conditions a stationary point.*

Proof Define for $u \in U_{\text{ad}}$ the cone

$$N_{U_{\text{ad}}}(u) := \{x \in \mathbb{R}^{d_u} : x^T(\bar{u} - u) \leq 0 \ \forall \bar{u} \in U_{\text{ad}}\}.$$

Using Lemma 11 ((i) for "⇒" and (ii) for "⇐"), it is straightforward to see that for $x \in \mathbb{R}^{d_u}$ and $u \in U_{\text{ad}}$,

$$\mathcal{P}_{U_{\text{ad}}}(x) = u \quad \Leftrightarrow \quad (x - u) \in N_{U_{\text{ad}}}(u).$$

Since $-\nabla h \in N_{U_{\text{ad}}}(u^*)$, the result follows. □

3.2 Error in the approximate gradient

In this subsection, we analyze the error in the nth iteration of the approximate gradient $\hat{G}_n$ and the gradient of the objective functional ∇F_n. To do this, we define for $v \in V_{\text{ad}}, \omega \in \Omega$ the sequence of random variables $(X_n)_{n\in\mathbb{N}}$ by the following:

$$X_n(\omega; v) := \min_{k=1,\dots,n} \left\| \begin{pmatrix} u_k(\omega) - u_n(\omega) \\ \omega_k - v \end{pmatrix} \right\|_*.$$

Lemma 13 (**Convergence result**) *For $v \in V_{ad}$,*

$$\sum_{n=1}^{\infty} \mathbb{P}\left(X_n(\cdot; v) > \varepsilon_n\right) < \infty,$$

where $\varepsilon_n := 2C_{\nabla_{uf}} c_2 |V_{ad}| n^{-\frac{\delta}{2}} + \tilde{\varepsilon}_n$ and $\tilde{\varepsilon}_n := n^{\frac{\delta}{2} - \frac{1}{\max(2,d_v)}}$ with c_2 and δ defined in Assumption 6 and $C_{\nabla_{uf}}$ in Definition 4. Moreover,

$$\sum_{n=1}^{\infty} \sup_{v \in V_{ad}^{\varepsilon_n}} \mathbb{P}\left(X_n(\cdot; v) > \varepsilon_n\right) < \infty,$$

with $V_{ad}^{\varepsilon_n}$ defined in Assumption 5.

Proof By item (iii) in Lemma 11 we have

$$\begin{aligned}
&\mathbb{P}(X_n(\cdot; v) \geq \varepsilon_n) \\
&\leq \mathbb{P}\left(\min_{k=i_0,\dots,n} \left\| \begin{pmatrix} u_k(\omega) - u_n(\omega) \\ \omega_k - v \end{pmatrix} \right\|_* \geq \varepsilon_n \right) \\
&\leq \mathbb{P}\left(\sum_{i=i_0}^{n-1} \|\tau_i \hat{G}_i\| + \min_{k=i_0,\dots,n} \|\omega_k - v\| \geq \varepsilon_n \right) \\
&\leq \mathbb{P}\left(C_{\nabla_{uf}} c_2 |V_{ad}| \sum_{i=i_0}^{n-1} \frac{1}{i^\kappa} + \min_{k=i_0,\dots,n} \|\omega_k - v\| \geq \varepsilon_n \right)
\end{aligned}$$

where $i_0 := \lceil n - a_n + 1 \rceil$ and $\kappa \in (0,1)$ given by $\kappa := 1 - \frac{1}{\max\{d_v,2\}} + \delta$. For $d_v = 1$, we choose $a_n = \sqrt{n}$ and if $d_v \geq 2$, we choose $a_n = n^{1-\frac{\delta}{2}}$. Observe that for $n > 2$ we obtain

$$\begin{aligned}
&\sum_{i=i_0}^{n-1} \frac{1}{i^\kappa} \leq \int_{i_0-1}^{n} \frac{1}{s^\kappa}\, \mathrm{d}s = \frac{1}{1-\kappa} \cdot \left(n^{1-\kappa} - (\lceil n - a_n \rceil)^{1-\kappa} \right) \\
&\leq \frac{1}{1-\kappa} \cdot \left(n^{1-\kappa} - (n-a_n)^{1-\kappa} \right) = \frac{1}{1-\kappa} \cdot \left(\frac{n}{n^\kappa} - \frac{n-a_n}{(n-a_n)^\kappa} \right) \\
&= \frac{1}{1-\kappa} \cdot \left(\frac{n^\kappa (1 - \frac{a_n}{n})^\kappa - n^\kappa}{n^\kappa \cdot (n-a_n)^\kappa} \right) + \frac{a_n}{(1-\kappa)(n-a_n)^\kappa}
\end{aligned}$$

applying Bernoulli's inequality in the first term

$$\leq \frac{1}{1-\kappa} \cdot \left(\frac{-\kappa a_n}{(n-a_n)^\kappa} \right) + \frac{a_n}{(1-\kappa)(n-a_n)^\kappa} = \frac{a_n}{n^\kappa} (1 - \tfrac{a_n}{n})^{-\kappa}$$

As $\frac{a_n}{n} = n^{\frac{\delta}{2} - \frac{1}{\max\{d_v,2\}}} \leq n^{-\frac{1}{2\max\{d_n,2\}}}$ we obtain

$$\sum_{i=i_0}^{n-1} \|\tau_i \hat{G}_i\| \leq \frac{C_{\nabla_{uf}} c_2 |V_{ad}|}{1 - 2^{-\frac{1}{2\max\{d_n,2\}}}} n^{-\frac{\delta}{2}} = \varepsilon_n. \tag{3.1}$$

For all $v \in V_{ad}$ there exists n large enough such that $B_{\tilde{\varepsilon}_n}(v) \subset V_{ad}$. Hence,

$$\begin{aligned}
&\mathbb{P}(X_n(\cdot; v) \geq \varepsilon_n) \leq \mathbb{P}\left(\min_{k=i_0,\dots,n-1} \|\omega_k - v\| \geq \tilde{\varepsilon}_n \right) \\
&= \mathbb{P}\left(\|\omega_k - v\| \geq \tilde{\varepsilon}_n \quad \forall k \in \{i_0, \dots, n-1\} \right) \\
&= \prod_{k=i_0}^{n-1} \mathbb{P}\left(\|\omega_k - v\| \geq \tilde{\varepsilon}_n \right) = \prod_{k=i_0}^{n-1} \left(1 - \frac{|B_{\tilde{\varepsilon}_n}(v)|}{|V_{ad}|} \right) \\
&\leq \left(1 - \frac{|B_{\tilde{\varepsilon}_n}(v)|}{|V_{ad}|} \right)^{a_n} = \left(1 - \nu \cdot n^{-\frac{d_v}{\max(2,d_v)} + \frac{d_v \delta}{2}} \right)^{a_n},
\end{aligned}$$

with $\nu := \frac{\pi^{\frac{d_v}{2}}}{\Gamma(\frac{d_v}{2}+1)|V_{ad}|}$. Thus

$$\mathbb{P}(X_n(\cdot; v) \geq \varepsilon_n) \leq \left(1 - \nu \cdot \frac{n^{\frac{\delta}{2}}}{a_n} \right)^{a_n}.$$

By the limit comparison test, the corresponding series converge. Finally, note that Assumption 5 gives that $v \in V_{ad}^{\varepsilon_n}$ implies $B_{\varepsilon_n}(v) \subset V_{ad}$ and therefore $\sup_{v \in V_{ad}^{\varepsilon_n}} \mathbb{P}(X_n(\cdot; v) \geq \varepsilon_n) \leq \left(1 - \nu \cdot \frac{n^{\frac{\delta}{2}}}{a_n} \right)^{a_n}$.

□

As a direct consequence of the latter result, we obtain almost sure convergence:

Corollary 14 (**Density of ω in V_{ad}**) *For all $v \in V_{ad}$*

$$X_n(\cdot; v) \longrightarrow a.s.0 \quad \text{for} \quad n \to \infty.$$

Proof The result follows by Lemma 1 and the Borel-Cantelli Lemma, see, e.g., Klenke (2013, Thm. 6.12). □

Thus, due to the Lipschitz continuity of $\nabla_u f$, and D_J the integral in $\nabla F(u_n)$ is increasingly is increasingly better approximated by $\hat{G}_n$ for $n \to \infty$:

Corollary 15 (**Error in gradient approximation**) *The norm of the difference between approximate gradient $\hat{G}_n$ in the nth iteration (defined in Algorithm 1) and the gradient of the exact objective functional ∇F in u_n goes to zero, i.e.,*

$$\|\hat{G}_n - \nabla F_n\| \longrightarrow a.s.0, \quad \lim_{n\to\infty} \mathbb{E}\left[\|\hat{G}_n - \nabla F_n\| \right] = 0$$

and

$$\sum_{n=0}^{\infty} \tau_n \mathbb{E}\left[\|\hat{G}_n - \nabla F_n\| \right] < \infty. \tag{3.2}$$

Proof For $v \in V_{\text{ad}}$, define

$$k^n(v) := \operatorname*{arg\,min}_{k=1,\dots,n} \left\{ \left\| \begin{pmatrix} u_k(\omega) - u_n(\omega) \\ \omega_k - v \end{pmatrix} \right\|_* \right\}.$$

Then,

$$\hat{G}_n = \int_{V_{\text{ad}}} D_J(\hat{f}(u_n, \cdot), \omega_{k^n(v)}) \nabla_u \hat{f}(u_n, v)\, \mathrm{d}v,$$

where $\hat{f}(u_n, v) := f(u_{k^n(v)}, \omega_{k^n(v)})$. By the Lipschitz continuity assumed in Assumption 1, we therefore obtain the following:

$$\begin{aligned} \|\hat{G}_n - F_n\| \le \Big(& C_{\nabla_u f} L_{D_J}^{(1)} \max\{L_f^{(1)}, L_f^{(2)}\} + C_{\nabla_u f} L_{D_J}^{(2)} \\ & + C_{D_J} \max\{L_{\nabla_u f}^{(1)}, L_{\nabla_u f}^{(2)}\} \Big) \int_{V_{\text{ad}}} X_n(\omega; v)\, \mathrm{d}v, \end{aligned}$$

with constants defined in Definition 4 and $X_n(\cdot; v)$ as defined in Lemma 1. Recall that U_{ad}, V_{ad} are bounded. Now, the almost sure convergence, as well as the convergence of the expectations, is followed by Lebesgue's dominated convergence result.

Finally, let C be a generic constant and $\varepsilon > 0$. Since $\sup_{v \in V_{\text{ad}}} X_n(\cdot; v) \le D < \infty$, where $D := \operatorname{diam}(V_{\text{ad}}) + \operatorname{diam}(U_{\text{ad}})$ denotes the diameter of V_{ad} plus the diameter of U_{ad}, and by Fubini's theorem, we have the following:

$$\begin{aligned} & \mathbb{E}\left[\|\hat{G}_n - \nabla F_n\| \right] \le C\, \mathbb{E}\left[\int_{V_{\text{ad}}} X_n(\cdot; v) \mathrm{d}v \right] \\ \le\ & C \int_{V_{\text{ad}}} \varepsilon \mathbb{P}(X_n(\cdot; v) \le \varepsilon) + 2D\, \mathbb{P}(X_n(\cdot; v) > \varepsilon)\, \mathrm{d}v \\ \le\ & C \left(\varepsilon + \int_{V_{\text{ad}} \setminus V_{\text{ad}}^{\varepsilon}} \mathbb{P}(X_n(\cdot; v) > \varepsilon)\, \mathrm{d}v + \int_{V_{\text{ad}}^{\varepsilon}} \mathbb{P}(X_n(\cdot; v) > \varepsilon)\, \mathrm{d}v \right) \\ \le\ & C \left(2\varepsilon + \sup_{v \in V_{\text{ad}}^{\varepsilon}} \mathbb{P}(X_n(\cdot; v) > \varepsilon) \right), \end{aligned}$$

where V_{ad}^r is given in Assumption 5. If we choose $\varepsilon = \varepsilon_n = 2C_{\nabla_u f} c_2 \cdot n^{-\frac{1}{2}} + n^{-\frac{1}{\max(2, d_v)} + \frac{\delta}{2}}$ as in Lemma 1, we obtain the following:

$$\begin{aligned} & \sum_{n=1}^{\infty} \tau_n \mathbb{E}\left[\|\hat{G}_n - \nabla F_n\| \right] \\ & \le C \left(\sum_{n=1}^{\infty} \frac{1}{n^{1+\delta}} + \frac{1}{n^{1+\frac{\delta}{2}}} + \sup_{v \in V_{\text{ad}}^{\varepsilon_n}} \mathbb{P}(X_n(\cdot; v) > \varepsilon_n) \right) < \infty, \end{aligned}$$

which concludes the proof. □

3.3 Convergence result

As we have seen in Corollary 14, the error $\|\hat{G}_n - \nabla F_n\|$ converges almost surely and in expectation to zero for $n \to \infty$. It remains to provide sufficient conditions under which the algorithm converges to a stationary point.

Lemma 16 (Objective functional values) *The difference of the objective functional values in iteration $n \in \mathbb{N}$ can be approximated as follows:*

$$F_{n+1} - F_n \le -\frac{1}{\tau_n} \|u_{n+1} - u_n\|^2 + \phi_n,$$

with $\phi_n := \tau_n \|\nabla F_n - \hat{G}_n\| \cdot \|\hat{G}_n\| + \tau_n^2 C \|\hat{G}_n\|^2$.

with a constant $C \in \mathbb{R}_{\ge 0}$ depending on the lipschitz constants and suprema of the involved functions.

Proof By the mean value theorem, there is a $c \in (0, 1)$ such that (we set $\nabla F_n^c := \nabla F((1-c)u_n + c u_{n+1})$

$$\begin{aligned} F_{n+1} - F_n &= (\nabla F_n^c)^T (u_{n+1} - u_n) \\ &= \nabla F_n^T (u_{n+1} - u_n) + (\nabla F_n^c - \nabla F_n)^T (u_{n+1} - u_n) \\ &\le \nabla F_n^T (u_{n+1} - u_n) + C \|u_{n+1} - u_n\|^2, \end{aligned}$$

using the Cauchy–Schwartz inequality and Definition 4. Recall that $u_{n+1} = P_{U_{\text{ad}}}(u_n - \tau_n \hat{G}_n)$. With Lemma 11 (b), (c), and the Cauchy–Schwartz inequality for the first term of the right-hand side of the latter equation, we obtain the following:

$$\begin{aligned} & \nabla F_n^T \left(\mathcal{P}_{U_{\text{ad}}}(u_n - \tau_n \hat{G}_n) - u_n \right) \\ =\ & \hat{G}_n^T \left(\mathcal{P}_{U_{\text{ad}}}(u_n - \tau_n \hat{G}_n) - u_n \right) \\ & + (\nabla F_n - \hat{G}_n)^T \left(\mathcal{P}_{U_{ad}}(u_n - \tau_n \hat{G}_n) - u_n \right) \\ \le\ & -\frac{1}{\tau_n} (u_n - \tau_n \hat{G}_n - u_n)^T \cdot \left(\mathcal{P}_{U_{ad}}(u_n - \tau_n \hat{G}_n) - u_n \right) \\ & + \|\nabla F_n - \hat{G}_n\| \|\mathcal{P}_{U_{ad}}(u_n - \tau_n \hat{G}_n) - u_n\| \\ \le\ & -\frac{1}{\tau_n} \|u_{n+1} - u_n\|^2 + \tau_n \|\nabla F_n - \hat{G}_n\| \|\hat{G}_n\|. \end{aligned}$$

Applying Lemma 11 (c) to the second term yields $\|\mathcal{P}_{U_{\text{ad}}}(u_n - \tau_n \hat{G}_n) - u_n\|^2 \le \tau_n^2 \|\hat{G}_n\|^2$. □

Since the first term in right-hand side of the estimate in the above Lemma is strictly negative while the second term is strictly positive, we may expect a descent, provided $\mathbb{E}[\phi_n]$ is small enough.

Corollary 17 (Convergence result) *We have the following:*

$$\sum_{n=0}^{\infty} \mathbb{E}[\phi_n] < \infty,$$

where $\phi_n := \tau_n \|\nabla F_n - \hat{G}_n\| \|\hat{G}_n\| + \tau_n^2 \frac{C_{\nabla_u f}}{2} \|\hat{G}_n\|^2$.

Proof Since $\|\hat{G}_n\|$ is bounded and $\sum_{n=1}^{\infty} \tau_n^2 < \infty$ by Assumption 6, the result follows by (3.2). □

Before we present our main results, we need the following auxiliary result:

Lemma 18 (**Projection of gradient steps**)

$$\|P_{U_{ad}}(u_n - t\hat{G}_n) - u_n\| \leq \frac{t}{\tau_n}\|u_{n+1} - u_n\| \quad t \geq 0.$$

Proof Define $x := u_n$, $y := \hat{G}_n$, and $\tau := \tau_n$. We assume that $x - \tau y \notin U_{\text{ad}}$ (otherwise the result follows by Lemma 11) and set $n_\tau := x - \tau y - \mathcal{P}_{U_{\text{ad}}}(x - \tau y)$ and

$$H := \{u \in \mathbb{R}^{d_u} \mid u^T n_\tau \leq \mathcal{P}_{U_{\text{ad}}}(x - ty)^T n_\tau\}.$$

Since U_{ad} is convex, we have $U_{\text{ad}} \subset H$, and therefore $\forall u \in \mathbb{R}^{d_u}$ by Lemma 11,

$$\|u - x\| \geq \|\mathcal{P}_H(u) - x\| \geq \|\mathcal{P}_{U_{\text{ad}}}(u) - x\|, \tag{3.3}$$

where $\mathcal{P}_H$ is the orthogonal projection onto H (compare Fig. 3). With $B := \frac{t}{\tau}\left(\mathcal{P}_{U_{\text{ad}}}(x - \tau y) - x\right) + x$ and (3.3), we have the following:

$$\begin{aligned} &\frac{t}{\tau}\|\mathcal{P}_{U_{\text{ad}}}(x - \tau y) - x\| \geq \|\mathcal{P}_H(B) - x\| \\ &= \|\mathcal{P}_H(x - ty) - x\| \geq \|\mathcal{P}_{U_{\text{ad}}}(x - \text{ty}) - x\|. \end{aligned}$$

□

Recalling the characterization of stationary points from Corollary 12, we obtain our first main result:

Theorem 19 (**Convergence result**) *For all* $t \geq 0$,

$$\sum_{n=0}^{\infty} \tau_n \mathbb{E}\left[\|\mathcal{P}_{U_{ad}}(u_n - t\nabla F_n) - u_n\|^2\right] < \infty.$$

Proof First, note that by the compactness of U_{ad} and regularity of F, $F_{\text{inf}} := \inf_{u \in U_{\text{ad}}} F(u) > -\infty$. Summing both sides of the inequality in Lemma 16 up to an $N \in \mathbb{N}$ gives the following:

$$\begin{aligned} F_{\text{inf}} - F_0 &\leq \mathbb{E}\left[F_{N+1}\right] - F_0 = \sum_{n=0}^{N} \mathbb{E}\left[F_{n+1} - F_n\right] \\ &\leq \sum_{n=0}^{N} \left(-\frac{1}{\tau_n}\mathbb{E}\left[\|u_{n+1} - u_n\|^2\right] + \mathbb{E}\left[\phi_n\right]\right). \end{aligned}$$

Hence, by Corollary 17,

$$\sum_{n=0}^{\infty} \frac{1}{\tau_n}\mathbb{E}\left[\|u_{n+1} - u_n\|^2\right] \leq F_0 - F_{\text{inf}} + \sum_{n=0}^{\infty} \mathbb{E}\left[\phi_n\right] < \infty.$$

Using Lemma 11 (ii) together with Lemma 18 gives for all $n \in \mathbb{N}$ sufficiently large,

$$\begin{aligned} &\|\mathcal{P}_{U_{\text{ad}}}(u_n - t\nabla F_n) - u_n\|^2 \\ &\leq \Big(\|\mathcal{P}_{U_{\text{ad}}}(u_n - t\hat{G}_n) - u_n\| \\ &\quad + \|\mathcal{P}_{U_{\text{ad}}}(u_n - t\nabla F_n) - \mathcal{P}_{U_{\text{ad}}}(u_n - t\hat{G}_n)\|\Big)^2 \\ &\leq \left(\frac{t}{\tau_n}\|u_{n+1} - u_n\| + t\|\hat{G}_n - \nabla F_n\|\right)^2. \end{aligned}$$

Since $\|u_{n+1} - u_n\| \leq \tau_n\|\hat{G}_n\|$, this combined with Corollary 15 gives the result. □

As a direct consequence, we have the following:

Theorem 20 (**Main theorem**) *Let* $(u_n)_{n\in\mathbb{N}}$ *be generated by Algorithm 1. Then, there exists a sub-sequence* $(u_{n_k})_{k\in\mathbb{N}}$ *converging to a stationary point, i.e.,*

$$\liminf_{n\to\infty} \mathbb{E}\left[\|\mathcal{P}_{U_{ad}}(u_n - t\nabla F_n) - u_n\|^2\right] = 0 \quad \forall t > 0.$$

Proof Direct consequence of Theorem 19. □

For applications, the condition on the step length (Assumption 6) is inconvenient, since the step length becomes very small and the algorithm thus progresses only slowly. If the algorithm is performed with a constant stepsize, and if the sequence $(u_n)_{n\in\mathbb{N}}$ converges to some $u^* \in U_{\text{ad}}$, then the following theorem demonstrates that u^* is a stationary point.

Theorem 21 (**Convergent series**) *Assume the time-step series* $(\tau_n)_{n\in\mathbb{N}}$ *satisfies* $\tau_n \geq \tau \quad \forall n \in \mathbb{N}$ *for some* $\tau > 0$. *Let further* $(v_n)_{n\in\mathbb{N}}$ *be dense in* V_{ad} *and assume* $(u_n)_{n\in\mathbb{N}}$ *converges to* $u^* \in U_{ad}$. *Then,* u^* *is a stationary point of* F, *i.e.,*

$$\mathbb{E}\left[\|\mathcal{P}_{U_{ad}}(u^* - t\nabla F(u^*)) - u^*\|^2\right] = 0 \quad \forall t > 0.$$

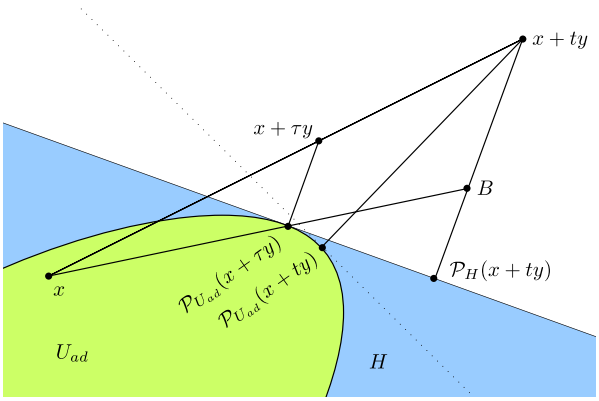

Fig. 3 Illustration of the intercept theorem for the proof of Lemma 17

Proof Similar to Corollary 15, $\|\hat{G}_n - \nabla F_n\| \to 0$. Thus, by convergence of $(u_n)_{n\in\mathbb{N}}$, we obtain the following:

$$\begin{aligned}
&\|\mathcal{P}_{U_{\text{ad}}}(u^* - t\nabla F(u^*)) - u^*\| \\
&= \lim_{n\to\infty} \|\mathcal{P}_{U_{\text{ad}}}(u_n - t\nabla F_n) - u_n\| \\
&\le \lim_{n\to\infty} \Big(\|\mathcal{P}_{U_{\text{ad}}}(u_n - t\hat{G}_n) - u_n\| \\
&\quad + \|\mathcal{P}_{U_{\text{ad}}}(u_n - t\hat{G}_n) - \mathcal{P}_{U_{\text{ad}}}(u_n - t\nabla F_n)\| \Big) \\
&\le \lim_{n\to\infty} \frac{t}{\tau_n} \Big(\|\mathcal{P}_{U_{\text{ad}}}(u_n - \tau_n\hat{G}_n) - u_n\| \\
&\quad + \|\mathcal{P}_{U_{\text{ad}}}(u_n - t\hat{G}_n) - \mathcal{P}_{U_{\text{ad}}}(u_n - t\nabla F_n)\| \Big) \\
&\le \lim_{n\to\infty} \frac{t}{\tau} \|\mathcal{P}_{U_{\text{ad}}}(u_n - \tau_n\hat{G}_n) - u_n\| + \lim_{n\to\infty} t\|\hat{G}_n - \nabla F_n\| \\
&= \frac{t}{\tau} \lim_{n\to\infty} \|u_{n+1} - u_n\| + t \lim_{n\to\infty} \|\hat{G}_n - \nabla F_n\| = 0
\end{aligned}$$

by Lemma 18 and Lemma 11 (c). □

Note that almost all sequences $(v_n)_{n\in\mathbb{N}}$ that are given by the random number generator in Algorithm 1 are dense in V_{ad}. In addition to the convergence properties shown in the latter theorems, the algorithm also approximates the objective functional value with arbitrary accuracy:

Corollary 22 (**Approximation of** F) *Let the series* $(u_n)_{n\in\mathbb{N}}$ *be generated by Algorithm 1. Then, for all convergent subsequences* $(u_{n_k})_{k\in\mathbb{N}}$ *with* $u_{n_k} \to u^*$, *we obtain for* $\hat{F}$ *as defined in Alg. 1:* $|\hat{F}_{n_k} - F(u^*)| \overset{a.s.}{\to} 0$, *assuming further* $(v_{n_k})_{k\in\mathbb{N}}$ *is dense in* V_{ad}, *we obtain* $\lim_{k\to\infty} \hat{F}_{n_k} = F(u^*)$.

Proof The proof is similar to the proof of Lemma 1 relying on the Lipschitz continuity of F—as a direct consequence of Assumption 1—and Corollary 13. □

Remark 23 (**Termination condition**) By the regularity assumption on the objective functional in Assumption 1, the termination condition in Algorithm 1 can be posed as follows:

$$\|\mathcal{P}_{U_{\text{ad}}}\left(u_n - \hat{G}_n\right) - u_n\| \le \epsilon$$

for $\epsilon > 0$. This is obviously not possible for SG. To satisfy such a condition by the SAG method, the discretization of the objective functional has to be sufficiently fine in order to approximate the gradient with sufficiently accurate. However, depending on the particular example, an a priori discretization satisfying this property can be hard to determine.

4 Numerical results

In this section, we will compare the following stochastic optimization methods mentioned in the introductory section:

- CSG (continuous stochastic gradient method): as introduced in Section 2, this scheme relies on the computation of a single gradient in each iteration and the interpolation with previously computed information.
- SG (stochastic gradient method): the classical stochastic optimization scheme as outlined in Bottou et al. (2018). The convergence of the method is based on on decreasing stepsizes.
- SAG (stochastic average gradient method): an improved stochastic gradient scheme as introduced in Schmidt et al. (2017) restricted to the case of a finite sum as objective rather than an integral. The true advantage of possible larger stepsizes can be seen in the examples and is also valid for CSG. We will write SAG_n ($n \in \mathbb{N}$) for an SAG method relying on an n-step quadrature rule to discretize the integral in the original objective.

The performance of these methods strongly depends on the chosen stepsizes. In the following examples, the stepsizes are chosen such that all the schemes have a good performance, in the range of their possibilities. However, adaptive stepsize control (also known as learning rate in the field of machine learning) is itself a subject of research for stochastic optimization schemes and is not the focus of this contribution. For example, in Kingma and Ba (2015), the stepsizes are derived from estimates of first and second moments of the gradients and in Tan et al. (2016), a Barzilai-Borwein-type stepsize adaption is discussed.

To compare the methods, we have chosen the following way to display the results. For a large number of optimization runs, we compute the quantile curves $\alpha_p(n)$ defined as $\mathbb{P}(u_n > \alpha_p(n)) = p$ for $p \in (0, 1)$. With this, we define the quantile sets $P_{p,q}$ which lie in between the p and the q-quantile, i.e.,

$$P_{p,q} := \left\{(n, v) \in \mathbb{N} \times U_{\text{ad}} \colon \alpha_p(n)) < u_n < \alpha_q(n))\right\}. \quad (4.1)$$

These areas will be colored in various degrees of opacity in order to show the behavior of the optimization procedure in its probabilistic nature.

First, we will compare the algorithms by optimizing the function defined in (2.1) and equivalently in (4.2), as well as an additional academic objective functional which will be defined in (4.3).

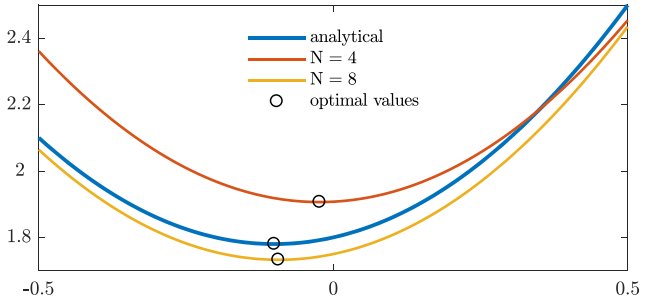

Fig. 4 The analytic function F (blue) given in (4.3) and the function F discretized with 4 (red), 8 (yellow) equidistant discretization points

4.1 Academic examples

We will study the behavior of the algorithms in the following two cases:

$$F(u) := \int_{-1}^{1} \frac{v^2}{(u-v)^2 + 10^{-3}} \, \mathrm{d}v \tag{4.2}$$

as introduced in (2.1) (see Fig. 1) and

$$F(u) := \int_{-1}^{1} (\tanh(10v - 1) - u)^2 \, \mathrm{d}v. \tag{4.3}$$

(see Fig. 4).

To be able to apply the SAG algorithm, we approximate the integral by the trapezoidal rule in both cases. For this, we use an equidistant grid, i.e., for $N \in \mathbb{N}$, $v_0 := -1$, we define $v_i = v_0 + \mathrm{k}h \quad k = 1, \ldots, N+1$ with $h := \frac{|V_{\mathrm{ad}}|}{N} = \frac{2}{N}$. In this way, F is approximated as follows:

$$F(u) = \sum_{i=1}^{N} \int_{v_{i-1}}^{v_i} f\left(u, v_{i+\frac{1}{2}}\right) \mathrm{d}v \approx \frac{2}{m} \sum_{i=1}^{N} f\left(u, v_{i-\frac{1}{2}}\right).$$

The optimization problem considered in the SAG case, thus reads as follows:

$$\min_{u \in U_{\mathrm{ad}}} \frac{2}{N} \sum_{i=1}^{N} f\left(u, v_{i-\frac{1}{2}}\right). \tag{4.4}$$

The approximation error directly depends on the second derivative of f w.r.t., the second argument and the grid-spacing h, that is, the number of intervals. A finer grid thus leads to a better approximation, but also, for a deterministic gradient descent method, to a high number of problems to solve in each iteration.

The comparison of the algorithms is based on the number of function evaluations as these constitute the time-consuming steps for complex optimization tasks. We compare two different settings, one with a stepsize which is proportional to $n^{-0.6}$ (see left column in Figs. 5 and 6)

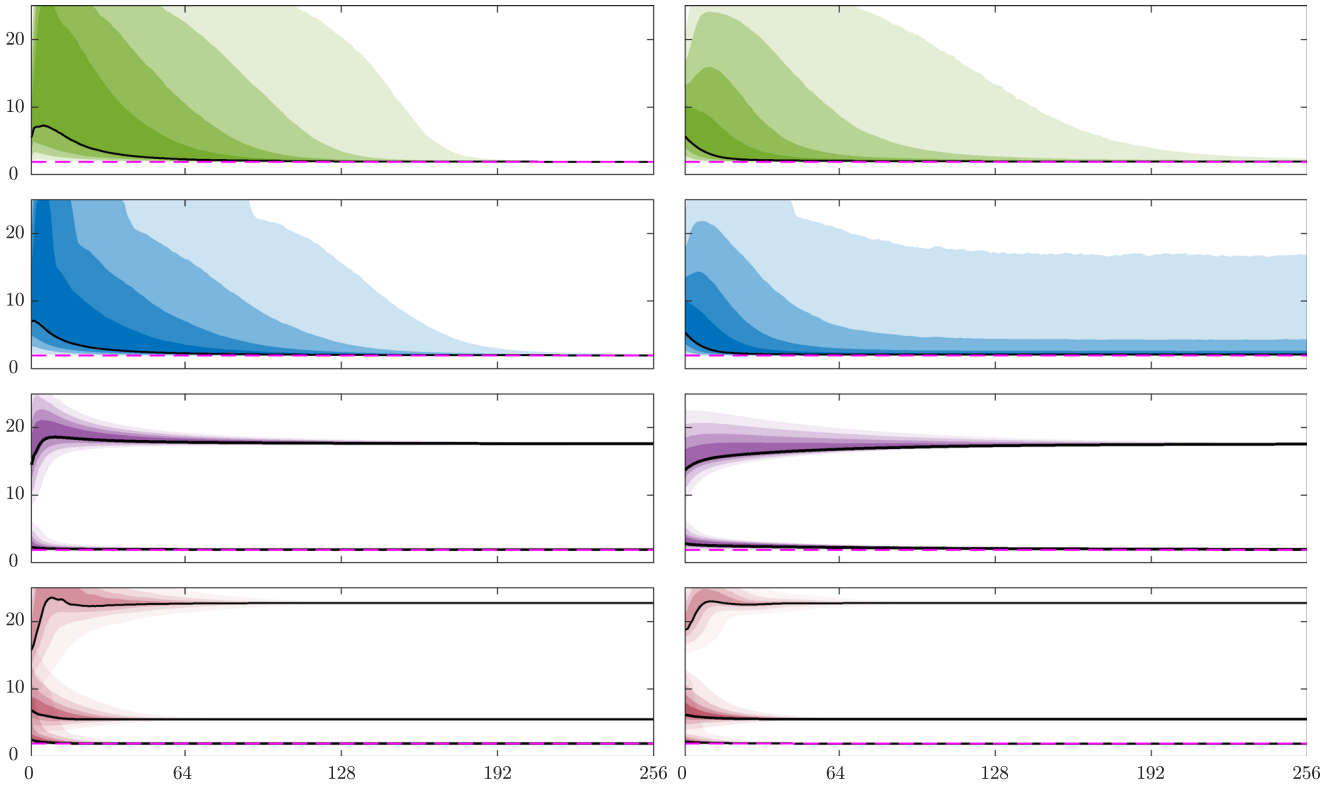

Fig. 5 Comparison of CSG (top row, green), SG (second row, blue), SAG_4 (third row, purple), and SAG_8 (bottom row, red) in the case of objective functional (4.2) with stepsize $\tau_n = 10^{-1} n^{-0.6}$ (left column) as well as constant stepsize $\tau_n = 10^{-4}$ (right column) for 256 optimization steps each. The shaded areas denote from light to dark the quantile sets $P_{0.1,0.9}$, $P_{0.2,0.8}$, $P_{0.3,0.7}$, and $P_{0.4,0.6}$ as defined in (4.1). The dashed magenta line denote the optimal value of F. It is noted that in the case of SAG_4 and SAG_8, convergence towards different local minima introduced by the discretization is observed

and one with an appropriately chosen constant stepsize (see right column in Figs. 5 and 6). The shaded areas in Figs. 5 and 6 denote the quantile sets as defined in (4.1) for the 10 and 90% quantile (light), 20 and 80% quantile and 30 and 70% quantile (medium dark), as well as 40 and 60% quantile (dark). The quantiles are based on 10^5 optimization runs each with randomized initial datum. The black lines in the Figs. 5 and 6 identify the median for CSG and SG. In SAG_4 and SAG_8, they identify the median of the series converging to one of the local minima. In addition to this, the line thickness and the patch opacity is proportional to the probability of converging to the respective local minimum.

Results of numerical experiments In case of objective functional (4.2) optimized by SAG_4 and SAG_8, the algorithm converges to one of the three or five local minima of the discretized objective functional, respectively (see Fig. 1 with $N = 4$ and $N = 8$). In contrast to SAG, SG (in the case of decreasing stepsizes, see Fig. 5, left) and CSG (in both considered cases, see Figs. 5 and 6) converge to the optimal value of F. The "failure" of SAG is due to the fact that SAG is only approximating the original objective functionals. On the other hand, it is clear that for a sufficiently regular function (see Assumptions 1), the optimal values of SAG_N converge to the optimal solution of the original problem for $N \to \infty$. However, a sufficiently large number N is in general not known a priori. Thus, even for a large N, it is not clear how (local) optimal solutions u^* as well as their objective functional values $F(u^*)$ are affected by the discretization of the integral. Moreover, the convergence of the SAG method becomes slower with larger N as can be clearly seen from the more diffuse quantile sets in Figs. 5 and 6.

Figures 5 and 6 clearly show the advantage of CSG as it converges considerably faster in comparison to SG and does not converge to an artificial local minimum as SAG. While SG also approaches the optimal value at least in the case with decreasing stepsize, the speed of convergence is considerably lower compared to CSG. The true potential of CSG comes into play whenever constant stepsizes τ_n are chosen. This can be seen in the right columns of Figs. 5 and 6. It should be noted that the stepsize could be adapted individually for each method, in order to approach a slightly better convergence behavior. In particular, the

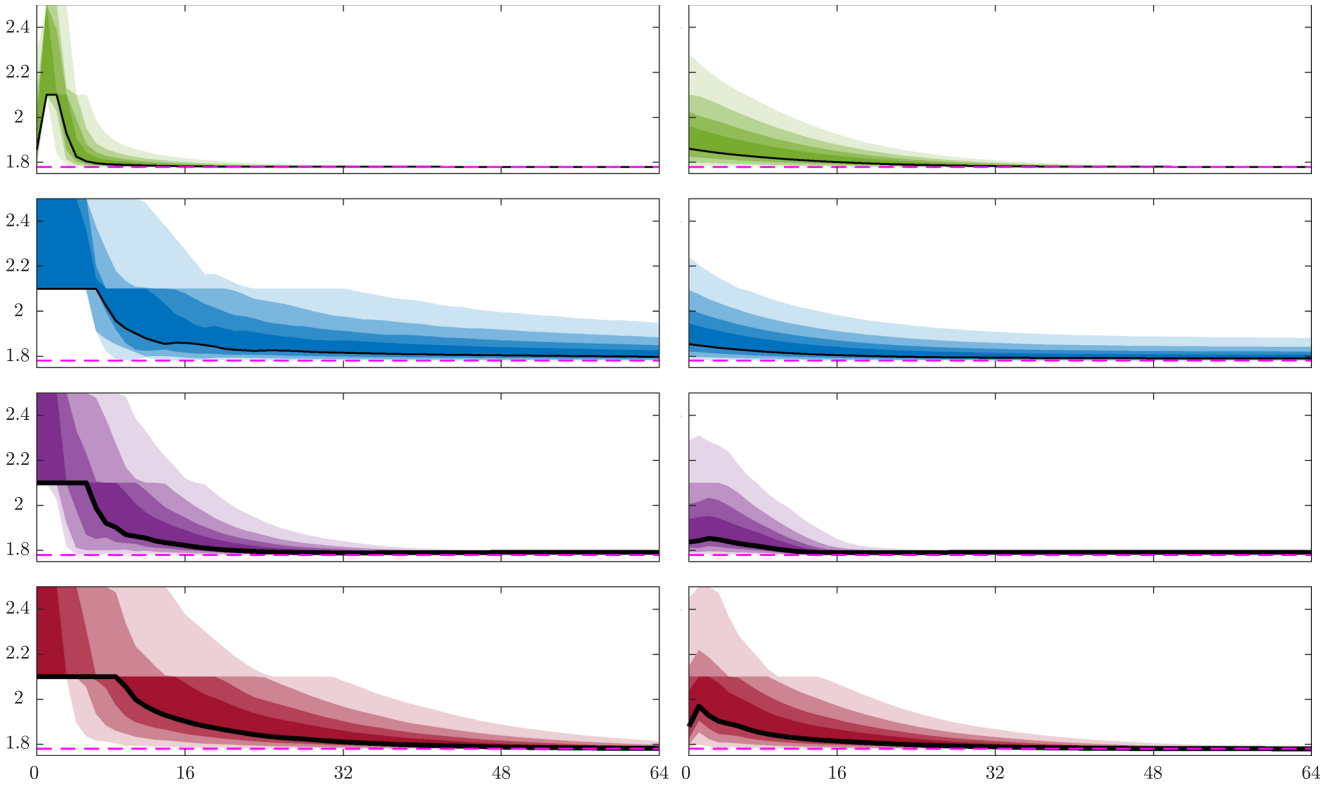

Fig. 6 Comparison of CSG (top row, green), SG (second row, blue), SAG_4 (third row, purple), and SAG_8 (bottom row, red) in the case of objective functional (4.3), stepsize $\tau_n = n^{-0.6}$ (left column) as well as constant stepsize $\tau_n = 10^{-2}$ (right column) for 64 optimization steps each. Note that the stationary point for SAG differs from SG and CSG due to the discretization of the objective functional. The shaded areas indicate, from light to dark, the quantile sets $P_{0.1,0.9}$, $P_{0.2,0.8}$, $P_{0.3,0.7}$, and $P_{0.4,0.6}$ as defined in (4.1). The dashed magenta line denote the optimal value of F. The artificially looking jump in the area plots are due to the non-symmetry of the objective functional

constant stepsize we have chosen uniformly for all methods seems to be too large for SG. On the other hand, CSG can easily handle this stepsize.

Finally, it is observed that CSG combines the advantages of SAG and SG. For instance, in the left column in Fig. 6, SAG_4 converges quickly but unfortunately to the "wrong" result, while SG converges to the correct limit point, but convergence is very slow. In contrast, CSG converges quickly to the correct limit point.

Another advantage of CSG is that according to Theorem 20, Corollary 21 and as discussed in Remark 22, the CSG algorithm can be stopped whenever the objective function is approximated with defined accuracy and the first-order optimality conditions are satisfied within a given error tolerance. This is, in general, neither possible for SG nor for SAG.

4.2 Structural optimization example

As a design optimization test case, we have chosen the optimization of a 2D tire, fixed at the midpoint and loaded from an arbitrary direction. A weighted sum of the expected compliance, a volume penalization term and a regularization term form the objective functional.

In detail, we consider the design domain $\Omega := \{x \in \mathbb{R}^2 \mid 0.1 < \|x\| < 1\}$—an annulus—which is subject to a load described by the function

$$g(x) = h_\alpha(\arctan2(x_1, x_2))n(x) \tag{4.5}$$

on the outer boundary $\Gamma_N \subset \partial\Omega$, and fixed with a homogeneous Dirichlet condition on the inner boundary $\Gamma_D \subset \partial\Omega$ (see Fig. 7). Here, $n(x)$ denotes the outer normal vector in $x \in \partial\Omega$ and $h_\alpha : \mathbb{R} \to \mathbb{R}$ denotes for $\alpha \in \mathbb{R}$ the scalar function:

$$h_\alpha(\beta) = 1 + \tanh\left(10^3\left(\cos(\beta - \alpha) - 1\right) + 10^{-1}\right). \tag{4.6}$$

The angle α describes the position where the boundary force takes its maximum value and the function can be seen as a smoothed Dirac force on the boundary (see Fig. 8).

In Ω, material properties are defined using a pseudo density function ρ which is used to scale a given isotropic material characterized by Lamé parameters λ and μ. Denoting this material by E, the resulting material function is given by the SIMP law $\rho^p E$, with a penalty parameter $p > 1$, see Bendsøe (1989). We assume that the material properties are fixed close to the outer boundary and thus set the pseudo density ρ is set to 1 in this part of the domain, i.e., $\rho|_{\hat{\Omega}} \equiv 1$ with $\hat{\Omega} := \{x \in \mathbb{R}^2 \mid 0.9 < \|x\| < 1\}$ (see Fig. 7, yellow marked area). In the rest of the domain, ρ serves as design variable and is allowed to vary between a small positive value ε and 1.

Now, for each admissible design ρ and each $\alpha \in [0, 2\pi]$ a linear elasticity problem, the so-called state problem is

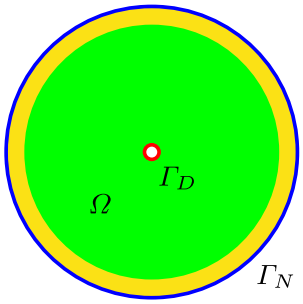

Fig. 7 Design setting for optimization problem (4.8). The normal force defined in (4.5) acts on the Neumann boundary Γ_N; Γ_D is a homogeneous Dirichlet boundary and Ω is subdivided into the green region which is subject to optimization and a yellow region $\hat{\Omega}$ with $\rho = 1$

defined on Ω applying boundary conditions as described above. The corresponding state solution is denoted by $u(\rho, \alpha)$.

The optimization goal is to minimize the expected compliance for angles $\alpha \in [0, 2\pi]$. In addition, we introduce a term to penalize the total material consumption and a filter regularization term as proposed in Semmler et al. (2018, Section 3.2). This leads to the objective functional as follows:

$$\int_0^{2\pi} J(\rho, u(\rho, \alpha))\, \mathrm{d}\alpha,$$

where

$$J(\rho, u) := \gamma_0 \int_{\Gamma_N} u(x) \cdot g(x)\, \mathrm{d}x + \gamma_v \int_\Omega \rho\, \mathrm{d}x + \gamma_\varphi \int_\Omega \left(\rho - \frac{\rho * \varphi}{1 * \varphi}\right)^2 \mathrm{d}x, \tag{4.7}$$

$\gamma_0, \gamma_v, \gamma_\varphi > 0$ are given scaling parameters, " $*$ " denotes a convolution operator in $\mathbb{R}^2$ and the filter kernel $\varphi : \mathbb{R}^2 \to \mathbb{R}$ is defined by the following:

$$\varphi(x) := \max\{0, r_0 - \|x\|\}$$

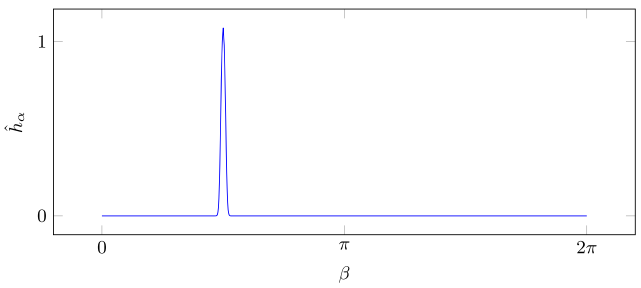

Fig. 8 Plot of h_α as defined in (4.6) for $\alpha = \frac{\pi}{2}$, showing the magnitude of the normal force g defined in (4.5) acting on Γ_N of the domain visualized in Fig. 7

with radius $r_0 > 0$. By finite element discretization, this leads to the following optimization problem:

$$\min_{\rho_h \in [\varepsilon,1]^M} \int_0^{2\pi} j_h(\rho_h, u_h(\rho_h, \alpha))\, \mathrm{d}\alpha,$$
$$s.t. \quad K(\rho_h)u_h(\rho_h, \alpha) = g_h(\alpha),\ \alpha \in [0, 2\pi]. \tag{4.8}$$

Here, M is the number of design variables, $K(\rho_h) \in \mathbb{R}^{N\times N}$ denotes the global stiffness matrix with N degrees of freedom, and $g_h(\alpha) \in \mathbb{R}^N$ the right-hand side of the linear elastic state problem with the load centered in angle α in finite element notation. Moreover, j_h is the discretized equivalent of J. In the following example, the parameters are chosen as follows: $\gamma_0 = 1$, $\gamma_v = 0.1$, $\gamma_\varphi = 1$, $r_0 = 0.05$, and SIMP parameter $p = 3$ (see, e.g., Bendsøe (1989)). As material parameters, we have chosen $\lambda = \mu = 1$ and, choosing $\varepsilon = 10^{-2}$, the void stiffness was defined as 10^{-6}.

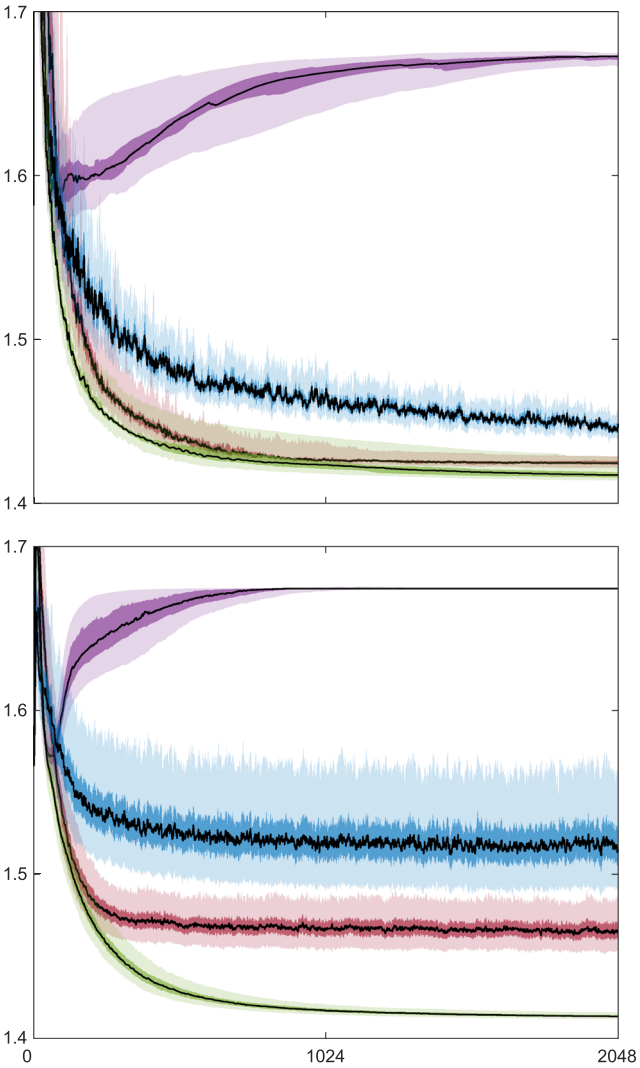

Fig. 9 Comparison of CSG (green), SG (blue), SAG_8 (purple), and SAG_{16} (red) in the case of objective functional (4.7) with $\tau_n = 5 \cdot 10^3 n^{-0.6}$ (top) as well as constant stepsize $\tau_n = 750$ (bottom) and 2048 optimization steps each. Note that the stationary point for SAG differs from SG and CSG due to the discretization of the objective functional. The shaded areas denote the quantile sets $P_{0.1,0.9}$ (light) and $P_{0.25,0.75}$ (dark) as defined in (4.1). This plot clearly shows the advantage of CSG in terms of speed of convergence as well as resulting objective functional value

Computational optimization results As in the academic examples presented in Section 4.1, we show and compare results for SG, SAG, and CSG. All optimization runs have been started with $\rho \equiv \frac{1}{2}$ in the design domain $\Omega \setminus \hat{\Omega}$. The linear elasticity problem is discretized using $\approx 40 \cdot 10^3$ unstructured triangular elements generated by TRIANGLE (see Shewchuk (1996)). Using first-order Lagrange basis functions, this discretization results in approximately the same number of degrees of freedom in terms of u_h. The design domain comprises roughly $20 \cdot 10^3$ degrees of freedom in terms of ρ_h. The implementation is performed in MATLAB and the linear system is solved using the direct solver available through the backslash operator.

Analogous to the previous experiments, we have chosen a suitable initial stepsize and have discretized the integral in (4.8) for SAG using a trapezoidal rule. Again, the SG method is applied to its undiscretized version to omit dependencies on the choice and accuracy of the quadrature rule. For validation and comparison purposes, the objective function is further approximated with the trapezoidal rule using a total of 180 equidistant discretization points to ensure a good approximation for the expected compliance.

In Fig. 9, the distribution of obtained objective functional values for 480 optimization runs with 2048 steps each is compared for the different methods with appropriately chosen stepsize rules. The presented results clearly show the fast convergence of CSG in comparison to the other schemes.

Moreover, in Fig. 10, a rapid convergence of the increasingly better approximated objective function values is observed when applying the CSG method to the structural optimization problem. It is noted that, by construction, this type of convergence can neither be expected for the SG nor for the SAG method.

In Fig. 11, the so-called physical density ρ^p is shown for SAG, SG, and CSG for constant stepsize. While SG does

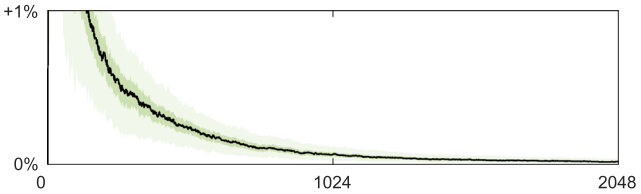

Fig. 10 Relative error between the approximated objective functional by CSG and the objective functional approximated by 180 load cases in the case of the functional stated in (4.7) and constant stepsize. It shows a clear convergence trend of the approximated objective functional value $\hat{F}$ (see Algorithm 1) to the true objective functional value F as predicted by Corollary 21

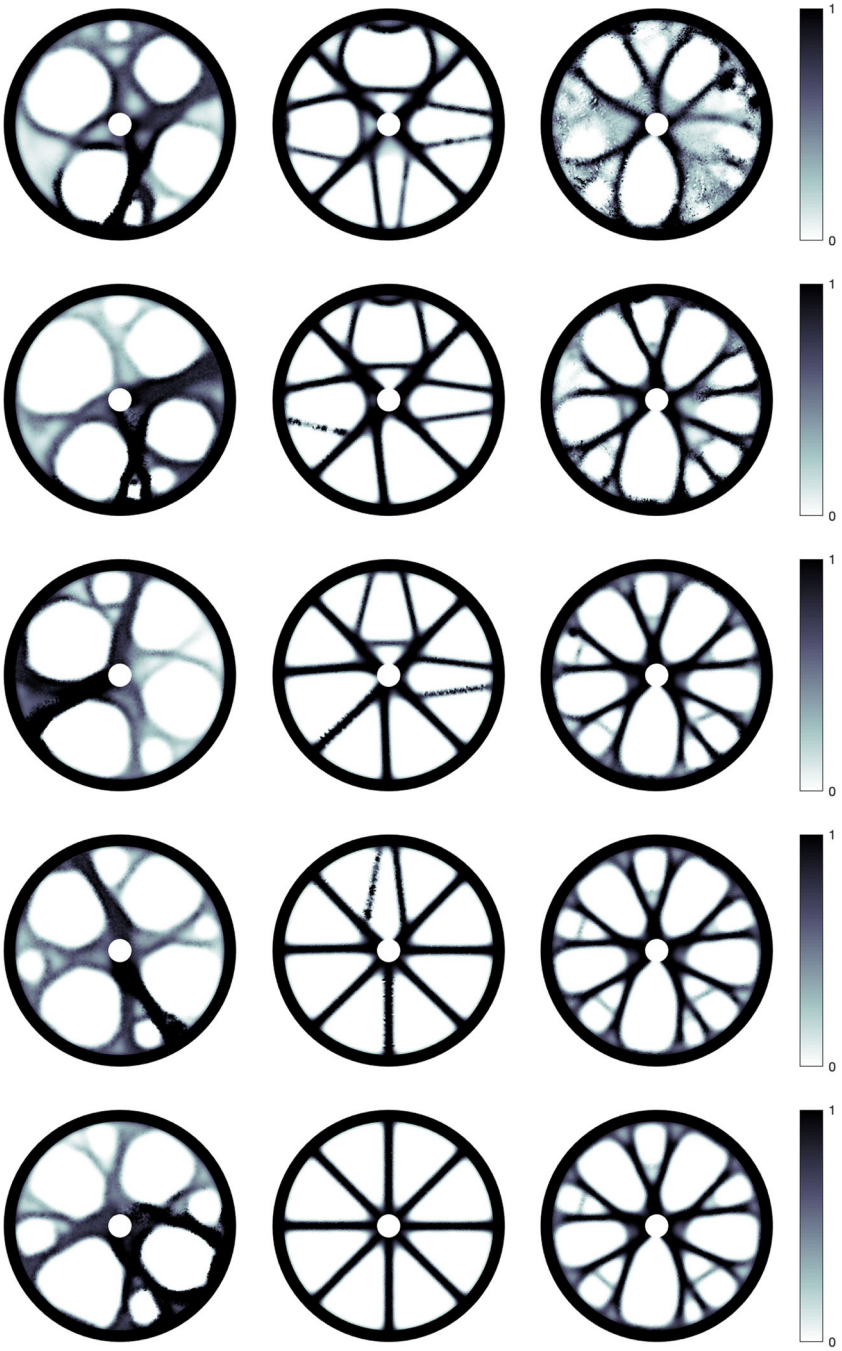

Fig. 11 Comparison of the solution ρ^p for SG (left column), SAG_8 (middle column), and CSG (right column). The rows show from top to bottom the solution of one optimization run after 128,256,512,1024, and 2048 optimization steps in the case of objective functional (4.7) and optimization problem defined in (4.8) with suitable constant stepsize. In the middle column, one can clearly see the eight load cases applied in the discretized objective functional of SAG_8. For each optimization method, the run with the lowest resulting objective functional was chosen

not converge in 2048 iteration, SAG converges, though the resulting density distribution is strongly influenced by the discretization of the objective functional—note the 8 struts corresponding to the 8 discrete load cases applied (see the middle column of Fig. 11). No such effect is observed in the case of the CSG result (right column of Fig. 11). Moreover, the CSG result appears to be much clearer compared to the SG result, which is due to faster convergence.

It is finally noted that, in the interest of comparability, we have not applied any continuation scheme (e.g., SIMP parameter $p \to \infty$), which would result in a true "black and white" solution, i.e., $\rho(x) \in \{0, 1\}\ \forall x \in \Omega \setminus \tilde{\Omega}$. This is of course necessary to achieve a physical interpretable and manufacturable solution. We leave the question of defining a continuation scheme suitable for CSG as a subject for further research.

5 Conclusion and outlook

In this work, we introduced the continuous stochastic gradient method, which is applicable for the solution of a

broad class of structural optimization problems. Preliminary experiments with notorious academic examples as well as an application from mechanics, in which an elastic structure has been optimized with respect to infinitely many load cases, revealed that the CSG method performed better than both, the traditional SG method and its relative, SAG, in the sense that a significantly lower function value could be obtained in a defined number of iterations. This is particularly interesting as the CSG algorithm requires roughly the same computational effort per optimization iteration as the SG and the SAG method. Moreover, like the SAG method, it benefits from gradient information obtained in previous iterations. Importantly, the CSG method does not require an a priori discretization of integrals entering the objective function, and the function value and gradient can be approximated with arbitrary precision throughout the optimization iterations. The latter results in the CSG method approaching a full gradient method throughout the course of the iterations.

While the CSG method appears promising, to obtain a full picture of its behavior when applied to practical applications, more examples, e.g., from robust optimization, acoustics, or optics should be tested in future.

Furthermore, from a theoretical point of view, an analysis covering the convergence rate for convex functions would be helpful to provide a deeper understanding of the algorithm. It should also be noted that the computational effort of computing the gradient weights $\alpha_0^n, \ldots, \alpha_n^n$ grows with each iteration. As compensation, an efficient implementation of the algorithm is crucial. While this can easily be done in the case of a one-dimensional index set V_{ad}, the question remains how this can be achieved in higher dimensions. Other interesting questions center around how Lipschitz constants can be estimated throughout the optimization process and how the stepsize can be automatically adapted.

Finally, similarly as suggested in De et al. (2020), a combination with established structural optimization algorithms as, for instance, GCMAA is conceivable.

Acknowledgments Open Access funding provided by Projekt DEAL. In additon, we thank J. Rodestock for his proofreading of this paper.

Funding information This work has been supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project-ID 416229255 - SFB 1411 and by DFG through the Cluster of Excellence "Engineering of Advanced Materials" at FAU Erlangen-Nürnberg.

Compliance with ethical standards

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

Replication of results All parameters and data required for the application of the CSG, SG, and SAG algorithms are given in the numerical section of this article. Moreover, the CSG algorithm is outlined in full length in Section 2. Finally, standard implementations of the SG and the SAG algorithm have been used.

References

Alvarez F, Carrasco M (2005) Minimization of the expected compliance as an alternative approach to multiload truss optimization. Struc Multidiscip Optim 29(6):470–476

Aubin JP (2000) Applied functional analysis, 2nd edn. Pure and Applied Mathematics New York, Wiley-Interscience, New York, with exercises by Bernard Cornet and Jean-Michel Lasry, Translated from the French by Carole Labrousse

Bendsøe MP (1989) Optimal shape design as a material distribution problem. Struc Optim 1(4):193–202

Bottou L, Cun YL (2004) Large scale online learning. In: Advances in neural information processing systems, pp 217–224

Bottou L, Curtis FE, Nocedal J (2018) Optimization methods for large-scale machine learning. Siam Rev 60(2):223–311

Bürgisser P, Cucker F (2013) Condition: the geometry of numerical algorithms, vol 349. Springer Science & Business Media, Berlin

Conti S, Held H, Pach M, Rumpf M, Schultz R (2009) Shape optimization under uncertainty—a stochastic programming perspective. SIAM J Optim 19(4):1610–1632

De S, Hampton J, Maute K, Doostan A (2020) Topology optimization under uncertainty using a stochastic gradient-based approach. Struct Multidiscip Optim. https://doi.org/10.1007/s00158-020-02599-z

Dilgen CB, Dilgen SB, Aage N, Jensen JS (2019) Topology optimization of acoustic mechanical interaction problems: a comparative review. Struct Multidiscip Optim 60(2):779–801

Dudley RM (1969) The speed of mean Glivenko-Cantelli convergence. Annals Math Stat 40(1):40–50

Fortune S (1995) Voronoi diagrams and delaunay triangulations. In: Computing in euclidean geometry. World Scientific, Singapore, pp 225–265

Haber E, Chung M, Herrmann F (2012) An effective method for parameter estimation with PDE constraints with multiple right-hand sides. SIAM J Optim 22(3):739–757

Jahn J (2007) Introduction to the theory of nonlinear optimization. Springer Science & Business Media, New York

Jensen J, Sigmund O (2011) Topology optimization for nano-photonics. Laser Photonics Rev 5(2):308–321

Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. CoRR abs/1412.6980

Klenke A (2013) Probability theory: a comprehensive course. Springer Science & Business Media, New York

Lazarov BS, Schevenels M, Sigmund O (2012) Topology optimization considering material and geometric uncertainties using stochastic collocation methods. Struct Multidiscip Optim 46(4):597–612

Maute K, Frangopol DM (2003) Reliability-based design of MEMS mechanisms by topology optimization. Comput Struc 81(8):813–824. K.J Bathe 60th Anniversary Issue

Robbins H, Monro S (1951) A stochastic approximation method. The Annals of Mathematical Statistics 22(3):400–407

Roosta-Khorasani F, van den Doel K, Ascher U (2014) Stochastic algorithms for inverse problems involving PDEs and many measurements. SIAM J Sci Comput 36(5):S3–S22

Schmidt M, Le Roux N, Bach F (2017) Minimizing finite sums with the stochastic average gradient. Math Program 162(1):83–112

Semmler J, Pflug L, Stingl M, Leugering G (2015) Shape optimization in electromagnetic applications. In: New trends in shape optimization. Springer International Publishing, pp 251–269

Semmler J, Pflug L, Stingl M (2018) Material optimization in transverse electromagnetic scattering applications. SIAM J Sci Comput 40(1):B85–B109

Shewchuk JR (1996) Triangle: engineering a 2D quality mesh generator and Delaunay triangulator. In: Workshop on applied computational geometry. Springer, pp 203–222

Svanberg K (2002) A class of globally convergent optimization methods based on conservative convex separable approximations. SIAM J Optim 12(2):555–573

Tan C, Ma S, Dai YH, Qian Y (2016) Barzilai-Borwein step size for stochastic gradient descent. In: Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R (eds) Advances in neural information processing systems, vol 29. Curran Associates Inc., pp 685–693

Zhang XS, de Sturler E, Paulino GH (2017) Stochastic sampling for deterministic structural topology optimization with many load cases: density-based and ground structure approaches. Comput Methods Appl Mech Eng 325:463–487