Random directions stochastic approximation with deterministic perturbations

Prashanth L A[*1], Shalabh Bhatnagar[†2], Nirav Bhavsar[‡1], Michael Fu[§3] and Steven I. Marcus[¶4]

[1]Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai
[2]Department of Computer Science and Automation and the Robert Bosch Centre for Cyber Physical Systems, Indian Institute of Science, Bangalore
[3]Robert H. Smith School of Business & Institute for Systems Research, University of Maryland, College Park, Maryland
[3]Department of Electrical and Computer Engineering & Institute for Systems Research, University of Maryland, College Park, Maryland

Abstract

We introduce deterministic perturbation schemes for the recently proposed random directions stochastic approximation (RDSA) [19], and propose new first-order and second-order algorithms. In the latter case, these are the first second-order algorithms to incorporate deterministic perturbations. We show that the gradient and/or Hessian estimates in the resulting algorithms with deterministic perturbations are asymptotically unbiased, so that the algorithms are provably convergent. Furthermore, we derive convergence rates to establish the superiority of the first-order and second-order algorithms, for the special case of a convex and quadratic optimization problem, respectively. Numerical experiments are used to validate the theoretical results.

1 Introduction

We consider the following problem:

$$\text{Find } x^* = \arg \min_{x \in \mathbb{R}^N} f(x). \tag{1}$$

We operate in a setting in which the analytical form of the objective function f is not known, but noisy measurements of the function can be obtained. Furthermore, noisy estimates of the objective function gradient are not directly available, so the function gradient needs to be estimated using the aforementioned noisy measurements. Robbins and Monro [21] developed an incremental-update algorithm that estimates the zeros of the function f when only its noisy measurements are available. This algorithm has found applications in several engineering domains such as signal processing, manufacturing, communication networks, autonomous systems, vehicular traffic networks, etc., where it is often used to find either (a) the fixed points of a certain function or (b) the optima of a certain objective given noisy function measurements.

[*]prashla@cse.iitm.ac.in

[†]shalabh@iisc.ac.in

[‡]cs17s016@smail.iitm.ac.in

[§]mfu@isr.umd.edu

[¶]marcus@umd.edu

The earliest gradient search algorithm in this setting is the Kiefer-Wolfowitz [17] procedure. This, however, requires $2N$ function measurements when the parameter dimension is N. Katkovnik and Kulchitsky [16, 22] proposed a random search technique that became known as the smoothed functional (SF) algorithm. The key idea here is that the convolution of the objective function gradient with a multivariate Gaussian PDF is seen via an integration-by-parts argument as the convolution of the objective function itself with a scaled multivariate Gaussian. Thus, a single noisy function measurement at a perturbed value of the parameter update, perturbed using a multivariate Gaussian, is sufficient to obtain an estimate of the full gradient. This results in a one-measurement estimator that however has high bias. A balanced two-sided estimator of the gradient (requiring two function measurements) that has significantly lower bias than the one-measurement SF estimator was proposed in [27], see also [7] for comparisons of the one-measurement and two-measurement SF algorithms.

Random directions stochastic approximation (RDSA) [18] is another gradient search procedure, in which the perturbation variables are considered to be uniformly distributed over the surface of the unit sphere in $\mathbb{R}^N$. Obtaining these perturbation random variables is, however, computationally expensive, particularly when the dimension N is large. In a landmark paper, Spall [26] introduced the simultaneous perturbation stochastic approximation algorithm, a random search technique that estimates the gradient using random perturbations that are independent, symmetric, zero-mean and satisfying an inverse moment bound. The most commonly used and studied class of perturbations within this category are those that are independent, symmetric, ± 1-valued, Bernoulli random variables. This algorithm (the standard SPSA, as it is known), requires two function measurements at each update step, and became popular because of its computational simplicity, as well as the convergence and rate guarantees that it provides. In another paper [15], Spall presented a one-measurement counterpart of SPSA. This algorithm, however, does not show good performance, as it suffers from a large bias in its gradient estimates. Bhatnagar et al. [5] presented certain deterministic perturbation variants of SPSA. Here two constructions for the perturbation variates were proposed, of which, a construction based on Hadamard matrices is seen to show remarkable improvements in the empirical performance of one-measurement SPSA.

Adaptive Newton-type schemes that estimate the Hessian using noisy objective function measurements, as with the gradient, have also gathered considerable attention over the years. The earliest such scheme, due to Fabian [10], estimated the Hessian using finite-difference estimates and required $O(N^2)$ samples of the objective function at each update epoch. Spall [24], presented a simultaneous perturbation estimate of the Hessian that was based on four noisy function measurements. Two of these measurements also estimate the gradient. In the case when noisy gradient measurements are directly available, he also presented a Newton scheme requiring three measurements. Bhatnagar [1] presented three additional algorithms that estimate the Hessian as well as the gradient, using simultaneous perturbation estimates. In the process, new gradient and Hessian SPSA estimators were developed. Bhatnagar and Prashanth [3] presented a balanced estimator of the Hessian using three function measurements. This paper also presented two algorithms, one of which estimated the inverse Hessian using a recursive procedure based on the Sherman-Morrison-Woodbury lemma, while the other did not require one to compute or estimate the inverse Hessian at each step. It was shown nonetheless that the asymptotic behaviour of the latter algorithm is analogous to a Newton algorithm that would involve a computation of the inverse Hessian matrix at each update step. Spall [25] presented enhancements to the four-simulation Hessian estimator of [24] using certain weighting and feedback mechanisms. These enhancements are seen to improve the performance of the resulting scheme.

The class of SF algorithms was extended by Bhatnagar [2] to include two Newton-based algorithms governed by standard Gaussian perturbations. As with the gradient estimator, the Hessian estimator was obtained from the idea that if one convolves the Hessian with a multivariate Gaussian density, then from an integration-by-parts argument applied twice, the same can be viewed as a convolution of the objective function with a scaled multivariate Gaussian. This results in a single-measurement Hessian estimator - the same measurement also estimates the gradient. A two-measurement SF algorithm presented there, involving

a balanced (two-measurement) Hessian estimator, is seen to work better in practice - again the same two measurements also estimate the gradient. In [12, 8], gradient and Newton SF algorithms based on the multi-variate q-Gaussian density as the smoothing functional, i.e., the perturbation distribution, have been presented. This gives rise to a class of smoothing densities parameterized by the q-parameter. Densities such as multivariate Normal, Cauchy and Uniform that were known to satisfy the properties required of smoothing functionals [23] in SF algorithms, emerge as special cases of the q-Gaussian density for different values of the parameter q. Thus, these papers have served to significantly extend the class of perturbations that play the role of smoothing densities in SF algorithms.

Finally, the RDSA procedure has recently been revisited in detail by Prashanth et al. [19], and novel gradient and Newton algorithms have been devised. Recall that in the original RDSA procedure described in [18], the perturbation variates are required to be uniformly distributed over the surface of the unit sphere in $\mathbb{R}^N$, N being the parameter dimension. The approach taken in [19] involves a uniform distribution over a unit cube as opposed to the surface of the unit sphere. The perturbation (component) random variables are thus allowed to be independent, symmetric, and uniformly distributed over an interval that is symmetric around zero. Another class of perturbations, namely asymmetric Bernoulli, have been investigated and found to work nearly as well as SPSA in both theory and practice. Hessian estimators derived from these perturbations have also been proposed in [19], and both gradient and Newton algorithms have been investigated in detail. The reader is referred to [4] for a rigorous introduction to the class of simultaneous perturbation methods.

In this paper, we are concerned with developing deterministic perturbation variants of first and second-order RDSA algorithms, henceforth referred to as RDSA-DP family of algorithms, with 1RDSA-DP (resp. 2RDSA-DP) denoting first (resp. second) order variants. The principal aim is to incorporate deterministic perturbation sequences into RDSA, such that the resulting gradient estimates are still asymptotically unbiased and the overall stochastic gradient algorithm converges, preferably at the same rate as that of the random perturbation RDSA counterparts. We consider two novel choices for deterministic perturbations - a semi-lexicographic sequence and a permutation matrix-based sequence. We combine the two sequences with first and second-order RDSA.

In the case of 1RDSA-DP, the resulting algorithms, under both choices for deterministic perturbations, possess theoretical guarantees that are comparable to those of their random perturbation counterparts. This statement is true when we consider the asymptotic unbiasedness of the gradient estimation and asymptotic convergence of the overall 1RDSA-DP family of algorithms. Moreover, from a non-asymptotic bound that we derive for the special case of strongly-convex objective functions, we observe that the permutation matrix-based perturbations perform best, and even match the rate of a first-order method, whose gradients are directly available.

In the case of second-order RDSA, we incorporate both perturbation sequences to arrive at two variants of 2RDSA-DP, say 2RDSA-Lex-DP and 2RDSA-Perm-DP. However, the theoretical guarantees for the two variants differ significantly. For 2RDSA-Lex-DP, the asymptotic unbiasedness claim holds for the full Hessian, while a similar claim holds only for the Jacobi variant of 2RDSA-Perm-DP involving a diagonal matrix with diagonal elements being those of the Hessian. Furthermore, for the special case of a quadratic optimization problem in the noise-free regime, 2RDSA-Lex-DP is shown to exhibit a convergence rate that is comparable to that of 2SPSA with an adaptive feedback sequence that was proposed in [25]. Note that a similar rate result does not exist for regular 2RDSA, and we believe, cannot be established. In any case, to the best of our knowledge, no deterministic perturbation sequences exist for the class of second-order simultaneous perturbation algorithms, including the popular 2SPSA [15].

In comparison to [5], which is the closest related work, we remark that (i) we propose a novel deterministic perturbation scheme and combine it with first-order and second-order RDSA, while the deterministic perturbation schemes in [5] are only for first-order SPSA; (ii) unlike [5], we provide asymptotic normality results that quantify the convergence rate; and (iii) the permutation matrix-based perturbations that we

propose are much easier to implement and require much less computational memory in comparison to the deterministic perturbation sequences proposed in [5]. In particular, the permutation matrices have a linear dependence on the dimension N, while the lexicographic/Hadamard matrix-based perturbations in [5] scale exponentially with N.

The rest of the paper is organized as follows: Section 2 presents the first-order RDSA variants with two deterministic perturbation sequences, and Section 3 describes deterministic perturbation variants of the second-order RDSA algorithm. The main theoretical guarantees for 1RDSA-DP and 2RDSA-DP algorithms are presented in Sections 2–3, while Section 4 provides detailed convergence proofs. Section 5 presents simulation experiments that compare the performance of the DP variants of RDSA with several algorithms that employ the simultaneous perturbation technique. Finally, Section 6 presents the concluding remarks.

2 First-order RDSA with deterministic perturbations (1RDSA-DP)

A first-order method, given the gradient $\nabla f(\cdot)$, would feature an incremental update as follows:

$$x_{n+1} = x_n - a_n \nabla f(x_n). \tag{2}$$

In the simulation optimization setting, we are given noisy function measurements, from which the gradient has to be estimated. The simultaneous perturbation method [4] is a popular approach for obtaining such gradients. The recently proposed RDSA algorithm estimates $\nabla f(x_n)$ as follows:

$$\widehat{\nabla} f(x_n) = \frac{1}{1+\epsilon} d_n \left[\frac{y_n^+ - y_n^-}{2\delta_n} \right], \tag{3}$$

where $y_n^\pm = f(x_n \pm \delta_n d_n) + \xi$, ξ is the measure noise and δ_n is a perturbation constant. Further, $d_n = (d_n^1, \ldots, d_n^N)^\mathsf{T}$ is the random perturbation vector, with d_n^i, $i = 1, \ldots, N$ chosen using the asymmetric Bernoulli distribution, i.e., $d_n^i = -1$ with probability (w.p.) $\dfrac{(1+\epsilon)}{(2+\epsilon)}$ and $1 + \epsilon$ w.p. $\dfrac{1}{(2+\epsilon)}$ for some $\epsilon > 0$.

In this paper, we propose a variant of 1RDSA that loops through a deterministic sequence to cancel out the bias in the gradient estimate – a property that regular RDSA achieves in expectation through a zero-mean random perturbation. We consider two deterministic constructions for the perturbations d_n. The first choice is based on a semi-lexicographic sequence, while the second employs permutation matrices. In both cases, we perform gradient descent similar to (2), with a gradient estimate inspired from that of 1RDSA. However, unlike (3) that has a random source for perturbations d_n, we loop through a deterministic sequence (cf. Tables 1 and 2 below).

2.1 Semi-lexicographic sequence-based perturbations

Algorithm 1 presents the pseudocode for the 1RDSA-Lex-DP algorithm that employs a semi-lexicographic sequence for perturbations. Our proposed construction for perturbations d_m is illustrated for the case when $N = 2$ and $N = 3$ in Tables 1a and 1b, respectively. Letting $\mathcal{I}_N$ denote the $N \times N$ identity matrix, for $N = 2$, we have

$$\sum_{m=0}^{3^2-1} d_m d_m^\mathsf{T} = \begin{bmatrix} 18 & 0 \\ 0 & 18 \end{bmatrix} \implies \frac{1}{2 \times 3^2} \sum_{m=0}^{3^2-1} d_m d_m^\mathsf{T} = \mathcal{I}_2.$$

In a similar fashion, for $N = 3$, we have

$$\sum_{m=0}^{3^3-1} d_m d_m^\mathsf{T} = \begin{bmatrix} 54 & 0 & 0 \\ 0 & 54 & 0 \\ 0 & 0 & 54 \end{bmatrix} \implies \frac{1}{2 \times 3^3} \sum_{m=0}^{3^3-1} d_m d_m^\mathsf{T} = \mathcal{I}_3.$$

4

Table 1: Illustration of the deterministic perturbation sequence construction for two-dimensional and three-dimensional settings.

(a) Case $N = 2$

Inner loop counter m	D_2^1	D_2^2
0	-1	-1
1	-1	-1
2	-1	2
3	-1	-1
4	-1	-1
5	-1	2
6	2	-1
7	2	-1
8	2	2

(b) Case $N = 3$

Inner loop counter m	D_3^1	D_3^2	D_3^3	Inner loop counter m	D_3^1	D_3^2	D_3^3	Inner loop counter m	D_3^1	D_3^2	D_3^3
0	-1	-1	-1	9	-1	-1	-1	18	2	-1	-1
1	-1	-1	-1	10	-1	-1	-1	19	2	-1	-1
2	-1	-1	2	11	-1	-1	2	20	2	-1	2
3	-1	-1	-1	12	-1	-1	-1	21	2	-1	-1
4	-1	-1	-1	13	-1	-1	-1	22	2	-1	-1
5	-1	-1	2	14	-1	-1	2	23	2	-1	2
6	-1	2	-1	15	-1	2	-1	24	2	2	-1
7	-1	2	-1	16	-1	2	-1	25	2	2	-1
8	-1	2	2	17	-1	2	2	26	2	2	2

For any N, we require that $\frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} d_m d_m^\mathsf{T} = \mathcal{I}_N$ to ensure that the gradient estimate $\widehat{\nabla} f(x_n)$ (see (5) in Algorithm 1 is asymptotically unbiased. The crucial ingredient in the asymptotic-unbiasedness proof, presented later in Lemma 2, is the following step that uses suitable Taylor's series expansions:

$$d_m \left[\frac{f(x_n + \delta_n d_m) - f(x_n - \delta_n d_m)}{2\delta_n} \right] = d_m d_m^\mathsf{T} \nabla f(x_n) + O(\delta_n^2).$$

Hence, if the product $d_m d_m^\mathsf{T}$ sums to identity over a loop, then $\widehat{\nabla} f(x_n)$ would be asymptotically unbiased.

We now present the deterministic perturbation sequence for a general N. Set $D_1^1 = \begin{bmatrix} -1 \\ -1 \\ 2 \end{bmatrix}$ and apply

the following recursion $N - 1$ times to obtain D_N^i:

$$D_{k+1}^i = \begin{bmatrix} D_k^{i-1} \\ D_k^{i-1} \\ D_k^{i-1} \end{bmatrix}, \quad i = 2, \dots, k+1. \tag{4}$$

The deterministic perturbation sequence loops through the rows in the matrix, say D_N, with columns D_N^i. Notice that each column $D_N^i, i = 1, \dots, N$ in D_N is of length 3^N. Further, in the first column of D_N, the

first 2×3^k elements are -1 and the remaining 3^k elements are 2. On the other hand, the columns 2 through N in D_N are obtained from $D_{N-1}^1, \ldots, D_{N-1}^{N-1}$, respectively by concatenating the D_{N-1}^i columns thrice.

Algorithm 1 1RDSA-Lex-DP

Input: initial parameter $x_0 \in \mathbb{R}^N$, perturbation constants $\delta_n > 0$, step-sizes a_n, deterministic perturbations $\{d_0, \ldots, d_{3^N-1}\}$.

for $n = 0, 1, 2, \ldots$ **do**

$\qquad\qquad\qquad\qquad\qquad$ ▷ *Fix x_n and loop through the rows of matrix D_N for perturbations d_m.*

$\quad$ **for** $m = 0, 1, 2, \ldots, 3^N - 1$ **do**

$\qquad$ Obtain function values $y_m^+ = f(x_n + \delta_{n3^N+m} d_m) + \xi$ and $y_m^- = f(x_n - \delta_{n3^N+m} d_m) + \xi$, where ξ is the measure noise.

$\qquad$ Set $g_m = d_m \left[\dfrac{y_m^+ - y_m^-}{2\delta_{n3^N+m}} \right]$.

$\quad$ **end for**

$$\text{Gradient estimate:} \qquad \widehat{\nabla} f(x_n) = \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} g_m. \qquad (5)$$

$$\text{Parameter update:} \qquad x_{n+1} = x_n - a_n \widehat{\nabla} f(x_n). \qquad (6)$$

end for
Return x_n.

2.2 Permutation matrix-based perturbations

While the semi-lexicographic sequence-based perturbations result in a gradient estimate that is asymptotically unbiased, the inner loop for the perturbations becomes exponentially longer as a function of the dimension N. This exponential dependence on N is problematic, because the descent in parameter x_n occurs at the end of the inner loop (see Algorithm 1), and hence, a long inner loop would imply slow updates (and slow convergence) to x_n.

In this section, we propose an efficient alternative to the semi-lexicographic deterministic sequence; the approach is based on permutation matrices. A permutation matrix is a matrix whose rows are the rows of an identity matrix in some order. For instance, the permutation matrices in two dimension are

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

In three dimensions, there are 6 permutation matrices. In general, there are $N!$ permutation matrices in dimension N.

In the case of permutation matrix-based deterministic perturbations, the overall algorithm follows the template provided in Algorithm 1, except that the perturbations are generated using a permutation matrix in N-dimensions, the inner loop for m runs from 0 to $N - 1$ and the gradient estimate in (5) is replaced by

$$\widehat{\nabla} f(x_n) = \sum_{m=0}^{N-1} g_m. \qquad (7)$$

Table 2 illustrates the perturbations d_m used in Algorithm 1, for $N = 2$ and $N = 3$. In a nutshell, the sequence shown in Table 2 loops through the rows of the identity matrix in some order.

6

Table 2: Illustration of the permutation matrix-based deterministic perturbation sequence construction for two-dimensional and three-dimensional settings.

(a) Case $N = 2$

Inner loop counter m	D_2^1	D_2^2
0	1	0
1	0	1

(b) Case $N = 3$

Inner loop counter m	D_3^1	D_3^2	D_3^3
0	0	1	0
1	0	0	1
2	1	0	0

Remark 1. *The classic Kiefer-Wolfowitz (K-W) algorithm [17] obtains $2N$ function samples per iteration, corresponding to parameters $x_n \pm \delta_n e_i$, $i = 1, \ldots, N$ and updates the parameter as follows:*

$$x_{n+1}^i = x_n^i - a_n \left(\frac{y_n^{i+} - y_n^{i-}}{2\delta_n} \right),$$

where $y_n^{i\pm} = f(x_n \pm \delta_n e_i)$, $i = 1, \ldots, N$.

The 1RDSA-Perm-DP algorithm that we propose resembles K-W in the sense that the inner loop obtains $2N$ samples before updating the parameter x_n. However, the gradient estimate features a product with the perturbation vector d_m and this is unlike K-W, where the individual coordinates are independently updated.

2.3 Main results

Let $D_N^1, \ldots, D_N^N$ denote the N columns of the semi-lexicographic perturbation variables. Consider the matrix

$$M_N = \begin{bmatrix} (D_N^1)^\mathsf{T} D_N^1 & (D_N^1)^\mathsf{T} D_N^2 & \ldots & (D_N^1)^\mathsf{T} D_N^N \\ (D_N^2)^\mathsf{T} D_N^1 & (D_N^2)^\mathsf{T} D_N^2 & \ldots & (D_N^2)^\mathsf{T} D_N^N \\ \vdots & \vdots & & \vdots \\ (D_N^N)^\mathsf{T} D_N^1 & (D_N^N)^\mathsf{T} D_N^2 & \ldots & (D_N^N)^\mathsf{T} D_N^N \end{bmatrix}.$$

Lemma 1. *For 1RDSA-Lex-DP $M_N = 2 \times 3^N \mathcal{I}_N$, and for 1RDSA-Perm-DP $M_N = \mathcal{I}_N$.*

Proof. See Section 4.1.1. $\qquad\square$

Before providing the convergence claims for 1RDSA-DP with either perturbation choice, we outline the necessary assumptions below.

(A1) $f : \mathbb{R}^N \to \mathbb{R}$ is three-times continuously differentiable[1] with $\left| \nabla_{i_1 i_2 i_3}^3 f(x_n) \right| < \alpha_0 < \infty$, for $i_1, i_2, i_3 = 1, \ldots, N$ and for all n.

(A2) $\{\xi_m^+, \xi_m^-, m = 0, \ldots, P, n = 1, 2, \ldots\}$ satisfy $\mathbb{E}\left[\xi_m^+ - \xi_m^- \mid \mathcal{F}_n\right] = 0$, where $P = 3^N - 1$ for semi-lexicographic 1RDSA-DP and $P = N - 1$ for permutation matrix-based 1RDSA-DP.

(A3) For some $\alpha_0, \alpha_1 > 0$ and for all m, n, $\mathbb{E}\left|\xi_m^\pm\right|^2 \leq \alpha_0$, $\mathbb{E}\left|f(x_n \pm \delta d_m)\right|^2 \leq \alpha_1$ for any $\delta > 0$ and d_m, $m = 0, \ldots, P$.

---

[1]Here $\nabla^3 f(x) = \dfrac{\partial^3 f(x)}{\partial x^\mathsf{T} \partial x^\mathsf{T} \partial x^\mathsf{T}}$ denotes the third derivative of f at x and $\nabla_{i_1 i_2 i_3}^3 f(x)$ denotes the $(i_1 i_2 i_3)$th entry of $\nabla^3 f(x)$, for $i_1, i_2, i_3 = 1, \ldots, N$.

(A4) The step-sizes a_n and perturbation constants δ_n are positive, for all n and satisfy

$$a_n, \delta_n \to 0 \text{ as } n \to \infty, \sum_n a_n = \infty \text{ and } \sum_n \left(\frac{a_n}{\delta_n}\right)^2 < \infty.$$

(A5) $\sup_n \|x_n\| < \infty$ w.p. 1.

The assumptions above are common to the analysis of simultaneous perturbation methods, and can be found, for instance, in the context of 1SPSA [26] – see also [4] for the analysis of other simultaneous perturbation schemes that employ similar assumptions. The first two are necessary to establish asymptotic unbiasedness of the 1RDSA-DP gradient estimate through a Taylor series expansion facilitated by (A1), while ignoring the noise owing to (A2). The third and fourth assumptions are necessary to ignore the effects of noise on the convergence behavior of x_n. The final assumption requiring boundedness of the iterates x_n can be ensured by constraining the iterates x_n to evolve in a certain compact region and projecting them back each time they go out of the region, see [18] (Chapter 5). If the projected region contains the optima, then the stochastic gradient algorithms (RDSA or SPSA) would converge to this point, and in the complementary case, the algorithm would get stuck on the boundary of the projection region. In the literature, there also exist approaches to overcome the latter case, by either growing the projection region [6], or performing sparse projections [9] (i.e., at time instants that grow exponentially to infinity, while not projecting the iterates at the remaining time instants).

Lemma 2. *(Asymptotic unbiasedness of 1RDSA-DP gradient estimate) Under (A1)-(A5),*

(i) for $\widehat{\nabla} f(x_n)$ defined according to (5), we have a.s. that[2]

$$\left| \mathbb{E}\left[\widehat{\nabla}_i f(x_n) \middle| \mathcal{F}_n \right] - \nabla_i f(x_n) \right| = C_0 \delta_{n3^N}^2,$$

for $i = 1, \ldots, N$, where $C_0 = \alpha_0 N^3 3^{N-1}$ and $\mathcal{F}_n = \sigma(x_k, k \le n)$, $n \ge 1$.

(ii) for $\widehat{\nabla} f(x_n)$ defined according to (7), we have a.s. that

$$\left| \mathbb{E}\left[\widehat{\nabla}_i f(x_n) \middle| \mathcal{F}_n \right] - \nabla_i f(x_n) \right| = C_0 \delta_{nN}^2, \tag{8}$$

for $i = 1, \ldots, N$, where $C_0 = \alpha_0 N^3/6$.

Proof. See Section 4.1.2. □

The advantage of the permutation matrix approach is that the dependence on the dimension N is linear, whereas the semi-lexicographic sequence has an exponential dependence on N.

We now have an asymptotic convergence claim for x_n updated according to (6); the claim is verbatim from Theorem 2 of [19].

Theorem 3. *(Strong Convergence) Let x^* be an asymptotically stable equilibrium of the following ordinary differential equation (ODE): $\dot{x}_t = -\nabla f(x_t)$, with domain of attraction $D(x^*)$, i.e., $D(x^*) = \{x_0 \mid \lim_{t\to\infty} x(t \mid x_0) = x^*\}$, where $x(t \mid x_0)$ is the solution to the ODE with initial condition x_0. Assume (A1)-(A5), and also that there exists a compact subset $\mathcal{D}$ of $D(x^*)$ such that $x_n \in \mathcal{D}$ infinitely often. Let x_n be governed by (6), with the gradient estimate $\widehat{\nabla} f(x_n)$ defined either according to (5) or (7). Then,*

$$x_n \to x^* \text{ a.s. as } n \to \infty.$$

---

[2]Here $\widehat{\nabla}_i f(x_n)$ and $\nabla_i f(x_n)$ denote the ith coordinates in the gradient estimate $\widehat{\nabla} f(x_n)$ and true gradient $\nabla f(x_n)$, respectively.

Proof. See Section 4.1.3. □

For the special case when the objective f is strongly-convex, we present a non-asymptotic bound for 1RDSA-DP with permutation matrix-based perturbations. More precisely, we assume the objective function f satisfies the following assumption:
(A1') For any x, x', we have

$$(\nabla f(x) - \nabla f(x'))^\top (x - x') \geq \mu \left\| x - x' \right\|_2^2,$$

for some $\mu > 0$.

Theorem 4. *Under (A1') and (A2)-(A5), we have,*

$$\mathbb{E} \left\| x_{n+1} - x^* \right\|_2 \leq \underbrace{\sqrt{2} \exp(-\mu\Gamma_n) \left\| x_0 - x^* \right\|_2}_{\textit{initial error}}$$

$$+ \left(\underbrace{3 \sum_{k=1}^n a_k^2 \exp(-2\mu(\Gamma_n - \Gamma_k)) C_0^2 \delta_k^4}_{\textit{bias error}} + \underbrace{2 \sum_{k=1}^n a_k^2 \exp(-2\mu(\Gamma_n - \Gamma_k)) C_1 \delta_k^{-2}}_{\textit{sampling error}} \right)^{\frac{1}{2}}, \quad (9)$$

where x^* is the global minimizer of f, $\Gamma_k := \sum_{i=1}^k a_i$, C_0 is as defined in Lemma 2, and $C_1 = \alpha_1 N/2$.

Proof. See Section 4.1.4. □

The initial error depends on the starting point x_0 of the algorithm. The sampling error relates to a martingale difference sequence, which arises due to the fact that only noisy measurements of the objective function are available. The bias error arises out of the need to estimate gradients from function measurements and quantifies the error in gradient estimation. The initial and sampling error components are common to classic stochastic convex optimization settings, while the bias error is specific to the simulation optimization framework, i.e., a setting where gradients are not directly available and have to be estimated from noisy function measurements.

Now we specialize the result above by choosing the step-size a_n and perturbation constant δ_n to obtain an order $O\left(\frac{1}{\sqrt{n}}\right)$ bound in expectation on the optimization error of the algorithm.

Theorem 5. *Let $a_k = c/k$ and $\delta_k = \delta_0/k^\delta$. Then under (A1'), (A2) and (A3),*

$$\mathbb{E} \left\| x_n - x^* \right\|_2 \leq \frac{\sqrt{2} \left\| x_0 - x^* \right\|_2}{n^{\mu c}} + \frac{\sqrt{3} c C_0 \delta_0^2}{\sqrt{2\mu c - 4\delta - 1}} n^{-\frac{(1+4\delta)}{2}} + \frac{\sqrt{3 C_1} c}{\delta_0 \sqrt{2\mu c + 2\delta - 1}} n^{\delta - \frac{1}{2}}.$$

Proof. See Section 4.2. □

Choosing $\delta = 0$, one can recover the optimal rate of the order $O\left(n^{-1/2}\right)$ for simultaneous perturbation schemes. Further, choosing c such that $\mu c > 1/2$, it is easy to observe that the initial error is forgotten faster than the other error components. In contrast, for the more general case of non-convex objective f, the authors in [26, 7] are able to establish a rate of $O\left(n^{-1/3}\right)$ obtained from an asymptotic mean square error analysis using the second moment of the limiting normal distribution. More recently, for the case of convex (and not necessarily strongly-convex) objective f, an error of the order $O\left(n^{-1/3}\right)$ is unavoidable from an information-theoretic (or minimax) viewpoint – see [14] for further details.

3 Second-order RDSA with deterministic perturbations (2RDSA-DP)

Second-order methods provide many advantages over their first-order counterparts. The main benefit of second-order methods over first-order methods is that they converge at the optimum rate without requiring knowledge of minimum eigenvalue of $\nabla^2 f(x^*)$. Other benefits include (i) faster convergence in the final phase, i.e., when the iterate is close to the optima as second-order methods minimize a quadratic model of f and (ii) scale-invariance, i.e., second-order methods adjust automatically to the scale of the parameter and hence, the update rule is unaffected. On the flip side, second-order schemes require estimating the Hessian in addition to the gradient of f and have a higher per-iteration cost due to matrix inversion.

Using the two deterministic sequences, i.e., semi-lexicographic and permutation matrix-based choices, presented in the previous section, we provide two variants of the 2RDSA algorithm proposed in [19].

3.1 Semi-lexicographic sequence-based perturbations

The reason such deterministic choices for perturbations work in the context of 2RDSA can be seen as follows: Using suitable Taylor's series expansions (see Lemma 6 below), we have

$$\mathbb{E}[\widehat{H}_n \mid \mathcal{F}_n] = \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} \left(M_m \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) + O(\delta_n^2) \right). \quad (10)$$

From Lemma 1, it can be seen that

$$\frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} \sum_{i=1}^{N} (d_m^i)^2 = 1, \text{ and } \sum_{m=0}^{3^N-1} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j = 0.$$

The above equality can be directly verified for the case when $N = 2$ and $N = 3$ using Table 1.

Plugging the fact above into (10) followed by a tedious calculation (see Lemma 6 below), we obtain

$$\mathbb{E}[\widehat{H}_n(i,j) \mid \mathcal{F}_n] = \nabla_{ij}^2 f(x_n) + O(\delta_{n3^N}^2).$$

Remark 2. *(**Jacobi variant**) If the Hessian is known to be in a diagonal form, i.e., if the requirement is for an algorithm to estimate $\nabla_{ii}^2 f(\cdot)$, then the estimate of Algorithm 2 can be replaced by the following:*

$$\widehat{H}_n = \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} \tilde{H}_m,$$

with the inner-loop Hessian estimate given by

$$\tilde{H}_m = \left[\frac{y_m^+ + y_m^- - 2y_n}{\delta_{n3^N+m}^2} \right].$$

Notice that, unlike Algorithm 2, the scheme above (the so-called Jacobi variant of stochastic Newton algorithms - cf. [1]) can estimate the diagonal entries of the Hessian, and, more importantly, cannot estimate the off-diagonal entries of the Hessian, as the off-diagonal perturbation terms of interest zero out over the inner loop, i.e., $\sum_{m=0}^{3^N-1} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j = 0$.

Algorithm 2 2RDSA-Lex-DP

Input: initial parameter $x_0 \in \mathbb{R}^N$, perturbation constants $\delta_n > 0$, step-sizes $\{a_n, b_n\}$, matrix projection operator Υ. The deterministic perturbation $\{d_m\}$ sequence is chosen in the same manner as in 1RDSA-DP.

for $n = 0, 1, 2, \ldots$ **do**

    Obtain function value $y_n = f(x_n) + \xi$, where ξ is the measure noise.

        $\triangleright$ As in 1RDSA, fix x_n and loop through the rows of matrix D_N for perturbations d_m.

    for $m = 0, 1, 2, \ldots, 3^N - 1$ **do**

        Obtain function values $y_m^+ = f(x_n + \delta_{n3^N+m} d_m) + \xi$ and $y_m^- = f(x_n - \delta_{n3^N+m} d_m) + \xi$, where ξ is the measure noise.

$$\text{Set} \quad g_m = d_m \left[\frac{y_m^+ - y_m^-}{2\delta_{n3^N+m}} \right], \tag{11}$$

$$\text{Set} \quad \tilde{H}_m = M_m \left[\frac{y_m^+ + y_m^- - 2y_n}{\delta_{n3^N+m}^2} \right], \quad \text{where} \tag{12}$$

$$M_m = \begin{bmatrix} \kappa \left((d_m^1)^2 - 2 \times 3^N \right) & \cdots & d_m^1 d_m^N \\ d_m^2 d_m^1 & \cdots & d_m^2 d_m^N \\ \vdots & \vdots & \vdots \\ d_m^N d_m^1 & \cdots & \kappa \left((d_m^N)^2 - 2 \times 3^N \right) \end{bmatrix},$$

$$\text{and } \kappa = \left(\frac{1}{2 \times 3^{N-1}} - 1 \right)^{-1}.$$

    end for

$$\text{Gradient estimate:} \quad \widehat{\nabla} f(x_n) = \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} g_m. \tag{13}$$

$$\text{Hessian estimate:} \quad \widehat{H}_n = \frac{1}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} \tilde{H}_m. \tag{14}$$

$$\text{Hessian update:} \quad \overline{H}_n = (1 - b_n)\overline{H}_{n-1} + b_n \widehat{H}_n. \tag{15}$$

$$\text{Parameter update:} \quad x_{n+1} = x_n - a_n \Upsilon(\overline{H}_n)^{-1} \widehat{\nabla} f(x_n). \tag{16}$$

end for
Return x_n.

3.2 Permutation matrix-based perturbations

In the case of permutation matrix-based deterministic perturbations, it is not possible to estimate the off-diagonal entries of the Hessian. This is because $\sum_{m=0}^{N-1} d_m^i d_m^j = 0$, $i \neq j$ for permutation matrix-based perturbations. While a similar property holds for semi-lexicographic perturbations as well, we could add correction factors through the M_m matrix (see Algorithm 2) to produce an estimate for all the entries in the Hessian matrix. A similar correction factor is not feasible for the case of permutation matrix-based perturbations, because each column of a permutation matrix contains only one positive ($= 1$) entry, while the rest are zero. In other words, in (10), the second term inside the brackets always sums to zero when the outside summation for m goes up to $N - 1$, irrespective of the choice for M_m.

However, using a permutation matrix, it is possible to estimate the diagonal entries of the Hessian. In this case, the overall algorithm follows the template provided in Algorithm 2, except that the perturbations are generated using a permutation matrix in N-dimensions, the inner loop for m would run from 0 to $N - 1$, and the gradient/Hessian estimates in Algorithm 2 are replaced by

$$\text{Gradient estimate:} \quad \widehat{\nabla} f(x_n) = \sum_{m=0}^{N-1} g_m, \tag{17}$$

$$\text{Hessian estimate:} \quad \widehat{H}_n = \sum_{m=0}^{N-1} \tilde{H}_m, \tag{18}$$

where

$$\tilde{H}_m = \left[\frac{y_m^+ + y_m^- - 2y_n}{\delta_{nN+m}^2} \right].$$

Alternative using two permutation matrices

Let D_N and $\hat{D}_N$ be two N-dimensional permutation matrices that are not identical. Let $y_m^+ = f(x_n + \delta_n d_m + \delta_n \hat{d}_m) + \xi$ and $y_m^- = f(x_n - \delta_n d_m - \delta_n \hat{d}_m) + \xi$ be function measurements, where ξ is the measure noise, d_m and $\hat{d}_m$ are sourced from D_N and $\hat{D}_N$, respectively. In other words, d_m and $\hat{d}_m$ would loop through the rows of D_N and $\hat{D}_N$, respectively. Consider the following estimate for the Hessian, in place of (14):

$$\text{Hessian estimate:} \quad \widehat{H}_n = \sum_{m=0}^{N-1} \tilde{H}_m, \tag{19}$$

where

$$\tilde{H}_m = \left[\frac{y_m^+ + y_m^- - 2y_n}{\delta_{nN+m}^2} \right].$$

3.3 Main results

The analysis of 2RDSA-DP is under assumptions that match those employed for studying the convergence behavior of regular second-order SPSA and RDSA algorithms (i.e., with random perturbations), and we list them below for the sake of completeness.

(C1) The function f is four-times differentiable[3] with $\left|\nabla^4_{i_1 i_2 i_3 i_4} f(x_n)\right| < \infty$, for $i_1, i_2, i_3, i_4 = 1, \ldots, N$ and for all n.

(C2) For each n and all x, there exists a $\rho > 0$ not dependent on n and x, such that $(x - x^*)^{\mathsf{T}} \bar{f}_n(x) \geq \rho \|x_n - x\|$, where $\bar{f}_n(x) = \Upsilon(\overline{H}_n)^{-1} \nabla f(x)$.

(C3) $\{\xi_m, \xi_m^+, \xi_m^-, m = 0, \ldots, P, n = 1, 2, \ldots\}$ satisfy $\mathbb{E}\left[\xi_m^+ + \xi_m^- - 2\xi_m \mid \mathcal{F}_n\right] = 0$, where $P = 3^N - 1$ for semi-lexicographic 2RDSA-DP and $P = N - 1$ for permutation matrix-based 2RDSA-DP.

(C4) Same as (A4).

(C5) For each $i = 1, \ldots, N$ and any $\rho > 0$, $P(\{\bar{f}_{ni}(x_n) \geq 0 \text{ i.o}\} \cap \{\bar{f}_{ni}(x_n) < 0 \text{ i.o}\} \mid \{|x_{ni} - x_i^*| \geq \rho \quad \forall n\}) = 0$.

(C6) The operator Υ satisfies $\delta_n^2 \Upsilon(H_n)^{-1} \to 0$ a.s. and $E(\|\Upsilon(H_n)^{-1}\|^{2+\zeta}) \leq \rho$ for some $\zeta, \rho > 0$.

(C7) For any $\varsigma > 0$ and nonempty $S \subseteq \{1, \ldots, N\}$, there exists a $\rho'(\varsigma, S) > \varsigma$ such that

$$\limsup_{n \to \infty} \left| \frac{\sum_{i \notin S}(x - x^*)_i \bar{f}_{ni}(x)}{\sum_{i \in S}(x - x^*)_i \bar{f}_{ni}(x)} \right| < 1 \text{ a.s.}$$

for all $|(x - x^*)_i| < \varsigma$ when $i \notin S$ and $|(x - x^*)_i| \geq \rho'(\varsigma, S)$ when $i \in S$.

(C8) For some $\alpha_0, \alpha_1 > 0$ and for all m, n, $\mathbb{E}|\xi_m|^2 \leq \alpha_0$, $\mathbb{E}|\xi_m^\pm|^2 \leq \alpha_0$, $\mathbb{E}|f(x_n)|^2 \leq \alpha_1$ and $\mathbb{E}|f(x_n \pm \delta d_m)|^2 \leq \alpha_1$, for any $\delta > 0$.

(C9) $\sum_n \frac{1}{(n+1)^2 \delta_n^4} < \infty$.

Comments on assumptions (C1)-(C9): (C1) and (C2) are basic assumptions about the smoothness and steepness of the function f. (C1) holds if f twice continuously differentiable with a bounded second derivative on $\mathbb{R}^N$ and (C2) ensures the function f has enough curvature. (C3) and (C4) are common martingale-difference noise and step-sizes conditions and can be motivated in a similar manner as in the case of 1RDSA-DP (see Section II-C). (C5) says that if x_n is uniformly bounded away from x^*, then x_n cannot be bouncing around causing the change in signs of the normalized gradient elements an infinite number of times. (C6) can be ensured by having $\Upsilon(A)$ defined as performing an eigen-decomposition of A followed by projecting the eigenvalues to the positive side by adding a large enough scalar. (C7) ensures that, after sufficiently large iterations, each element of $\bar{f}_n(x)$ tends to make a non-negligible contribution to products of the form $(x - x^*)^T \bar{f}_n(x)$ (see C2). (C5) and (C7) are not necessary if the iterates are bounded, i.e., $\sup_n \|x_n\| < \infty$ a.s. Finally, (C8) and (C9) are necessary to ensure convergence of the Hessian recursion, in particular, to invoke a martingale convergence result (see Theorem 7 and the proof of [24, Theorem 2a]). For a more detailed interpretation of the above conditions, the reader is referred to Section III and Appendix B of [24].

The main claim that establishes the asymptotic unbiasedness of the Hessian estimate in the DP variants of 2RDSA that we propose is given below.

Lemma 6. *(Asymptotic unbiasedness of 2RDSA-DP Hessian estimate) Under (C1)-(C9),*

---

[3]Here $\nabla^4 f(x) = \dfrac{\partial^4 f(x)}{\partial x^{\mathsf{T}} \partial x^{\mathsf{T}} \partial x^{\mathsf{T}} \partial x^{\mathsf{T}}}$ denotes the fourth derivative of f at x and $\nabla^4_{i_1 i_2 i_3 i_4} f(x)$ denotes the $(i_1 i_2 i_3 i_4)$th entry of $\nabla^4 f(x)$, for $i_1, i_2, i_3, i_4 = 1, \ldots, N$.

(i) for $\widehat{H}_n$ defined according to (14), we have a.s. that[4], for $i, j = 1, \ldots, N$,

$$\left| \mathbb{E}\left[\left.\widehat{H}_n(i,j)\right| \mathcal{F}_n\right] - \nabla_{ij}^2 f(x_n) \right| = O(\delta_{n3N}^2). \tag{20}$$

(ii) for $\widehat{H}_n$ defined according to (18), we have a.s. that

$$\left| \mathbb{E}\left[\left.\widehat{H}_n(i,i)\right| \mathcal{F}_n\right] - \nabla_{ii}^2 f(x_n) \right| = O(\delta_{nN}^2), \tag{21}$$

for $i = 1, \ldots, N$.

Proof. See Section 4.3.1. □

Once we have the asymptotic unbiasedness for the 2RDSA-DP Hessian estimate, the convergence of the Hessian recursion is immediate and is given below for the sake of completeness.

Theorem 7. *(**Strong Convergence**) Assume (C1)-(C9). Then $x_n \to x^*$ a.s. as $n \to \infty$, where x_n is given by (16). For 2RDSA-Lex-DP $\overline{H}_n \to \nabla^2 f(x^*)$ a.s. as $n \to \infty$, furthermore, if the true Hessian is diagonal, then even for 2RDSA-Perm-DP $\overline{H}_n \to \nabla^2 f(x^*)$ a.s. as $n \to \infty$, where $\overline{H}_n$ is governed by (15).*

Proof. See Section 4.3.4. □

We next present a convergence rate result for the special case of a quadratic objective function under the following additional assumptions, which have been used to establish a rate result for a variant of 2SPSA in [25]:

(C10) f is quadratic and $\nabla^2 f(x^*) > 0$.

(C11) The operator Υ is chosen such that $\mathbb{E} \parallel \Upsilon(\overline{H}_n) - \overline{H}_n \parallel^2 = o(e^{-2b_0 n^{1-r}/(1-r)})$ and $\parallel \Upsilon(H) - H \parallel^2 / \left(1 + \parallel H \parallel^2\right)$ is uniformly bounded.

The assumptions (C10) and (C11) are much stronger compared to (C1) and (C6), respectively. In a noise-free setting, after suitable Taylor series expansions, the Hessian estimate of 2SPSA can be written as

$$\widehat{H}_n(2SPSA) = \nabla^2 H(x_n) + \Psi(H(x_n)),$$

where $H(x_n)$ is the Hessian at iterate x_n and $\Psi((H(x_n))$ is a function that involves the Hessian at x_n and random perturbations used in 2SPSA. More importantly, $\Psi(H(x_n))$ is zero-mean. In [25], since the true Hessian $H(x_n)$ is not known in practice, in place of $\widehat{H}_n$ in the Hessian update in (15), the author uses the following improved Hessian estimate: $\widehat{H}_n(2SPSA) - \Psi(\overline{H}_n)$. The rationale underlying this replacement is that subtracting the $\Psi(\overline{H}_n)$ term reduces the error in Hessian estimation. In [20], the authors use a similar trick to reduce the Hessian estimation error in regular 2RDSA. We claim such a feedback term is not necessary in the context of 2RDSA-DP and this can be argued as follows: From the proof passage leading to (37), it is easy to infer the following after ignoring the noise terms:

$$\widehat{H}_n(i,j) = \nabla_{ij}^2 f(x_n) + O(\delta_{n3N}^2).$$

The principal difference with feedback variants of 2SPSA/2RDSA is that the equation above implies that there are no zero-mean terms involving the perturbations that one needs to worry about in case of 2RDSA-DP.

To substantiate the claim that the feedback terms are not necessary for 2RDSA-DP, we provide a rate result that parallels a corresponding result for 2SPSA with feedback [25]. The proof given later is also much simpler than the corresponding version for 2SPSA, due to the fact that there are no extra terms involving perturbations to handle.

---

[4]Here $\widehat{H}_n(i,j)$ and $\nabla_{ij}^2 f(\cdot)$ denote the (i,j)th entry in the Hessian estimate $\widehat{H}_n$ and the true Hessian $\nabla^2 f(\cdot)$, respectively.

Theorem 8. *Assume (C9), (C10) and (C11), and also that the setting is noise-free. Let $b_n = b_0/n^r$, $n = 1, 2, \ldots, k$, where $1/2 < r < 1$ and $0 < b_0 \leq 1$. Let $H^* = \nabla^2 f(x)$ for any x, and $\Lambda_k = \overline{H}_k - H^*$. Then, we have*

$$trace[\mathbb{E}(\Lambda_n^\intercal \Lambda_n)] = O(e^{-2b_0 n^{1-r}/(1-r)}). \tag{22}$$

Proof. See Section 4.3.5. □

4 Convergence proofs

4.1 Proofs for 1RDSA variants with deterministic perturbations

4.1.1 Proof of Lemma 1

Proof.

Case 1: Semi-lexicographic sequence-based perturbations

We prove the claim by induction. Consider first the case of $N = 1$. Now $D_1^1 = \begin{bmatrix} -1 \\ -1 \\ 2 \end{bmatrix}$. Note that

$M_1 = (D_1^1)^\intercal D_1^1 = 6$. Thus the result holds for $N = 1$.

Suppose that the claim above holds for $N = k$, i.e., $M_k = 2 \times 3^k \mathcal{I}_k$. Consider the case of $N = k + 1$. We make the following observations:

1. The size of each column D_{k+1}^i, $i = 1, \ldots, k+1$ is 3^{k+1}.

2. The columns $D_{k+1}^2, \ldots, D_{k+1}^{k+1}$ are obtained from $D_k^1, \ldots, D_k^k$, respectively by concatenating the D_k^i columns thrice, i.e.,

$$D_{k+1}^i = \begin{bmatrix} D_k^{i-1} \\ D_k^{i-1} \\ D_k^{i-1} \end{bmatrix}, \quad i = 2, \ldots, k+1. \tag{23}$$

3. In the first column, i.e., D_{k+1}^1, the first 2×3^k elements are -1 and the remaining 3^k elements are 2.

4. The off-diagonal terms in M_{k+1} are all zero. This is argued as follows: For $i \neq j$, $i, j \in \{2, \ldots, k+1\}$, we have that $(D_{k+1}^i)^\intercal D_{k+1}^j = 0$. This follows from (23) because $(D_{k+1}^i)^\intercal D_{k+1}^j = 3(D_k^{i-1})^\intercal D_k^{j-1} = 0$ (by induction hypothesis).

   For off-diagonal terms of the type $(D_{k+1}^1)^\intercal D_{k+1}^i$, where $i \in \{2, \ldots, k+1\}$, we first re-write D_{k+1}^1 and D_{k+1}^i as follows:

$$D_{k+1}^1 = \begin{bmatrix} -\mathbf{1} \\ -\mathbf{1} \\ \mathbf{2} \end{bmatrix}, \tag{24}$$

   where $-\mathbf{1}$ and $\mathbf{2}$ are $k \times 1$ vectors with each entry -1 and 2, respectively. Thus, from (23) and the foregoing,

$$(D_{k+1}^1)^\intercal D_{k+1}^i = -\mathbf{1}^\intercal D_k^{i-1} - \mathbf{1}^\intercal D_k^{i-1} + \mathbf{2}^\intercal D_k^{i-1} = 0.$$

5. Finally, the diagonal terms of M_{k+1} are $2 \times 3^{k+1}$. This can be seen as follows: For $i = 1, \ldots, k+1$,

$$(D_{k+1}^i)^\mathsf{T} D_{k+1}^i = \frac{2}{3} \times (-1)^2 \times 3^{k+1} + \frac{1}{3} \times (2)^2 \times 3^{k+1}$$
$$= 2 \times 3^{k+1}.$$

In the above, we have used the fact that D_{k+1}^i is a 3^{k+1}-dimensional vector with two-third entries -1 and the remaining one-third entries as 2.

Thus, $M_{k+1} = 2 \times 3^{k+1} \mathcal{I}_{k+1}$ and the claim follows for semi-lexicographic perturbations.

Case 2: Permutation matrix-based perturbations

Let D_N, the columns D_N^i, be the N-dimensional permutation matrix. Recall that 1RDSA-Perm-DP loops through the rows in D_N and the columns D_N^i in D_N are in N-dimensions. It is well-known that $D_N^{-1} = D_N^\mathsf{T}$ for a permutation matrix D_N, cf. Proposition 2.7.21 of [13] for a detailed proof. Hence, the claim that $M_N = \mathcal{I}_N$ follows for the case of permutation matrix based deterministic perturbation sequences. $\qquad\square$

4.1.2 Proof of Lemma 2

Proof. We prove the claim for semi-lexicographic perturbations; the claim for the case of permutation matrix-based perturbations follows by an analogous argument.

By Taylor's series expansions, we obtain, a.s.,

$$f(x_n \pm \delta_n d_m) = f(x_n) \pm \delta_n d_m^T \nabla f(x_n) + \frac{\delta_n^2}{2} d_m^T \nabla^2 f(x_n) d_m \pm \frac{\delta_n^3}{6} \nabla^3 f(\tilde{x}_n^+)(d_m \otimes d_m \otimes d_m)$$

where $\otimes$ denotes the Kronecker product and $\tilde{x}_n^+$ (respectively, $\tilde{x}_n^-$) are on the line segment between x_n and $(x_n + \delta_n d_m)$ (respectively, $(x_n - \delta_n d_m)$). Then

$$\mathbb{E}\left[\widehat{\nabla} f(x_n) \middle| \mathcal{F}_n\right]$$

$$= \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} \mathbb{E}\left[g_m \middle| \mathcal{F}_n\right]$$

$$= \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} \mathbb{E}\left[\frac{d_m}{2\delta_{(n3^N+m)}} \times \left(f(x_n + \delta_{(n3^N+m)} d_m) - f(x_n - \delta_{(n3^N+m)} d_m)\right) \middle| \mathcal{F}_n\right]$$

$$= \frac{1}{2 \times 3^N} \sum_{m=0}^{3^N-1} \left[d_m d_m^\mathsf{T} \nabla f(x_n) + \frac{\delta_{(n3^N+m)}^2}{12} d_m (\nabla^3 f(\tilde{x}_n^+) + \nabla^3 f(\tilde{x}_n^-))(d_m \otimes d_m \otimes d_m)\right]$$

$$= \nabla f(x_n) + C_0 \delta_{n3^N}^2. \tag{25}$$

The first term on the RHS of (25) follows from Lemma 1. Now, the lth coordinate of the second term in the RHS of (25) can be upper-bounded as follows:

$$\sum_{m=0}^{3^N-1} \frac{\delta_{(n3^N+m)}^2}{12} d_m (\nabla^3 f(\tilde{x}_n^+) + \nabla^3 f(\tilde{x}_n^-))(d_m \otimes d_m \otimes d_m)$$

$$\leq \frac{\alpha_0 \delta_{n3^N}^2}{6} \sum_{m=0}^{3^N-1} \sum_{i_1=1}^{N} \sum_{i_2=1}^{N} \sum_{i_3=1}^{N} (d_m^l d_m^{i_1} d_m^{i_2} d_m^{i_3})$$

16

$$\leq \frac{\alpha_0 \delta_{n3N}^2 N^3 (2 \times 3^N)^2}{6}$$

The first inequality above follows from (A1), while the second inequality follows from the fact that $\sum_{m=0}^{3^N-1} (d_m^l d_m^{i_1} d_m^{i_2} d_m^{i_3})$ is non-zero only if either $l = i_2$ and $i_1 = i_3$ or vice-versa and in this case, we have

$$\sum_{m=0}^{3^N-1} (d_m^l)^2 (d_m^{i_1})^2 = (2 \times 3^N)^2.$$

The equality above can be easily inferred using induction arguments similar to proof of Lemma 1 and we omit the details. $\qquad\square$

4.1.3 Proof of Theorem 3

Proof. Follows in exactly the same manner as the proof of Theorem 2 of [19], given the asymptotic unbiasedness result in Lemma 2. $\qquad\square$

4.1.4 Proof of Theorem 4

Proof. Fixing δ_n for the inner loop and using arguments from the proof of Lemma 2, we obtain

$$\widehat{\nabla} f(x_n) = \nabla f(x_n) + C_0 \delta_n^2 + \sum_{m=0}^{N-1} d_m \left(\frac{\xi_m^+ - \xi_m^-}{2\delta_n} \right).$$

Letting $\eta_n = \sum_{m=0}^{N-1} d_m \left(\frac{\xi_m^+ - \xi_m^-}{2\delta_n} \right)$, the update in (6) is equivalent to

$$x_{n+1} = x_n - a_n \left[\nabla f(x_n) + C_0 \delta_n^2 + \eta_n \right]. \tag{26}$$

Since $\nabla f(x^*) = 0$, we have the following from the fundamental theorem of calculus:

$$\left(\int_0^1 \nabla^2 f(x^* + \lambda(x_n - x^*)) d\lambda \right) z_n = \nabla f(x_n).$$

Here $z_n \triangleq x_n - x^*$ denotes the optimization error at instant n of 1RDSA-DP. Then, using (26), we have the following recursive update form for z_n:

$$\begin{aligned} z_{n+1} &= (I - a_n J_n) z_n + a_n \left(C_0 \delta_n^2 + \eta_n \right) \\ &= \Pi_n z_0 + \sum_{k=1}^n a_k \Pi_n \Pi_k^{-1} (C_0 \delta_k^2 + \eta_k), \end{aligned} \tag{27}$$

where $J_n := \int_0^1 \nabla^2 f(x^* + \lambda(x_n - x^*)) d\lambda$ and $\Pi_n := \prod_{k=1}^n (I - a_k J_k)$. A similar unrolling of a general stochastic approximation recursion can be found in [11]. However, our setting involves biased gradient estimates and the non-asymptotic bounds require a careful handling of the perturbation constant δ_n, so that the overall convergence rate is of the order $O(1/\sqrt{n})$. Moreover, we make all the constants explicit in the final bound.

Now, for the square of the error $\|z_{n+1}\|_2^2$, we use (27) and Jensen's inequality to obtain

$$(\mathbb{E} \|z_{n+1}\|_2)^2 \leq \mathbb{E}(\langle z_n, z_n \rangle)$$

$$= \mathbb{E}\Bigg[\|\Pi_n z_0\|_2^2 + \Bigg\|\sum_{k=1}^{n} a_k \Pi_n \Pi_k^{-1} C_0 \delta_k^2 \Bigg\|_2^2 + \Bigg\|\sum_{k=1}^{n} a_k \Pi_n \Pi_k^{-1} \eta_k \Bigg\|_2^2 + \Bigg\langle \Pi_n z_0, \sum_{k=1}^{n} a_k \Pi_n \Pi_k^{-1} C_0 \delta_k^2 \Bigg\rangle$$

$$+ \Bigg\langle \Pi_n z_0, \sum_{k=1}^{n} a_k \Pi_n \Pi_k^{-1} \eta_k \Bigg\rangle + \Bigg\langle \sum_{k=1}^{n} a_k \Pi_n \Pi_k^{-1} C_0 \delta_k^2, \sum_{k=1}^{n} a_k \Pi_n \Pi_k^{-1} \eta_k \Bigg\rangle \Bigg]$$

$$\leq 2\|\Pi_n z_0\|_2^2 + 3\sum_{k=1}^{n} a_k^2 \|\Pi_n \Pi_k^{-1}\|_2^2 C_0^2 \delta_k^4 + 2\sum_{k=1}^{n} a_k^2 \|\Pi_n \Pi_k^{-1}\|_2^2 \mathbb{E}\|\eta_k\|_2^2. \tag{28}$$

For the last inequality, we have used the fact that η_k is zero-mean (see (A2)) in order to ignore a cross term. For the other two cross terms, we have used Cauchy-Schwarz to conclude $\langle a, b \rangle \leq \max(\|a\|_2^2, \|b\|_2^2)$ and hence, the first and last square terms can appear at most twice, while the second square term can appear at most thrice.

The second moment of the noise factor η_n can be bounded as follows:

$$\mathbb{E}\|\eta_n\|_2^2 \leq \sum_{m=0}^{N-1} \|d_m\|_2^2 \, \mathbb{E}\left(\frac{\xi_m^+ - \xi_m^-}{2\delta_n}\right)^2 \leq \frac{N\alpha_1}{2\delta_n^2}, \tag{29}$$

where we have used the fact that for permutation matrix-based perturbation $\|d_m\|_2 = 1$ and $\mathbb{E}(\xi_m^+ - \xi_m^-)^2 \leq 2\alpha_1$ from assumption (A3).

The term $\|\Pi_n \Pi_k^{-1}\|_2$ is bounded as follows:

$$\|\Pi_n \Pi_k^{-1}\|_2 = \Bigg\| \prod_{j=k+1}^{n} (I - a_j J_j) \Bigg\|_2$$

$$= \prod_{j=k+1}^{n} \|(1 - a_j\mu)I - a_j(J_j - \mu I)\|_2$$

$$\leq \prod_{j=k+1}^{n} \|(1 - a_j\mu)I\|_2 \leq \prod_{j=k+1}^{n} (1 - a_j\mu)$$

$$\leq \exp\left(-\mu(\Gamma_n - \Gamma_k)\right). \tag{30}$$

The second inequality above follows by observing that $\|I - a_n J_n\|_2 \leq \exp(-\mu a_n)$, since $\nabla^2 f(x) - \mu I$ is positive semi-definite owing to strong-convexity of f (see (A1')).

The main claim now follows by plugging in the bounds in (29) and (30) into (28). $\qquad\square$

4.2 Proof of Theorem 5

Proof. We bound each of the error terms on the RHS of (9) separately. For bounding the initial error, we use the following inequality:

$$\exp(-\mu\Gamma_n) \leq \exp(-\mu c \ln n) \leq n^{-\mu c}.$$

In arriving at the bound above, we have compared a sum with an integral.

Substituting $a_k = c/k$ and $\delta_k = \delta_0/k^\delta$ into the bias error term in (9), we obtain

$$\sum_{k=1}^{n} a_k^2 \exp(-2\mu(\Gamma_n - \Gamma_k)) C_0^2 \delta_k^4 \leq \sum_{k=1}^{n} \frac{c^2}{k^2} n^{-2\mu c} k^{2\mu c} C_0^2 \frac{\delta_0^4}{k^{4\delta}}$$

18

$$\leq c^2 n^{-2\mu c} C_0^2 \delta_0^4 \sum_{k=1}^{n} k^{2\mu c - 4\delta - 2}$$

$$\leq \frac{c^2 C_0^2 \delta_0^4}{(2\mu c - 4\delta - 1)} n^{-1-4\delta}.$$

Along similar lines, the sampling error term in (9) can be upper-bounded as follows:

$$\sum_{k=1}^{n} a_k^2 \exp(-2\mu(\Gamma_n - \Gamma_k)) \frac{C_1}{\delta_k^2} \leq \frac{c^2 C_1}{\delta_0^2 (2\mu c - 4\delta - 1)} n^{-1+2\delta}.$$

The claim follows by combining the bounds derived above on each of the error terms on the RHS of (9). $\quad\square$

4.3 Proofs for 2RDSA variants with deterministic perturbations

4.3.1 Proof of Lemma 6

Proof.
Case 1: Semi-lexicographic sequence-based perturbations

As in the proof of Lemma 4 in [19], we employ Taylor's series expansions of $f(\cdot)$ at $x_n \pm \delta_n d_n$ to obtain

$$\frac{f(x_n + \delta_{(n3^N+m)}d_m) + f(x_n - \delta_{(n3^N+m)}d_m) - 2f(x_n)}{\delta_{(n3^N+m)}^2}$$

$$= d_m^\mathsf{T} \nabla^2 f(x_n) d_m + O(\delta_{(n3^N+m)}^2)$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) + O(\delta_{(n3^N+m)}^2)$$

$$= \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) + O(\delta_{(n3^N+m)}^2). \qquad (31)$$

Recall that the Hessian estimate is given by

$$\widehat{H}_n = \frac{1}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} M_m \left(\frac{y_m^+ + y_m^- - 2y_m}{\delta_n^2} \right), \quad \text{where}$$

M_m is as defined in Algorithm 2. For the sake of simplicity, we ignore the zero-mean noise term $\xi_m^+ + \xi_m^- - 2\xi_m$ temporarily and analyze the following product inside the Hessian estimate:

$$\frac{1}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} M_m \left(\sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) \right). \qquad (32)$$

4.3.2 Off-diagonal terms in (32)

We now consider the (k, l)th term in (32): Assume without loss of generality, that $k < l$. Then,

$$\frac{1}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} \left[d_m^k d_m^l \left(\sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) \right) \right]$$

19

$$= \frac{1}{(2 \times 3^N)^2} \sum_{i=1}^{N} \nabla_{ii}^2 f(x_n) \sum_{m=0}^{3^N-1} \left(d_m^k d_m^l (d_m^i)^2 \right) + \frac{1}{(2 \times 3^N)^2} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \nabla_{ij}^2 f(x_n) \sum_{m=0}^{3^N-1} \left(d_m^k d_m^l d_m^i d_m^j \right) \tag{33}$$

$$= \nabla_{kl}^2 f(x_n). \tag{34}$$

The last equality follows from the fact that the first term in (33) is 0 since $k \neq l$ and

$$\sum_{m=0}^{3^N-1} \left(d_m^k d_m^l (d_m^i)^2 \right) = 0 \text{ for any } i,$$

while the second term in (33) can be seen to be equal to $\nabla_{kl}^2 f(x_n)$ using induction arguments similar to proof of Lemma 2.

4.3.3 Diagonal terms in (32)

Consider the lth diagonal term inside the conditional expectation in (32):

$$\frac{\kappa}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} \left((d_m^l)^2 - (2 \times 3^N) \right) \times \left(\sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) \right)$$

$$= \frac{\kappa}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} (d_m^l)^2 \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + \frac{2\kappa}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} (d_m^l)^2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n)$$

$$- \frac{\kappa}{(2 \times 3^N)} \sum_{m=0}^{3^N-1} \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) - \frac{2\kappa}{(2 \times 3^N)} \sum_{m=0}^{3^N-1} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n). \tag{35}$$

Now, we analyze each of the four terms on the RHS above.

The first term on the RHS of (35) can be simplified as follows:

$$\frac{\kappa}{(2 \times 3^N)^2} \sum_{m=0}^{3^N-1} (d_m^l)^2 \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n)$$

$$= \frac{\kappa}{(2 \times 3^N)^2} \left(\sum_{m=0}^{3^N-1} (d_m^l)^4 \nabla_{ll}^2 f(x_n) + \sum_{m=0}^{3^N-1} \sum_{i=1, i \neq l}^{N} (d_m^l)^2 (d_m^i)^2 \nabla_{ii}^2 f(x_n) \right)$$

$$= \frac{\kappa}{(2 \times 3^N)^2} \left((2 \times 3^{N+1}) \nabla_{ll}^2 f(x_n) + (2 \times 3^N)^2 \sum_{i=1, i \neq l}^{N} \nabla_{ii}^2 f(x_n) \right).$$

For the second equality above, we have used the fact that $\sum_{m=0}^{3^N-1} (d_m^l)^4 = 2 \times 3^{N+1}$ (easy to infer this claim along the lines of point (5) in the proof of Lemma 1) and $\sum_{m=0}^{3^N-1} (d_m^l)^2 (d_m^i)^2 = (2 \times 3^N)^2, \forall l \neq i$.

The second term in (35) is zero because $\sum_{m=0}^{3^N-1} \left(d_m^i d_m^j (d_m^l)^2 \right) = 0$ for any l and $i \neq j$ – a claim that can be easily proved using an induction argument.

The third term in (35) without the negative sign can be simplified as follows:

$$\frac{\kappa}{(2 \times 3^N)} \sum_{m=0}^{3^N-1} \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) = \frac{\kappa}{(2 \times 3^N)} \sum_{i=1}^{N} \sum_{m=0}^{3^N-1} (d_m^i)^2 \nabla_{ii}^2 f(x_n)$$

$$= \kappa \sum_{i=1}^{N} \nabla_{ii}^2 f(x_n), \text{ a.s.}$$

Combining the above followed by some algebra, we obtain

$$\frac{\kappa}{(2 \times 3^N)^2} \mathbb{E}\left[\left((d_m^l)^2 - (2 \times 3^N) \right) \left(\sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) \right) \middle| \mathcal{F}_n \right]$$

$$= \nabla_{ll}^2 f(x_n). \tag{36}$$

The fourth term in (35) is zero from Lemma 1.

Denote the product in (32) by (A) and the (i,j)th term there by $(A)_{i,j}$. Then,

$$\mathbb{E}[\widehat{H}_n(i,j) \mid \mathcal{F}_n] = \mathbb{E}\left[(A) \mid \mathcal{F}_n \right] + \mathbb{E}[\xi_n^+ + \xi_n^- - 2\xi_n \mid \mathcal{F}_n] \tag{37}$$

$$= \nabla_{ij}^2 f(x_n) + O(\delta_{(n3^N)}^2).$$

The last equality above follows from (C3), while the term involving the factor (A) reduces to the true Hessian with a bias of $O(\delta_{(n3^N)}^2)$ due to (34) and (36).

The main claim follows.

Case 2: Permutation matrix-based perturbations

From the first step involving Taylor series expansions in Case 1, we have

$$\frac{f(x_n + \delta_{(nN+m)} d_m) + f(x_n - \delta_{(nN+m)} d_m) - 2f(x_n)}{\delta_{(nN+m)}^2}$$

$$= \sum_{i=1}^{N} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_m^i d_m^j \nabla_{ij}^2 f(x_n) + O(\delta_{(nN+m)}^2). \tag{38}$$

Using the facts that $\sum_{m=0}^{N-1} \sum_{i=1}^{N} d_m^i d_m^j = 0$ for $i \neq j$ and $\sum_{m=0}^{N-1} (d_m^i)^2 = 1$ for any i, we obtain

$$\sum_{m=0}^{N-1} \left[\frac{f(x_n + \delta_{(nN+m)} d_m) + f(x_n - \delta_{(nN+m)} d_m) - 2f(x_n)}{\delta_{(nN+m)}^2} \right]$$

$$= \sum_{i=1}^{N} \sum_{m=0}^{N-1} (d_m^i)^2 \nabla_{ii}^2 f(x_n) + 2 \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \nabla_{ij}^2 f(x_n) \sum_{m=0}^{N-1} d_m^i d_m^j + O(\delta_{(nN+m)}^2) \tag{39}$$

$$= \nabla_{ii}^2 f(x_n) + O(\delta_{(nN)}^2).$$

Denote the term on LHS in (39) by (B), and following an argument similar to that used in simplifying the noise terms in (37), we obtain

$$\mathbb{E}[\widehat{H}_n(i,i) \mid \mathcal{F}_n] = \mathbb{E}\left[(B) \mid \mathcal{F}_n \right] + \mathbb{E}[\xi_n^+ + \xi_n^- - 2\xi_n \mid \mathcal{F}_n]$$

$$= \nabla_{ii}^2 f(x_n) + O(\delta_{(nN)}^2).$$

Hence proved. □

4.3.4 Proof of Theorem 7

Proof. Follows in a similar manner as that of the proofs of Theorem 5 and 6 in [19]. □

4.3.5 Proof of Theorem 8

Proof. From the quadratic assumption, we have that $H(x)$ is a constant, independent of x. From the proof leading up to (37), we have that $\widehat{H}_n = \nabla^2 H(x_n) = H^*$. Now, we follow the technique from [25] to derive the main claim.

Notice that

$$\Lambda_k = (1 - b_k)\Lambda_{k-1} + b_k(\overline{H}_k - H^*) = (1 - b_k)\Lambda_{k-1}.$$

Unrolling the recursion above, we obtain

$$\Lambda_n = \left[\prod_{k=1}^{n}(1 - b_k)\right]\Lambda_0.$$

Using the above, we can arrive at a simpler representation for the trace $\left[\mathbb{E}(\Lambda_n^T \Lambda_n)\right]$ as follows:

$$\mathbb{E}(\Lambda_n^T \Lambda_n) = \left[\prod_{k=1}^{n}(1 - b_k)\right]^2 \mathbb{E}(\Lambda_0^T \Lambda_0), \text{ leading to}$$

$$\text{trace}\left[\mathbb{E}(\Lambda_n^T \Lambda_n)\right] = \left[\prod_{k=1}^{n}(1 - b_k)\right]^2 \text{trace}\left[\mathbb{E}(\Lambda_0^T \Lambda_0)\right].$$

Simplifying further, using the fact that $1 - b_k = e^{-b_k}(1 - O(b_k^2))$, with the $O(b_k^2)$ being strictly positive as e^{-b_k} is convex, we have

$$\text{trace}\left[\mathbb{E}(\Lambda_n^T \Lambda_n)\right] = e^{-2b_{sum}(1,n)}c_{0n}\text{trace}\left[\mathbb{E}(\Lambda_0^T \Lambda_0)\right], \tag{40}$$

where $b_{sum}(i,j) = \sum_{k=i}^{j} b_k$ and $c_{kn} = \left[\prod_{i=k+1}^{n}(1 - O(b_i^2))\right]^2, k \geq 0$ and $c_{nn} = 1$. Since $0 < b_k < 1$, $\forall k \geq 2$, and $r > 1/2$, the c_{kn} are uniformly bounded in magnitude. Further,

$$b_{sum}(1,n) \geq \int_1^{n+1} \frac{b_0}{x^r}dx \geq \left(\frac{b_0}{1 - r}\right)\left(n^{1-r} - 1\right).$$

Using the bound derived above in (40), we obtain

$$\text{trace}\left[\mathbb{E}(\Lambda_n^T \Lambda_n)\right] \leq e^{-2b_0 n^{1-r}/(1-r)}e^{2b_0}c_{0n}\text{trace}\left[\mathbb{E}(\Lambda_0^T \Lambda_0)\right].$$

The main claim follows. □

5 Numerical Experiments

5.1 Implementation[5]

We consider the following problem:

$$\min_x \mathbb{E}_\xi\left[F(x,\xi)\right], \tag{41}$$

---

[5]The implementation is available at `https://github.com/prashla/RDSA/archive/master.zip`.

where $F(x, \xi)$ is the sample observation of the objective function $f(x)$ corrupted with zero mean noise ξ. In particular, the noise is $[x^T, 1]\xi$, where ξ is a multivariate Gaussian distribution with mean zero and covariance $\sigma^2 \mathcal{I}_{N+1}$. A similar noise structure has been used earlier in the implementation of both RDSA and SPSA algorithms in [19] and [24], respectively. For all the experiments, we consider two different settings of noise: (i) low noise with $\sigma = 0.001$; and (ii) high noise with $\sigma = 0.1$.

We consider three different functional forms for $F(x, \xi)$, namely quadratic, fourth-order and Rastrigin, respectively, in both $N = 5$ and $N = 10$ dimensions, for evaluating our algorithms. Before describing the example functions, we present the details about the algorithms implemented.

We implement the first-order and second-order algorithms proposed in this paper and compare them with several baselines that are based on the simultaneous perturbation method.

The first-order algorithms implemented include 1RDSA-Lex-DP, 1RDSA-Perm-DP and 1RDSA-KW-DP - the three deterministic perturbation variants of 1RDSA (see Section 2 for a detailed description). We compare these algorithms with the following baselines: RDSA with uniform and asymmetric Bernoulli perturbations proposed in [19] and henceforth referred to as 1RDSA-Unif and 1RDSA-AsymBer, respectively, and SPSA with Bernoulli perturbations, henceforth referred to as 1SPSA.

The second-order algorithms implemented include 2RDSA-Lex-DP and 2RDSA-Perm-DP - the two deterministic perturbation variants of 2RDSA (see Section 3 for a detailed description). We compare these algorithms with the following baselines: second-order RDSA with uniform and asymmetric Bernoulli perturbations proposed in [19] and henceforth referred to as 2RDSA-Unif and 2RDSA-AsymBer, respectively, and second-order SPSA with Bernoulli perturbations, henceforth referred to as 2SPSA.

The settings of the parameters δ_n and a_n for both first- and second-order algorithms are listed in Table 3, and a similar setting has been used in implementation of both RDSA and SPSA algorithms in [19] and [24], respectively. The distribution parameter for RDSA variants is set as follows: $\eta = 1$ for RDSA-Unif, $\epsilon = 0.0001$ for 1RDSA-AsymBer, and $\epsilon = 1$ for 2RDSA-AsymBer, and a similar setting has been used earlier for RDSA implementation in [19]. Each coordinate of the parameter is projected onto the set $[-2.048, 2.047]$, which helps to keep the iterates stable. All results are averages over 50 independent runs.

Table 3: Step-size and perturbation constant parameter settings, for first and second order algorithms.

Algorithms	δ_n	a_n
First-order	$1.9/n^{0.101}$	$1/(n + 50)$
Second-order	$3.8/n^{0.101}$	$1/n^{0.6}$

To compare the algorithms' performance, we use parameter error as the performance metric. For a given simulaton budget, the parameter error measures the distance between the final iterate obtained after the final update iteration and the optimum parameter x^*. More precisely, we use the following form for the parameter error, after suitable normalization:

$$\text{Parameter error} = \frac{||x_\tau - x^*||^2}{||x_0 - x^*||^2},$$

where τ is the number of times x is updated until the end of the simulation. Notice that τ varies with the algorithm and the number of function measurements. For example, with a budget of 5000 measurements for $N = 10$, $\tau = 250$ for 1RDSA-Perm-DP and 1RDSA-KW-DP as they use $2N$ measurements per iteration. Further, $\tau = 2500$ for 1SPSA as well as for both variants of 1RDSA, as they use two measurements per iteration. Notice that 1RDSA-Lex-DP does not make much progress under low simulation budgets, as it requires 2×3^N measurements per iteration. On the other hand, for second-order algorithms, an initial 20% of the measurements were used up by the corresponding first-order algorithm for initialization. Thus,

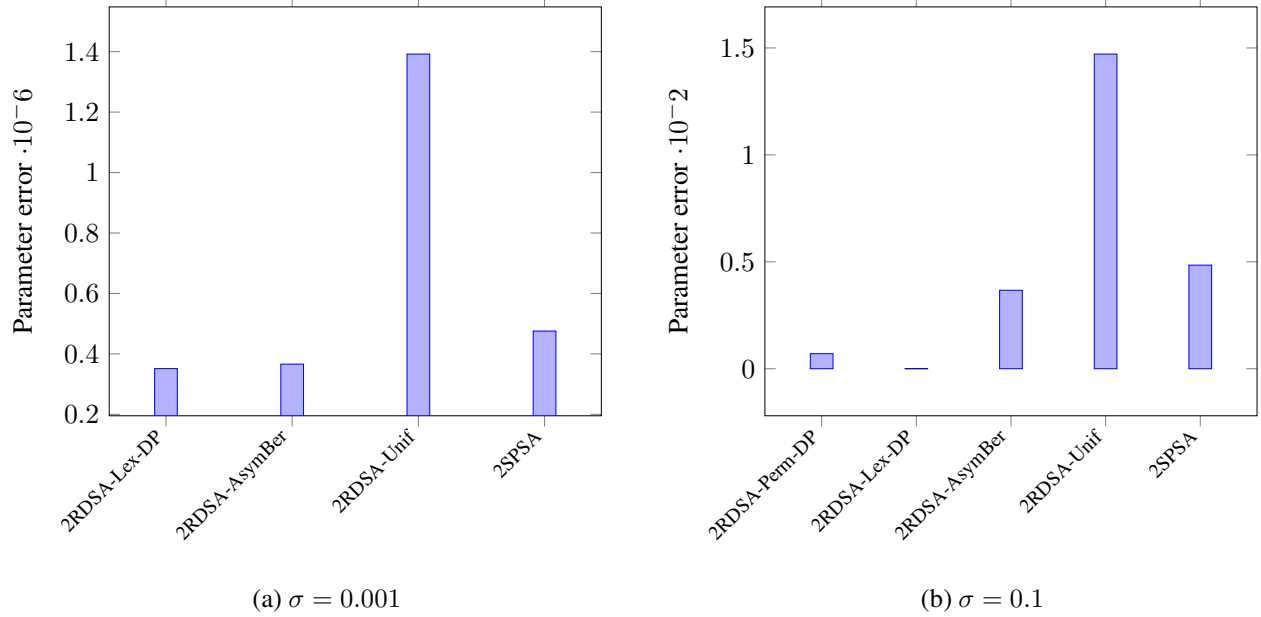(a) $\sigma = 0.001$ (b) $\sigma = 0.1$

Figure 1: Parameter error for various second-order algorithms under the quadratic objective (42) for a five-dimensional problem with a simulation budget of 50000 and $\sigma = 0.001$ and 0.1.

for $N = 10$ and a budget of 5000 measurements, the initial 1000 measurements are used for the first-order algorithm and the remaining 4000 are used by the second-order algorithm. As a consequence of the simulation budget split, the number of update iterations $\tau = 4000/30 \approx 133$ for 2RDSA-Perm-DP, $4000/4 = 1000$ for 2SPSA, and $4000/3 \approx 1333$ for 2RDSA algorithms. The 2RDSA-Lex-DP algorithm does not output a meaningful parameter with a low simulation budget, owing to its high inner loop length. The difference here is due to the fact that 2RDSA-Perm-DP uses $3N$ measurements per iteration, while 2RDSA-Lex-DP needs 3×3^N, 2RDSA needs 3 and 2SPSA needs 4.

5.2 Example 1: Quadratic objective

Let A be such that NA is an $N \times N$ upper triangular matrix with $a_{ij} = 1$ for $i \leq j$ and let b be an N-dimensional vector of ones. Then the quadratic objective function is defined as follows:

$$F(x, \xi) = x^T A x + b^T x + \xi. \tag{42}$$

The initial point x_0 is set to the N-dimensional vector of ones and the optimal point x^* is dimension dependent. For instance, with $N = 10$, the optimal point x^*, is the 10-dimensional vector of -0.9091 for the choice of A and b described earlier. Note that $f(x^*) = \mathbb{E}_\xi[F(x^*, \xi)] = -4.55$.

Figures 1 and 2 present the parameter error for the first-order and second-order algorithms under the quadratic objective (42) for dimension 5 and $\sigma = 0.001$ and 0.1.

Among the first-order algorithms, for both settings of noise, 1RDSA-Perm-DP and 1RDSA-KW-DP exhibited similar performance and outperformed the other algorithms. The parameter error in 1RDSA-Perm-DP and 1RDSA-KW-DP is of the order of 10^{-5}, while for the others, the same is of the order of 10^{-3}. 1RDSA-Lex-DP (not shown in the figure) showed a parameter error that was an order of magnitude higher than the other algorithms.

Among the second-order algorithms, for both settings of noise, 2RDSA-Lex-DP exhibited the best performance. Furthermore, it is interesting to see that for both settings of noise 2RDSA-Perm-DP, and 2RDSA-Lex-DP gave consistent performance with parameter error of the order of 10^{-4} and 10^{-7}, respectively, while

24

the parameter error of 2RDSA-AsymBer, 2RDSA-Unif, and 2SPSA increased with noise. Further, the benefit of using second-order algorithms is more noticeable under the low noise setting.

For the low noise setting, the parameter error of 2RDSA-Perm-DP is not shown in the figure, for the sake of readability in comparing the errors of the other algorithms.
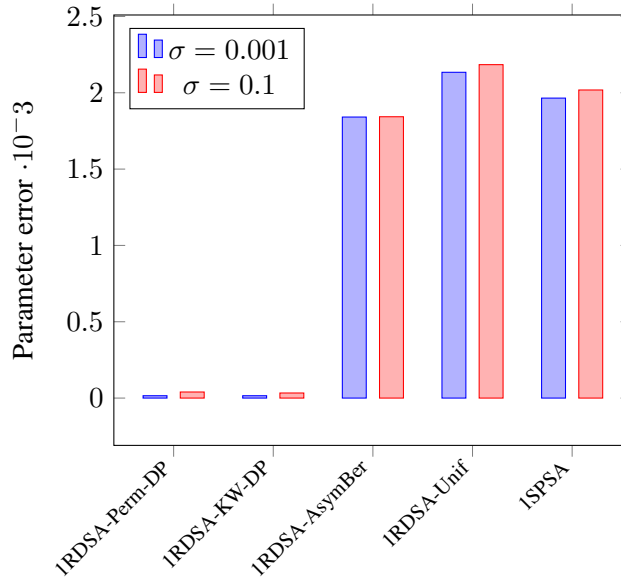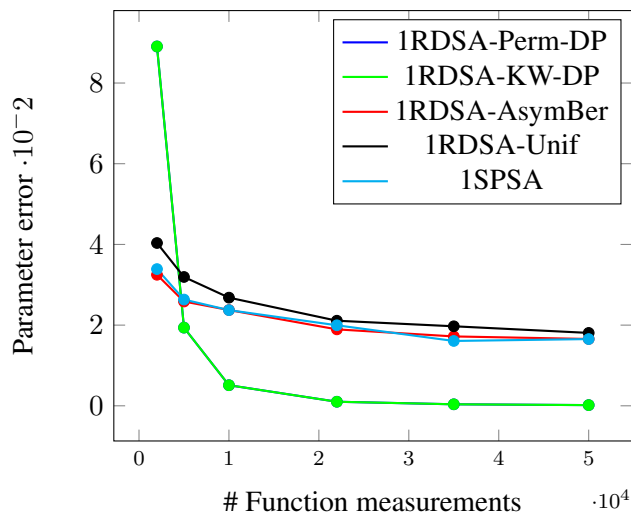

Figure 2: Parameter error for various first-order algorithms under the quadratic objective (42) for a five-dimensional problem with a simulation budget of 50000 and $\sigma = 0.001$ and 0.1.

Figure 3: Evolution of the parameter error as the simulation budget is varied, for the first-order algorithms under the quadratic objective with $N = 10$ and $\sigma = 0.001$.

Figure 3 compares the parameter error of 1RDSA-Perm-DP, 1RDSA-KW-DP, and both variants of 1RDSA and 1SPSA algorithms for the quadratic objective with dimension 10, $\sigma = 0.001$ and a simulation budget of 50000 function measurements. As in the case of the problem with dimension 5, 1RDSA-Perm-DP and 1RDSA-KW-DP performed best, while the result of 1RDSA-Lex-DP is not reported due to its high

inner loop length.

5.3   Example 2: Fourth-order objective

The function given below has been used for evaluating both RDSA and SPSA algorithms in [19] and [24], respectively.

$$F(x,\xi) = x^T A^T A x + 0.1 \sum_{j=1}^{N} (Ax)_j^3 + 0.01 \sum_{j=1}^{N} (Ax)_j^4 + \xi, \tag{43}$$
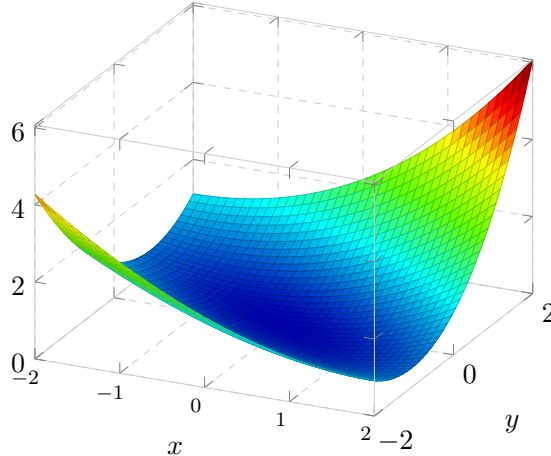

Figure 4: A plot of the fourth-order objective (43), $N = 2$.

where A and ξ are the same as in the quadratic objective. The initial point x_0 is set to the N-dimensional vector of ones and the optimal point x^* is the N-dimensional vector of zeros, with $f(x^*) = \mathbb{E}_\xi[F(x^*, \xi)] = 0$. Figure 4 shows a plot of the fourth-order objective (43).

Figure 5 presents the parameter error for both first and second order algorithms in the case of the fourth-order objective (43) with dimension 5 and $\sigma = 0.001$ and 0.1. Among the first-order algorithms, for both settings of noise, all algorithms except 1RDSA-Lex-DP exhibited similar performance.

Among the second-order algorithms, similar to the quadratic case, for both settings of noise, 2RDSA-Perm-DP and 2RDSA-Lex-DP gave consistent performance with parameter error of the order of 10^{-2}. Under the low noise setting, i.e., $\sigma = 0.001$, 2RDSA-Unif exhibited the best performance, while under the high noise setting with $\sigma = 0.1$, 2RDSA-Lex-DP outperformed the other algorithms. Thus, we observe that 2RDSA-Lex-DP and 2RDSA-Perm-DP algorithms are more tolerant to the noise, as compared to their random counterparts 2RDSA-AsymBer, 2RDSA-Unif, and 2SPSA.

Since the fourth-order objective is more difficult to optimize than the quadratic one, we observe that under the low noise setting, the parameter error in the case of the fourth-order objective for the first-order algorithms is higher, of the order of 10^{-2} compared to 10^{-5} for the quadratic case. For the second-order algorithms, the same is also of the order of 10^{-2} compared to 10^{-6} for the quadratic case. A similar trend is observed in the high noise regime.

Figure 6 compares the parameter error of 2RDSA-Perm-DP, as well as both variants of 2RDSA and 2SPSA algorithms, for the fourth-order objective function with dimension 10, $\sigma = 0.001$ and a simulation budget of 50000 function measurements. As in the five-dimensional problem, 2RDSA-Unif and 2SPSA exhibited similar performance and outperformed the other algorithms. The results of the 2RDSA-Lex-DP algorithm are not displayed as it requires 3×3^{10} function measurements per iteration.

26

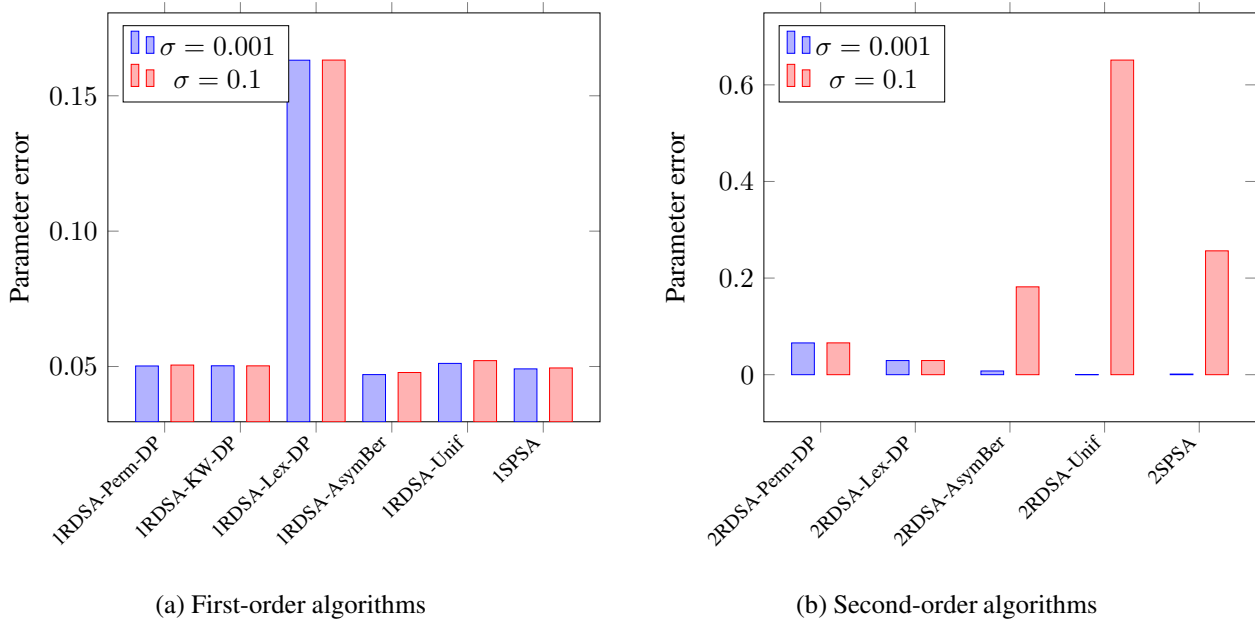(a) First-order algorithms   (b) Second-order algorithms

Figure 5: Parameter error for various algorithms under the fourth-order objective (43) for a five-dimensional problem with a simulation budget of 50000 and $\sigma = 0.001$ and 0.1.
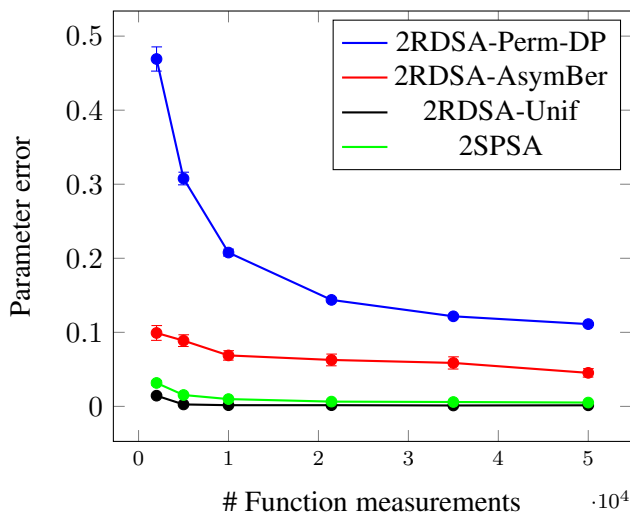

Figure 6: Evolution of the parameter error as the simulation budget is varied, for the second-order algorithms under the fourth-order objective with $N = 10$ and $\sigma = 0.001$.

5.4 Example 3: Rastrigin objective

The Rastrigin objective function is defined as follows:

$$F(x, \xi) = \sum_{i=1}^{N} (x_i^2 - 10\cos(2\pi x_i)) + 10N + 1 + \xi, \tag{44}$$

where ξ is the same as for the quadratic objective. The initial point x_0 is set to the N-dimensional vector of twos and the optimal point x^* is the N-dimensional vector of zeros, with $f(x^*) = \mathbb{E}_\xi[F(x^*, \xi)] = 1$. Figure

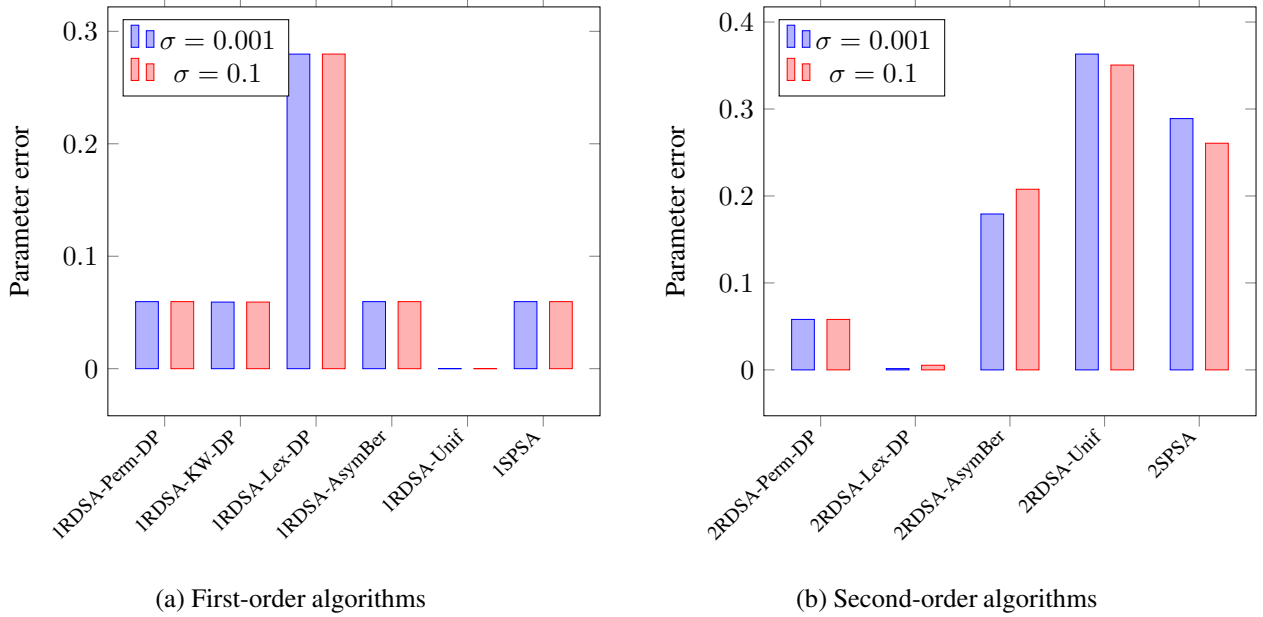(a) First-order algorithms

(b) Second-order algorithms

Figure 7: Parameter error for various algorithms under the Rastrigin objective (44) for a five-dimensional problem and a simulation budget of 50000 and $\sigma = 0.001$ and 0.1.

8 shows a plot of the Rastrigin objective (44), which has many local minima.

Figure 7 presents the parameter error for both first-order and second-order algorithms for the Rastrigin objective (44) with dimension 5 and $\sigma = 0.001$ and 0.1.

For both settings of noise, among the first-order algorithms, 1RDSA-Unif outperformed the other algorithms, while 2RDSA-Lex-DP performed the best among the second-order algorithms.

Similar to the quadratic and fourth-order objective, first-order algorithms, 2RDSA-Perm-DP and 2RDSA-Lex-DP gave consistent performance under both settings of noise.

In summary, for all the three objectives, among the first-order algorithms, we observed that 1RDSA-Perm-DP and 1RDSA-KW-DP performed best, while 1RDSA-Lex-DP showed poor performance. On the other hand, among the second-order algorithms, 2RDSA-Lex-DP exhibited the best performance.
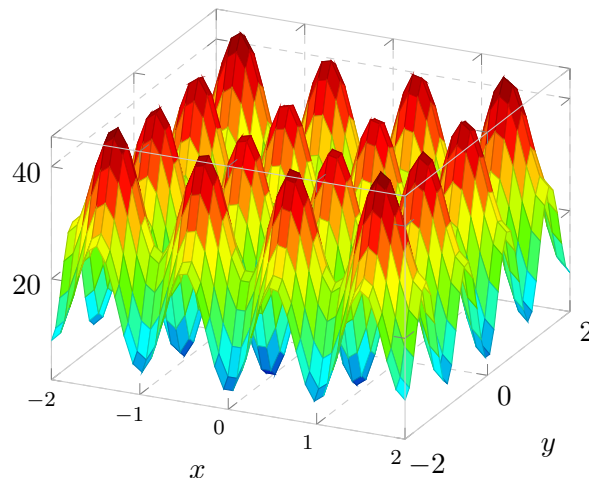

Figure 8: A plot of the Rastrigin objective (44), $N = 2$.

6 Conclusions and Future Work

We incorporated two novel deterministic perturbation (DP) schemes into the RDSA class of simultaneous perturbation algorithms. The proposed DP variants of first-order, as well as second-order, RDSA were shown to result in asymptotically unbiased gradient/Hessian estimates, thus resulting in provably convergent 1RDSA/2RDSA variants. We also performed numerical experiments to validate the theoretical findings.

As future work, it would be interesting to perform a finite-time analysis of the RDSA schemes and study the impact of deterministic perturbations therein. To the best of our knowledge, non-asymptotic bounds are not available for vanilla RDSA algorithms, especially. in a general simulation optimization setting where convexity cannot be assumed. Another fruitful direction would be try the deterministic perturbation variants of RDSA in sophisticated applications, e.g., in transportation, networks and service systems.

References

[1] S. Bhatnagar. Adaptive multivariate three-timescale stochastic approximation algorithms for simulation based optimization. *ACM TOMACS*, 15(1):74–107, 2005.

[2] S. Bhatnagar. Adaptive Newton-based smoothed functional algorithms for simulation optimization. *ACM Transactions on Modeling and Computer Simulation*, 18(1):2:1–2:35, 2007.

[3] S. Bhatnagar and L. A. Prashanth. Simultaneous perturbation Newton algorithms for simulation optimization. *Journal of Optimization Theory and Applications*, 164(2):621–643, 2015.

[4] S Bhatnagar, H. L. Prasad, and L. A. Prashanth. *Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods (Lecture Notes in Control and Information Sciences)*, volume 434. Springer, 2013.

[5] Shalabh Bhatnagar, Michael C Fu, Steven I Marcus, and I Wang. Two-timescale simultaneous perturbation stochastic approximation using deterministic perturbation sequences. *ACM TOMACS*, 13(2): 180–209, 2003.

[6] H. F. Chen, L. Guo, and A. J. Gao. Convergence and robustness of the Robbins-Monro algorithm truncated at randomly varying bounds. *Stochastic Processes and their Applications*, 27:217–231, 1987.

[7] D. C. Chin. Comparative study of stochastic algorithms for system optimization based on gradient approximations. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(2): 244–249, 1997.

[8] D. Ghoshdastidar and A. Dukkipati and S. Bhatnagar. Newton based stochastic optimization using q-Gaussian smoothed functional algorithms. *Automatica*, 50(10):2606–2614, 2014.

[9] Gal Dalal, Balazs Szorenyi, Gugan Thoppe, and Shie Mannor. Finite sample analysis of two-timescale stochastic approximation with applications to reinforcement learning. In *Conference on Learning Theory*, pages 1–35, 2018.

[10] V. Fabian. Stochastic approximation. In *Optimizing Methods in Statistics (ed. J.J.Rustagi)*, pages 439–470, New York, 1971. Academic Press.

[11] N. Frikha and S. Menozzi. Concentration bounds for stochastic approximations. *Electron. Commun. Probab*, 17(47):1–15, 2012.

[12] D. Ghoshdastidar, A. Dukkipati, and S. Bhatnagar. Smoothed functional algorithms for stochastic optimization using q-Gaussian distributions. *ACM Transactions on Modeling and Computer Simulation*, 24(3):17:1–17:26, 2014.

[13] Edgar G Goodaire. *Linear Algebra: Pure & Applied*. World Scientific Publishing Company, 2013.

[14] Xiaowei Hu, L. A. Prashanth, András György, and Csaba Szepesvári. (Bandit) convex optimization with biased noisy gradient oracles. In *AISTATS*, volume 41, pages 819—828, 2016.

[15] J. C. Spall. A one-measurement form of simultaneous perturbation stochastic approximation. *Automatica*, 33(1):109–112, 1997.

[16] V. Ya Katkovnik and Yu Kulchitsky. Convergence of a class of random search algorithms. *Automation Remote Control*, 8:1321–1326, 1972.

[17] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.*, 23:462–466, 1952.

[18] H. J. Kushner and D. S. Clark. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer Verlag, New York, 1978.

[19] L A Prashanth, Shalabh Bhatnagar, Michael C. Fu, and Steve I Marcus. Adaptive system optimization using random directions stochastic approximation. *IEEE Transactions on Automatic Control*, 62(5): 2223–2238, 2017.

[20] D Sai Koti Reddy, LA Prashanth, and Shalabh Bhatnagar. Improved Hessian estimation for adaptive random directions stochastic approximation. In *IEEE Conference on Decision and Control (CDC)*, pages 3682–3687, 2016.

[21] H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22:400–407, 1951.

[22] R. Y. Rubinstein. *Simulation and the Monte Carlo Method*. Wiley, New York, 1981.

[23] Reuven Y Rubinstein and Alexander Shapiro. *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method*. Wiley, New York, 1993.

[24] J. C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Trans. Autom. Contr.*, 45:1839–1853, 2000.

[25] J. C. Spall. Feedback and weighting mechanisms for improving Jacobian estimates in the adaptive simultaneous perturbation algorithm. *IEEE Trans. Autom. Contr.*, 54(6):1216–1229, 2009.

[26] James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

[27] M. A. Styblinski and T.-S. Tang. Experiments in nonconvex optimization: stochastic approximation with function smoothing and simulated annealing. *Neural Networks*, 3:467–483, 1990.