J Optim Theory Appl
DOI 10.1007/s10957-013-0507-1

Simultaneous Perturbation Newton Algorithms for Simulation Optimization

Shalabh Bhatnagar · L.A. Prashanth

Received: 17 April 2013 / Accepted: 6 December 2013

Abstract We present a new Hessian estimator based on the simultaneous perturbation procedure, that requires three system simulations regardless of the parameter dimension. We then present two Newton-based simulation optimization algorithms that incorporate this Hessian estimator. The two algorithms differ primarily in the manner in which the Hessian estimate is used. Both our algorithms do not compute the inverse Hessian explicitly, thereby saving on computational effort. While our first algorithm directly obtains the product of the inverse Hessian with the gradient of the objective, our second algorithm makes use of the Sherman–Morrison matrix inversion lemma to recursively estimate the inverse Hessian. We provide proofs of convergence for both our algorithms. Next, we consider an interesting application of our algorithms on a problem of road traffic control. Our algorithms are seen to exhibit better performance than two Newton algorithms from a recent prior work.

Keywords Newton algorithms · Stochastic approximation · Simultaneous perturbation stochastic approximation · Three-simulation · Hessian estimate · Sherman–Morrison lemma · Application to road traffic control

Communicated by Ilio Galligani.

S. Bhatnagar (✉)
Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India
e-mail: shalabh@csa.iisc.ernet.in

L.A. Prashanth
Nord Europe, Team SequeL, INRIA, Lille, France
e-mail: prashanth.la@inria.fr

Published online: 17 December 2013

1 Introduction

Simulation optimization approaches are a class of techniques that aim at solving the problem of optimizing a parameterized performance objective using simulated outcomes or observations. Many times the performance objective and so also its gradient or higher order derivatives are not known analytically, however, noisy samples obtained from simulation are available. The problem of interest is to perform optimization under such 'noisy' information, many times without knowing the system model such as the transition probabilities of an underlying Markov process. A host of methods developed over the course of the past few decades have been devised to tackle this problem. For continuous parameter optimization problems, perturbation analysis (PA) [1, 2], and likelihood ratio (LR) approaches [3] work with sample path gradients. Where applicable, these approaches are known to be highly efficient. However, in many systems of interest, PA and LR approaches cannot be easily applied as they require strict regularity conditions on the system model and performance objective.

A number of approaches have been aimed at estimating the gradient of the objective function. The earliest amongst them is due to Kiefer and Wolfowitz [4] whose gradient estimation procedure requires $2N$ simulations for one estimate of the performance gradient with respect to an N-dimensional parameter. A one-sided version of this procedure requires $(N+1)$ system simulations. The simultaneous perturbation stochastic approximation (SPSA) algorithm due to Spall [5], on the other hand, requires only two simulations regardless of the parameter dimension (N) and has been found to be very effective in several settings. The SPSA algorithm is a random search procedure that estimates the gradient at each update epoch by running the two aforementioned simulations with randomly perturbed parameters. A one-simulation version of this algorithm presented in [6], however, does not perform well in practice (unlike its two-simulation counterpart). In [7], SPSA-based algorithms for the long-run average cost objective have been presented. In [8], SPSA algorithms for the average cost objective that are based on certain deterministic perturbation constructions (instead of random perturbations) have been proposed. A one-simulation algorithm in [8] that incorporates a Hadamard matrix based construction for the perturbations is seen to perform significantly better than its corresponding random perturbation (one-simulation) counterpart. This is because the Hadamard matrix based construction is observed in [8] to result in a frequent and regular cancellation of bias terms in the gradient estimator, as opposed to its random perturbation counterpart. In [9, 10], random perturbation algorithms with Gaussian distributed perturbations have also been proposed. A textbook account of various simultaneous perturbation based stochastic search procedures is available in [11].

There has also been work done on the development of Newton-based algorithms and corresponding techniques for Hessian estimation. In [12], Hessian estimates that require $O(N^2)$ samples of the objective function at each update epoch have been presented. In a novel extension of the simultaneous perturbation technique for gradient estimation, Spall [13] presents simultaneous perturbation estimates of the Hessian that require four simulations. Four Newton-based simultaneous perturbation algorithms for the long-run average cost objective have been presented in [14]. These

require four, three, two and one simulation(s), respectively. The four-simulation algorithm in [14] incorporates a similar Hessian estimator as the one proposed in [13]. It was observed through experiments in [14] that the four-simulation algorithm shows the best results, followed by the three-simulation algorithm in cases where the parameter dimension is low. However, in the case of parameters with higher dimension, the three-simulation algorithm shows the best results overall, followed by the four-simulation algorithm.

The three-simulation Hessian estimate in [14] is not balanced (i.e., it is not symmetric) and hence results in certain bias terms (that however have zero mean). In this paper, we present a novel balanced estimate of the Hessian that is based on three simulations. As a consequence of its 'balanced' nature, many of the aforementioned bias terms are not present in our Hessian estimate. We also present two Newton-based algorithms that incorporate the (above) balanced Hessian estimate. In Newton-based algorithms, one typically needs to compute the inverse of the Hessian estimate at each update epoch. This operation is computationally intensive, particularly when the parameter dimension is high. However, our algorithms require less computation as they do not require an explicit Hessian inversion. Our first algorithm makes use of a parameter whose update involves the true Hessian as well as gradient estimates of the objective function, and which upon convergence gives implicitly the product of the Hessian inverse with the objective function gradient.

In our second algorithm, we circumvent the problem of computational complexity of the procedure that would otherwise result from an explicit Hessian inversion by incorporating an update procedure obtained from the Sherman–Morrison lemma for matrix inversion (that we derive for our case using the Woodbury identity). A similar procedure in the case of the two-simulation algorithm from [14] has been proposed in the context of discrete optimization in service systems (see Chap. 12 of [11]). This procedure directly updates the inverse of the Hessian and thereby has lower computational requirements. We provide a proof of convergence for both our algorithms.

Finally, as an application setting, we consider the problem of road traffic control, i.e., of finding an optimal order to switch traffic lights dynamically in vehicular traffic networks (see [15, 16]). Our algorithms are seen to perform significantly better than the two Newton algorithms proposed in [14], which are based on four and three simulations, respectively. As mentioned previously, the four-simulation algorithm of [14] uses similar gradient and Hessian estimators as the algorithm in [13]. Our algorithms also show better results in most cases over a large set of fixed threshold schemes. In order to save space, some details of the experimental set-up have been described in an online technical report [17].

The rest of the paper is organized as follows. In Sect. 2, we present the basic problem framework. We present the algorithms in Sect. 3. The convergence analysis is given in Sect. 4 while the simulation results, in Sect. 5. Finally, concluding remarks are in Sect. 6.

2 The Framework

Let $\{X_n, n \geq 1\}$ be an $\mathbb{R}^d$-valued ($d \geq 1$) Markov process parameterized with a parameter $\theta \in C \subset \mathbb{R}^N$, where C is a given compact and convex set. Let $p(\theta, dx, dy)$,

$x, y \in \mathbb{R}^d$ denote the transition probabilities. We assume that for any given $\theta \in C$, $\{X_n\}$ is ergodic Markov. Let $h : \mathbb{R}^d \to \mathbb{R}$ be a given Lipschitz continuous cost function. Our aim is to find a $\theta^* \in C$ that minimizes over all $\theta \in C$, the long-run average cost

$$J(\theta) := \lim_{l\to\infty} \frac{1}{l} \sum_{j=0}^{l-1} h(X_j). \tag{1}$$

The above limit exists because of the ergodicity of $\{X_n\}$ for any $\theta \in C$.

Assumption 2.1 *$J(\theta)$ is twice continuously differentiable in θ.*

Assumption 2.1 is a standard requirement in most Newton search schemes (see, for instance, [10, 11, 13, 14]). Let $\{\theta(n)\}$ be a sequence of random parameters on which the process $\{X_n\}$ depends. Let $\mathcal{H}_n := \sigma(\theta(m), X_m, m \le n)$, $n \ge 1$ be the associated σ-fields. We call $\{\theta(n)\}$ nonanticipative if for all Borel sets $A \subset \mathbb{R}^d$,

$$P(X_{n+1} \in A \mid \mathcal{H}_n) = p\big(\theta(n), X_n, A\big) \quad \text{a.s.}$$

The sequences $\{\theta(n)\}$ obtained using both our algorithms in the next section can be seen to be nonanticipative. We shall assume the existence of a stochastic Lyapunov function (below).

Assumption 2.2 *There exist $\epsilon_0 > 0$, $K \subset \mathbb{R}^d$ compact and $V \in C(\mathbb{R}^d)$ such that $\lim_{\|x\|\to\infty} V(x) = \infty$. Further, under any nonanticipative $\{\theta(n)\}$,*

1. $\sup_n E[V(X_n)^2] < \infty$ *and*
2. $E[V(X_{n+1}) \mid \mathcal{H}_n] \le V(X_n) - \epsilon_0$, *whenever* $X_n \notin K$, $n \ge 0$.

Here we let $\|\cdot\|$ be the Euclidean norm. Also, for any matrix $A \in \mathbb{R}^{N\times N}$, its norm is defined as the induced matrix norm $\|A\| := \max_{\{x\in\mathbb{R}^N \mid \|x\|=1\}} \|Ax\|$. By an abuse of notation, both the vector and the matrix norms are denoted using $\|\cdot\|$.

We require Assumption 2.2 to ensure that the system remains stable under a tunable parameter. In Newton search algorithms that are geared towards finding a local minimum, one normally projects the Hessian estimate after each iteration onto the space of positive definite and symmetric matrices in order for the algorithm to progress in the negative gradient direction. Let $\Gamma : \mathbb{R}^{N\times N} \to \{\text{positive definite and symmetric matrices}\}$ denote this projection operator. We let $\Gamma(A) = A$ if A is positive definite and symmetric. In general, Γ can be obtained from various methods such as the *modified Choleski factorization scheme* [18] or the procedures in [13] and [19].

Assumption 2.3 *Let $\{A_n\}$ and $\{B_n\}$ be any sequences of matrices in $\mathbb{R}^{N\times N}$ such that $\lim_{n\to\infty} \|A_n - B_n\| = 0$. Then $\lim_{n\to\infty} \|\Gamma(A_n) - \Gamma(B_n)\| = 0$ as well. Further, for any sequence $\{C_n\}$ of matrices in $\mathbb{R}^{N\times N}$ with $\sup_n \|C_n\| < \infty$, $\sup_n \|\Gamma(C_n)\|$, $\sup_n \|\{\Gamma(C_n)\}^{-1}\| < \infty$ as well.*

Assumption 2.3 has also been used in other Newton-based schemes (see, for instance, [10, 11, 14]). As argued in [10, 14], the continuity requirement in Assumption 2.3 can be easily imposed in the *modified Choleski factorization procedure* (see [13, 18]). The procedure in [19] has also been shown to satisfy this requirement. A sufficient condition for the other requirements in Assumption 2.3 is that for some scalars $c_1, c_2 > 0$,

$$c_1\|z\|^2 \leq z^T \Gamma(C_n) z \leq c_2\|z\|^2, \quad \forall z \in \mathbb{R}^N, n \geq 0. \tag{2}$$

Then all eigenvalues of $\Gamma(C_n)$, $\forall n$, lie between c_1 and c_2. Further, $\sup_n \|\Gamma(C_n)\|$, $\sup_n \|\{\Gamma(C_n)\}^{-1}\| < \infty$ (see Propositions A.9 and A.15 in [18]).

3 The Algorithms

Both our algorithms incorporate three step-size sequences denoted $\{a(n)\}$, $\{b(n)\}$ and $\{c(n)\}$. We let the step-sizes satisfy the properties in the assumption below.

Assumption 3.1 *Let $a(n), b(n), c(n) > 0$ for all n. Further,*

$$\sum_n a(n) = \sum_n b(n) = \sum_n c(n) = \infty, \qquad \sum_n \left(a(n)^2 + b(n)^2 + c(n)^2\right) < \infty, \tag{3}$$

$$\lim_{n\to\infty} \frac{a(n)}{c(n)} = \lim_{n\to\infty} \frac{c(n)}{b(n)} = 0. \tag{4}$$

The conditions (3) are routinely used for step-sizes. As a consequence of the second requirement in (3), the step-sizes $a(n), b(n), c(n) \to 0$ as $n \to \infty$ and, in particular, the 'noise terms' in the algorithm diminish asymptotically. Thus, the various recursions in the algorithm can be considered to be noisy Euler discretizations (with diminishing step-sizes) of associated ODEs. Diminishing step-sizes of the algorithm also result in shrinking of the corresponding timescales for the various recursions. The first requirement in (3) is therefore needed to ensure that any given finite time interval in ODE time can be simulated by a finite (that could however be large) number of iterations, as a result of which the algorithm does not converge prematurely. The requirements in (4) imply that $a(n)$ converges to 0 at a faster rate as compared to $c(n)$ (as $n \to \infty$). Likewise, $c(n)$ converges to 0 faster than $b(n)$. Thus recursions governed by $b(n)$ are the fastest while those governed by $a(n)$ are the slowest. Further, recursions governed by $c(n)$ converge faster than those governed by $a(n)$ and slower than the ones governed by $b(n)$. This is mainly because one expects that beyond some integer m_0 (i.e., $\forall n \geq m_0$), the increments in the recursions governed by $b(n)$ ($c(n)$) would be uniformly larger than corresponding increments in recursions governed by $c(n)$ ($a(n)$).

Both our algorithms incorporate a new Hessian estimation scheme. The motivation for the Hessian estimate comes from the following form of the second derivative of the objective function when θ is a scalar:

$$\frac{d^2 J(\theta)}{d\theta^2} = \lim_{\delta\to 0}\left(\frac{J(\theta+\delta) + J(\theta-\delta) - 2J(\theta)}{\delta^2}\right). \tag{5}$$

We extend the estimate in (5) to vector parameters by using the simultaneous perturbation approach. Our estimate involves three simulation runs with a nominal (unperturbed) parameter and two different perturbed parameter vectors.

Both our algorithms aim at reducing the computational load involved with inverting the Hessian matrix at each update epoch. Our first algorithm (Algorithm 1) achieves the inversion in an indirect manner by making use of an additional recursion to estimate the product of the inverse of the Hessian with the gradient of the objective and uses only the Hessian update in the process. The second algorithm (Algorithm 2), on the other hand, incorporates the Sherman–Morrison lemma for matrix inversion (that we derive here from the Hessian update using the Woodbury identity) to directly update the inverse of the Hessian.

Let $\Delta(n) = (\Delta_1(n), \ldots, \Delta_N(n))^T$ and $\hat{\Delta}(n) = (\hat{\Delta}_1(n), \ldots, \hat{\Delta}_N(n))^T$, where $\Delta_i(n)$, $\hat{\Delta}_i(n)$, $n \geq 0$, $i = 1, \ldots, N$, denote perturbation random variables. We make the following assumption on these:

Assumption 3.2 *The random variables* $\Delta_1(n), \ldots, \Delta_N(n)$, $\hat{\Delta}_1(n), \ldots, \hat{\Delta}_N(n)$, $n \geq 0$ *are independent and identically distributed* (*i.i.d.*), *mean-zero with each* $\Delta_j(n)$, $\hat{\Delta}_j(n)$, $j = 1, \ldots, N$, *taking values in a compact interval* $E \subset \mathbb{R}$. *In addition, there exists a constant* $\bar{K} < \infty$, *such that for any* $n \geq 0$, *and* $l \in \{1, \ldots, N\}$,

$$E\big[\big(\Delta_l(n)\big)^{-2}\big] = E\big[\big(\hat{\Delta}_l(n)\big)^{-2}\big] \leq \bar{K}.$$

Further, $\Delta(n)$, $\hat{\Delta}(n)$ *are both independent of* $\sigma(\theta(l), l \leq n)$, *the* σ*-field generated by the sequence of parameter updates obtained till the* n*th iteration.*

We denote by $(\Delta(n))^{-1}$, the quantity $(\Delta(n))^{-1} = (1/\Delta_1(n), \ldots, 1/\Delta_N(n))^T$. Both our algorithms perform data averaging to estimate the long-run average cost for a given parameter value along the fastest timescale (using step-sizes $b(n), n \geq 0$) while the parameter itself is updated along the slowest scale (using $a(n), n \geq 0$). The Hessian updates are performed along a timescale (with step-sizes $c(n), n \geq 0$) that lies in between the two scales.

3.1 Algorithm 1

We present below a step-by-step summary of the algorithm update procedure.

Step 1 Run three parallel simulations $X^{--}(l)$, $X^{++}(l)$ and $X^{o}(l)$, $l \geq 0$, which are respectively governed by the parameter sequences $\theta^{--}(n) := \theta(n) - \delta\Delta(n) - \delta\hat{\Delta}(n)$, $\theta^{++}(n) := \theta(n) + \delta\Delta(n) + \delta\hat{\Delta}(n)$ and $\theta^{o}(n) := \theta(n)$, $n \geq 0$. Here $\delta > 0$ be a given small constant. Further, l and n are related according to $l = nL + m$, for some $m \in \{0, 1, \ldots, L-1\}$ and a given $L \geq 1$.

Step 2 Let $Z^{w}(nL+m)$, $w \in \{--, ++, o\}$, denote quantities used for averaging the cost function in the four simulations. Initialize $Z^{w}(0) = 0$, $\forall w \in \{--, ++, o\}$ and update $Z^{w}(\cdot)$ on the fastest timescale as follows: $\forall m = 0, 1, \ldots, L-1$,

$$Z^{--}(nL+m+1) = Z^{--}(nL+m) + b(n)\big(h\big(X^{--}(nL+m)\big) - Z^{--}(nL+m)\big), \tag{6}$$

$$Z^{++}(nL+m+1) = Z^{++}(nL+m) + b(n)\big(h\big(X^{++}(nL+m)\big) - Z^{++}(nL+m)\big), \tag{7}$$

$$Z^{o}(nL+m+1) = Z^{o}(nL+m) + b(n)\big(h\big(X^{o}(nL+m)\big) - Z^{o}(nL+m)\big). \tag{8}$$

Step 3 Update the estimate of the Hessian on the medium timescale as follows: For $i, j = 1, \ldots, N$,

$$H_{i,j}(n+1) = H_{i,j}(n) + c(n)\left(\frac{Z^{++}(nL) + Z^{--}(nL) - 2Z^{o}(nL)}{\delta^2 \Delta_i(n)\hat{\Delta}_j(n)} - H_{i,j}(n)\right). \tag{9}$$

Next form the matrix $P(n) = \Gamma([[H_{k,l}(n)]]_{k,l=1}^{N})$. Update

$$\beta(n+1) = \beta(n) + c(n)\left(-P(n)\beta(n) + \left(\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta}\big(\Delta(n)\big)^{-1}\right)\right). \tag{10}$$

Step 4 With the quantity $\beta(n)$ computed in step 3, update the parameter $\theta(n)$ on the slowest timescale as follows:

$$\theta(n+1) = \Pi\big(\theta(n) - a(n)\beta(n)\big). \tag{11}$$

Step 5 Repeat steps 1-4 with the new parameter $\theta(n+1)$, until $n = T$, where T is a large positive integer.

Remark 3.1 Note that a direct inversion of the Hessian update in the above algorithm is avoided through the use of the recursion (10) that will be seen to converge (as $\delta \to 0$) for a given parameter update to the product of the projected Hessian inverse with the gradient of the objective. A timescale separation between the θ-update (11) with that of the β-recursion (10) helps in this process.

Remark 3.2 We observe here as with [10, 14] that updating the Hessian $H(n)$ and the parameter $\theta(n)$ along the subsequence $\{nL\}$ of (sequence) $\{n\}$ of recursion epochs, for a given $L > 1$, actually improves performance. Thus, in between two successive updates of $H_{i,j}$, β and θ, the quantities Z^{--}, Z^{++} and Z^{o} are updated for L iterations. The convergence proof however works for any finite value of $L \geq 1$. It is generally observed that a value of L in between 50 and 500 usually works well. For our experiments, we chose $L = 100$.

Remark 3.3 One may define $\theta^{--}(n) := \theta(n) - \delta_1\Delta(n) - \delta_2\hat{\Delta}(n)$ and $\theta^{++}(n) := \theta(n) + \delta_1\Delta(n) + \delta_2\hat{\Delta}(n)$, respectively, for some $\delta_1, \delta_2 > 0$ that are not necessarily equal (see [13, 14]). The convergence analysis in such a case would carry through in the same manner as for the case of $\delta_1 = \delta_2 = \delta$ (below).

3.2 Algorithm 2

Our second algorithm directly updates the Hessian inverse through an application of the Sherman–Morrison lemma for matrix inversion that we derive here from the Woodbury identity. A similar procedure for a two-simulation algorithm (2SA) from [14] has been suggested in the context of discrete optimization in service systems (see Chap. 12 of [11]). For matrices A, B, L and D of dimensions $n \times n$, $n \times m$, $m \times m$ and $m \times n$, respectively, $n, m \geq 1$, the Woodbury's identity states that

$$(A + BLD)^{-1} = A^{-1} - A^{-1}B\big(L^{-1} + DA^{-1}B\big)^{-1}DA^{-1}. \tag{12}$$

Now recall the Hessian update (9) and note that the same can be rewritten as

$$H(n+1) = A(n) + B(n)L(n)D(n),$$

where

$$\begin{aligned} A(n) &:= \big(1 - c(n)\big)H(n), \\ B(n) &:= \frac{c(n)}{\delta}\left(\frac{1}{\Delta_1(n)}, \ldots, \frac{1}{\Delta_N(n)}\right)^T, \\ D(n) &:= \frac{1}{\delta}\left(\frac{1}{\hat{\Delta}_1(n)}, \ldots, \frac{1}{\hat{\Delta}_N(n)}\right), \end{aligned}$$

and

$$L(n) := \big(Z^{++}(nL) + Z^{--}(nL) - 2Z^{o}(nL)\big).$$

Letting $\check{M}(n) := H(n)^{-1}$ assuming $H(n)$ is positive definite, one obtains from (12) the following:

$$\begin{aligned} \check{M}(n+1) = {} & \frac{\check{M}(n)}{(1 - c(n))} \\ & - \frac{\check{M}(n)}{(1 - c(n))}c(n)B(n)\left(\frac{1}{L(n)} + \frac{D(n)\check{M}(n)}{(1 - c(n))}c(n)B(n)\right)^{-1}\frac{D(n)\check{M}(n)}{(1 - c(n))}. \end{aligned}$$

Upon simplification, one obtains

$$\left(\frac{1}{L(n)} + \frac{D(n)\check{M}(n)}{(1 - c(n))}c(n)B(n)\right)^{-1} = \big(1 - c(n)\big)\hat{L}(n),$$

where

$$\hat{L}(n) = \frac{L(n)}{(1 - c(n) + c(n)L(n)D(n)\check{M}(n)B(n))}.$$

Note that $\hat{L}(n)$ is a scalar. Thus, one obtains the following recursion for directly updating $\check{M}(n) = H(n)^{-1}$ and which corresponds to an application of the Sherman–Morrison lemma for matrix inversion in this case.

$$\check{M}(n+1) = \frac{\check{M}(n)}{(1-c(n))}\big(I - c(n)\hat{L}(n)B(n)D(n)\check{M}(n)\big), \tag{13}$$

where I denotes the identity matrix. In practice, $H(n)$ may not be positive definite, however, $\Gamma(H(n))$ will be so. Hence, we project $\check{M}(n)$ after each update to the set of positive definite and symmetric matrices using the operator Γ. Thus, $M(n) = \Gamma(\check{M}(n))$ will be positive definite and symmetric.

The overall sequence of steps in Algorithm 2 is similar to that of Algorithm 1. In fact, steps 1 and 2 in Algorithm 2 are exactly the same as in Algorithm 1, and are not being described to prevent repetition. However, steps 3 and 4 in Algorithm 2 are considerably different from Algorithm 1 and are described below.

Step 3 Update the estimate $M(n)$ of the Hessian inverse on the medium timescale as follows: For $i, j = 1, \ldots, N$,

$$\check{M}(n+1) = \frac{\check{M}(n)}{(1-c(n))}\big(I - c(n)\hat{L}(n)B(n)D(n)\check{M}(n)\big), \tag{14}$$

with $\check{M}(n)$, $\hat{L}(n)$, $B(n)$, $D(n)$ as above. Next, set $M(n) = \Gamma(\check{M}(n))$. Finally, we have

Step 4 Update the parameter $\theta(n)$ on the slowest timescale using the Hessian inverse estimate $M(n)$ as well as the average cost estimates $Z^{++}(\cdot)$, $Z^{--}(\cdot)$ corresponding to the parameters $\theta^{++}(\cdot)$, $\theta^{--}(\cdot)$, respectively, as follows:

$$\theta(n+1) = \Pi\left(\theta(n) - a(n)M(n)\left(\frac{Z^{++}(nL) - Z^{--}(nL)}{2\delta}\big(\Delta(n)\big)^{-1}\right)\right). \tag{15}$$

Note that Remarks 3.2 and 3.3 also hold in the case of Algorithm 2.

4 Convergence Analysis

Both our algorithms involve multi-timescale stochastic approximation. We follow the ODE approach for their analysis. Chapter 6 of [20] contains a general treatment of multi-timescale algorithms using the ODE approach.

4.1 Convergence Analysis of Algorithm 1

We begin with an analysis of recursions (6)–(8). First note that since $h(\cdot)$ is Lipschitz continuous,

$$\big|h(x)\big| \leq \big|h(x) - h(0)\big| + \big|h(0)\big| \leq K\|x\| + \big|h(0)\big| \leq \hat{K}\big(1 + \|x\|\big),$$

where $K > 0$ is the Lipschitz constant and $\hat{K} = \max(K, |h(0)|)$. For $n \geq 0$, let $\hat{b}(n) := b([\frac{n}{L}])$, where $[\frac{n}{L}]$ denotes the integer part of $\frac{n}{L}$. Note that the requirements in (3)–(4) continue to hold when the sequence $b(n), n \geq 0$ is replaced with $\hat{b}(n), n \geq 0$. One can now rewrite (6)–(8) as follows: For $w \in \{--, ++, o\}$, $n \geq 0$,

$$Z^w(n+1) = Z^w(n) + \hat{b}(n)\big(h\big(X^w(n)\big) - Z^w(n)\big). \tag{16}$$

For $nL \leq l < (n+1)L$, let $\breve{\theta}(l) := \theta(n)$, $\breve{\Delta}(l) := \Delta(n)$ and $\breve{\hat{\Delta}}(l) := \hat{\Delta}(n)$. Let

$$\mathcal{F}(l) := \sigma\big(\breve{\theta}(p), \breve{\Delta}(p), \breve{\hat{\Delta}}(p), X_p^w,\ p \leq l,\ w \in \{--, ++, o\}\big), \quad l \geq 1,$$

denote a sequence of σ-fields. Consider sequences $\{M^w(p), p \geq 1\}$, $w \in \{--, ++, o\}$, that are defined as follows:

$$M^w(p) := \sum_{m=1}^{p} \hat{b}(m)\big(h\big(X_m^w\big) - E\big[h\big(X_m^w\big) \mid \mathcal{F}(m-1)\big]\big) = \sum_{m=1}^{p} \hat{b}(m) N^w(m),$$

where $N^w(m) := h(X_m^w) - E[h(X_m^w) \mid \mathcal{F}(m-1)]$, $m \geq 1$ is the corresponding martingale difference sequence.

Lemma 4.1 *The sequences* $\{(M^w(p), \mathcal{F}(p))\}$, $w \in \{--, ++, o\}$ *are almost surely convergent martingale sequences.*

Proof It is easy to see that $\{M^w(p), p \geq 1\}$, $w \in \{--, ++, o\}$ are martingale sequences. Now note that

$$\begin{aligned}
&\sum_p E\big[\big(M^w(p+1) - M^w(p)\big)^2 \mid \mathcal{F}(p)\big] \\
&\quad = \sum_p \hat{b}^2(p+1)\big(E\big[\big(h\big(X_{p+1}^w\big) - E\big[h\big(X_{p+1}^w\big) \mid \mathcal{F}(p)\big]\big)^2 \mid \mathcal{F}(p)\big]\big) \\
&\quad \leq K_0 \sum_p \hat{b}^2(p+1) E\big[\big(1 + \|X_{p+1}^w\|^2\big) \mid \mathcal{F}(p)\big],
\end{aligned}$$

almost surely, for some constant $K_0 > 0$. As a consequence of Assumption 2.2, $\sup_p E[\|X_{p+1}^w\|^2 \mid \mathcal{F}(p)] < \infty$ almost surely. Further, from the square summability of $\hat{b}(p)$ (cf. (3)), it follows that

$$\sum_p E\big[\big(M^w(p+1) - M^w(p)\big)^2 \mid \mathcal{F}(p)\big] < \infty \quad \text{a.s.}$$

Now by the martingale convergence theorem (cf. Theorem 3.3.4, pp. 53–54 of [21]), $\{M^w(p), p \geq 1\}$ are almost surely convergent sequences. □

Let $s(n) := \sum_{i=0}^{n-1} a(i)$, $n \geq 1$, with $s(0) := 0$. Define $\theta(t), t \in [0, \infty[$ as follows: $\theta(s(n)) := \theta(n), n \geq 0$ and for $t \in]s(n), s(n+1)[$, $\theta(t)$ is obtained from the continuous linear interpolation of $\theta(s(n))$ and $\theta(s(n+1))$. Also, define $\Delta(t), \hat{\Delta}(t), t \geq 0$

as follows: $\Delta(t) := \Delta(n)$ and $\hat{\Delta}(t) := \hat{\Delta}(n)$, for $t \in [s(n), s(n+1)[$, $n \geq 0$. Let $t(n) := \sum_{m=0}^{n-1} \hat{b}(m)$, $n \geq 1$ with $t(0) := 0$. Let $Z^w(t(n)) := Z^w(n)$ with linear interpolation on $[t(n), t(n+1)]$.

Consider now the following system of ODEs:

$$\dot{Z}^w(t) = J\big(\theta^w(t)\big) - Z^w(t), \quad w \in \{--, ++, o\}, \tag{17}$$

$$\dot{H}_{i,j}(t) = 0, \quad i, j \in \{1, \ldots, N\}, \dot{\beta}(t) = 0, \dot{\theta}(t) = 0. \tag{18}$$

In the light of (18), one may let $\theta^w(t) := \theta^w$ (i.e., independent of t). In such a case, (17) can be rewritten as

$$\dot{Z}^w(t) = J\big(\theta^w\big) - Z^w(t), \quad w \in \{--, ++, o\}. \tag{19}$$

Proposition 4.1 *We have $\|Z^w(n) - J(\theta^w(n))\| \to 0$ as $n \to \infty$ with probability one.*

Proof Rewrite recursions (9), (10) and (11) as follows: For $i, j = 1, \ldots, N$, $\forall n \geq 0$,

$$H_{i,j}(n+1) = H_{i,j}(n) + b(n)\frac{c(n)}{b(n)}\left(\frac{Z^{++}(nL) + Z^{--}(nL) - 2Z^o(nL)}{\delta^2 \Delta_i(n)\hat{\Delta}_j(n)} - H_{i,j}(n)\right), \tag{20}$$

$$\beta(n+1) = \beta(n) + b(n)\frac{c(n)}{b(n)}\left(-P(n)\beta(n) + \left(\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta}\big(\Delta(n)\big)^{-1}\right)\right), \tag{21}$$

$$\theta(n+1) = \Pi\left(\theta(n) - b(n)\frac{a(n)}{b(n)}\beta(n)\right). \tag{22}$$

Now since both $c(n)=o(b(n))$ and $a(n)=o(b(n))$, it is easy to see that $(H_{i,j}(t(n)+\cdot)$, $\beta(t(n)+\cdot)$, $\theta(t(n)+\cdot)$, $i, j = 1, \ldots, N)$ is a noisy Euler discretization of the ODEs (18) for large n. Thus, one may let (as a consequence of the above), $\theta^w(n) := \theta^w$ (i.e., invariant of n) for n sufficiently large. Now note that (16) can be rewritten as

$$Z^w(n+1) = Z^w(n) + \hat{b}(n)\big(J\big(\theta^w\big) + \xi_1(n) + N^w(n) - Z^w(n)\big), \tag{23}$$

where $\xi_1(n) = E[h(X^w(n)) \mid \mathcal{F}(n-1)] - J(\theta^w)$. In fact, $\xi_1(n) \to 0$ almost surely as $n \to \infty$ along the 'natural timescale'. This follows from the ergodicity of $X^w(n)$, $n \geq 0$, given θ^w (see Theorem 7, Corollary 8, p. 74 and Theorem 9, p. 75 in [20]). From Assumption 2.2, it follows that $\sup_n E[h(X^w(n)) \mid \mathcal{F}(n-1)] < \infty$ almost surely. Further, from Assumption 2.1 and the fact that $\theta^w \in C$, a compact set, it follows that $\sup_{\theta^w \in C} |J(\theta^w)| < \infty$. Thus, $\sup_n |\xi_1(n)| < \infty$ almost surely. Now $\hat{b}(n) \to 0$ as $n \to \infty$. Further, in lieu of the foregoing, $\sup_{\theta^w, n} |J(\theta^w) + \xi_1(n) + N^w(n)| < \infty$ with probability one. Thus, $\exists N_1 > 0$ such that for $n \geq N_1$, $Z^w(n+1)$ is a convex combination of $Z^w(n)$ and an almost surely uniformly bounded quantity. Thus, $\sup_n |Z^w(n)| < \infty$ almost surely and (23) can be seen to be a noisy Euler discretization (with diminishing noise) of (19). The claim now follows from Theorem 1 of [22, p. 339]. □

Now note that the recursion (9) can be rewritten as follows: $\forall i, j \in \{1, 2, \ldots, N\}$,

$$H_{i,j}(n+1) = \big(1 - c(n)\big) H_{i,j}(n) + c(n)\left(E\left[\frac{\hat{J}(\theta(n), \Delta(n), \hat{\Delta}(n))}{\delta^2 \triangle_i(n) \hat{\triangle}_j(n)} \,\middle|\, \mathcal{F}(n) \right] + \xi_2(n) + N_{i,j}(n) \right), \quad (24)$$

where

$$\hat{J}\big(\theta(n), \Delta(n), \hat{\Delta}(n)\big) := J\big(\theta(n) + \delta\Delta(n) + \delta\hat{\Delta}(n)\big) + J\big(\theta(n) - \delta\Delta(n) - \delta\hat{\Delta}(n)\big) - 2J\big(\theta(n)\big),$$

$$\xi_2(n) := \frac{(Z^+(nL) + Z^-(nL) - 2Z^o(nL))\triangle_i(n)\hat{\triangle}_j(n) - \hat{J}(\theta(n), \Delta(n), \hat{\Delta}(n))}{\delta^2 \triangle_i(n) \hat{\triangle}_j(n)},$$

$$N_{i,j}(n) := \frac{\hat{J}(\theta(n), \Delta(n), \hat{\Delta}(n))}{\delta^2 \triangle_i(n) \hat{\triangle}_j(n)} - E\left[\frac{\hat{J}(\theta(n), \Delta(n), \hat{\Delta}(n))}{\delta^2 \triangle_i(n) \hat{\triangle}_j(n)} \,\middle|\, \mathcal{F}(n-1) \right].$$

Proposition 4.2 *For all $i, j \in \{1, \ldots, N\}$,*

$$\lim_{\delta \to 0} \left| E\left[\frac{\hat{J}(\theta(n), \Delta(n), \hat{\Delta}(n))}{\delta^2 \triangle_i(n) \hat{\triangle}_j(n)} \,\middle|\, \mathcal{F}(n) \right] - \nabla^2_{i,j} J\big(\theta(n)\big) \right| = 0 \quad a.s.$$

Proof We first consider the case when $i, j \in \{1, \ldots, N\}$, $i \neq j$. It is easy to see from suitable Taylor's expansions that

$$\begin{aligned}
&\frac{\hat{J}(\theta(n), \Delta(n), \hat{\Delta}(n))}{\delta^2 \triangle_i(n) \hat{\triangle}_j(n)} \\
&= \frac{(\Delta(n) + \hat{\Delta}(n))^T \nabla^2 J(\theta(n))(\Delta(n) + \hat{\Delta}(n))}{\triangle_i(n) \hat{\triangle}_j(n)} + o(\delta) \\
&= \sum_{l=1}^{N} \sum_{m=1}^{N} \frac{\Delta_l(n) \nabla^2_{lm} J(\theta(n)) \Delta_m(n)}{\Delta_i(n) \hat{\Delta}_j(n)} + 2 \sum_{l=1}^{N} \sum_{m=1}^{N} \frac{\Delta_l(n) \nabla^2_{lm} J(\theta(n)) \hat{\Delta}_m(n)}{\Delta_i(n) \hat{\Delta}_j(n)} \\
&\quad + \sum_{l=1}^{N} \sum_{m=1}^{N} \frac{\hat{\Delta}_l(n) \nabla^2_{lm} J(\theta(n)) \hat{\Delta}_m(n)}{\Delta_i(n) \hat{\Delta}_j(n)} + o(\delta).
\end{aligned}$$

It is now easy to see that

- $E[\sum_{l=1}^{N} \sum_{m=1}^{N} \frac{\Delta_l(n) \nabla^2_{lm} J(\theta(n)) \Delta_m(n)}{\Delta_i(n) \hat{\Delta}_j(n)} \mid \mathcal{F}(n)]$
  $= E[\sum_{l=1}^{N} \sum_{m=1}^{N} \frac{\hat{\Delta}_l(n) \nabla^2_{lm} J(\theta(n)) \hat{\Delta}_m(n)}{\Delta_i(n) \hat{\Delta}_j(n)} \mid \mathcal{F}(n)] = 0$ a.s.,
- $E[\sum_{l=1}^{N} \sum_{m=1}^{N} \frac{\Delta_l(n) \nabla^2_{lm} J(\theta(n)) \hat{\Delta}_m(n)}{\Delta_i(n) \hat{\Delta}_j(n)} \mid \mathcal{F}(n)] = \nabla^2_{i,j} J(\theta(n))$ a.s..

Thus,

$$E\left[\frac{\hat{J}(\theta(n),\Delta(n),\hat{\Delta}(n))}{\delta^2\Delta_i(n)\hat{\Delta}_j(n)}\,\middle|\,\mathcal{F}(n)\right]=2\nabla^2_{i,j}J\big(\theta(n)\big)+o(\delta).$$

The case when $i=j$, $i,j\in\{1,\ldots,N\}$ follows in a similar manner. The claim follows. □

Consider now the following system of ODEs: $\forall i,j=1,\ldots,N$,

$$\dot{H}_{i,j}(t)=\nabla^2_{i,j}J\big(\theta(t)\big)-H_{i,j}(t),\quad \dot{\theta}(t)=0. \tag{25}$$

In lieu of (25), one may let $\theta(t):=\theta$ (invariant of t) as before. We now have the following result.

Lemma 4.2 *We have* $|H_{i,j}(n)-\nabla^2_{i,j}J(\theta(n))|\to 0$ *as* $n\to\infty$ *and* $\delta\to 0$ *with probability one.*

Proof Recall that the Hessian update (9) can be rewritten as (24). It has been shown in Proposition 4.1 that $\sup_n|Z^w(n)|<\infty$ a.s., $\forall w\in\{--,++,o\}$. Further, $\sup_{\theta\in C}J(\theta)<\infty$ by Assumption 2.1. Thus, $\sup_n|\xi_2(n)|<\infty$ almost surely. Further, $\xi_2(n)\to 0$ as $n\to\infty$ by Proposition 4.1. Also, as with Lemma 4.1, it is easy to see that $\sum_{l=1}^n c(l)N_{i,j}(l)$, $n\geq 0$ is an almost surely convergent martingale sequence. Hence, it follows that $\sup_n|H_{i,j}(n)|<\infty$ a.s. and that $N_{i,j}(n)=o(1)$ as well. The claim now follows Theorem 1 of [22, p. 339]. □

Corollary 4.1 $\|P(n)-\Gamma(\nabla^2J(\theta(n)))\|\to 0$ *as* $n\to\infty$ *and* $\delta\to 0$ *with probability one.*

Proof Follows from Lemma 4.2 and Assumption 2.3. □

Consider now the following N sequences: For $i=1,\ldots,N$,

$$\chi^i_4(n):=\sum_{m=0}^{n-1}c(m)\left(\frac{Z^{++}(mL)-Z^{--}(mL)}{2\delta\Delta_i(m)}-E\left[\frac{Z^{++}(mL)-Z^{--}(mL)}{2\delta\Delta_i(m)}\,\middle|\,\mathcal{F}(m)\right]\right),$$

$n\geq 1$.

Lemma 4.3 *Each of the sequences* $(\chi^i_4(n),\mathcal{F}(n))$, $n\geq 1$, $i=1,\ldots,N$ *is an almost surely convergent zero-mean martingale.*

Proof It is easy to see that $(\chi^i_4(n),\mathcal{F}(n))$ are zero-mean martingales. Now $\sup_n|Z^w(nL)|<\infty$ a. s. $\forall w$ (see proof of Proposition 4.1). Further, from the square summability condition on the step-sizes $c(n)$, $n\geq 0$ in (3), it can be seen that the quadratic variation processes of each of these martingales are almost surely convergent. The claim now follows as a consequence of the martingale convergence theorem (cf. [21, Theorem 3.3.4, pp. 53–54]) applied individually to each martingale. □

Lemma 4.4

$$\left\| E\left[\frac{Z^{++}(nL) - Z^{--}(nL)}{2\delta} \big(\Delta(n)\big)^{-1} \,\middle|\, \mathcal{F}(n) \right] - \nabla J\big(\theta(n)\big) \right\| \to 0,$$

as $n \to \infty$ *and* $\delta \to 0$ *with probability one.*

Proof Note that

$$\begin{aligned}
&\left\| E\left[\frac{Z^{++}(nL) - Z^{--}(nL)}{2\delta} \big(\Delta(n)\big)^{-1} \,\middle|\, \mathcal{F}(n) \right] - \nabla J\big(\theta(n)\big) \right\| \\
&\quad \le \left\| E\left[\frac{Z^{+}(nL) - Z^{-}(nL)}{2\delta} \big(\Delta(n)\big)^{-1} \,\middle|\, \mathcal{F}(n) \right] \right. \\
&\qquad \left. - E\left[\frac{J(\theta(n) + \delta\Delta(n) + \delta\hat{\Delta}(n)) - J(\theta(n) - \delta\Delta(n) - \delta\hat{\Delta}(n))}{2\delta} \big(\Delta(n)\big)^{-1} \,\middle|\, \mathcal{F}(n) \right] \right\| \\
&\qquad + \left\| E\left[\frac{J(\theta(n) + \delta\Delta(n) + \delta\hat{\Delta}(n)) - J(\theta(n) - \delta\Delta(n) - \delta\hat{\Delta}(n))}{2\delta} \big(\Delta(n)\big)^{-1} \,\middle|\, \mathcal{F}(n) \right] \right. \\
&\qquad \left. - \nabla J\big(\theta(n)\big) \right\|.
\end{aligned}$$

The term given by the first norm on the RHS above vanishes asymptotically with probability one as $n \to \infty$ by Proposition 4.1. From suitable Taylor's expansions of $J(\theta(n) + \delta\Delta(n) + \delta\hat{\Delta}(n))$ and $J(\theta(n) - \delta\Delta(n) - \delta\hat{\Delta}(n))$ around $\theta(n)$, it can be seen that

$$\begin{aligned}
&E\left[\frac{J(\theta(n) + \delta\Delta(n) + \delta\hat{\Delta}(n)) - J(\theta(n) - \delta\Delta(n) - \delta\hat{\Delta}(n))}{2\delta\Delta_i(n)} \,\middle|\, \mathcal{F}(n) \right] \\
&\quad = \nabla_i J\big(\theta(n)\big) + o(\delta).
\end{aligned}$$

The claim now follows. □

We consider now the recursion (10).

Proposition 4.3 *We have*

$$\left\| \beta(n) - \Gamma\big(\nabla^2 J\big(\theta(n)\big)\big)^{-1} \nabla J\big(\theta(n)\big) \right\| \to 0,$$

as $\delta \to 0$ *and* $n \to \infty$, *a.s.*

Proof The proof relies on an important result by Borkar and Meyn (cf. Theorems 2.1–2.2 of [23]; see also Theorem D.1 of [11, p. 296]). Note that (10) can be rewritten as

$$\beta(n+1) = \beta(n) + c(n)\big(-\Gamma\big(\nabla^2 J\big(\theta(n)\big)\big)\beta(n) - \nabla J\big(\theta(n)\big) + \xi_3(n) + \xi_4(n) + \xi_5(n)\big), \tag{26}$$

where

$$\xi_3(n) := (\Gamma(\nabla^2 J(\theta(n))) - P(n))\beta(n),$$

$$\xi_4(n) := \Bigg(-\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta}\big(\Delta(n)\big)^{-1} + E\bigg[\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta}\big(\Delta(n)\big)^{-1}\,\bigg|\,\mathcal{F}(n)\bigg]\Bigg),$$

$$\xi_5(n) := \Bigg(-E\bigg[\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta}\big(\Delta(n)\big)^{-1}\,\bigg|\,\mathcal{F}(n)\bigg] + \nabla J\big(\theta(n)\big)\Bigg).$$

Now note that $\xi_3(n) \to 0$ as $n \to \infty$ and $\delta \to 0$ from Corollary 4.1. Further, $\xi_5(n) \to 0$ as $n \to \infty$ and $\delta \to 0$, as a consequence of Lemma 4.4. Let

$$\xi_4^i(n) := -\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta\Delta_i(n)} + E\bigg[\frac{Z^{--}(nL) - Z^{++}(nL)}{2\delta\Delta_i(n)}\,\bigg|\,\mathcal{F}(n)\bigg].$$

Then, $\xi_4(n) = (\xi_4^i(n), i = 1, \ldots, N)^T$. Further, from Lemma 4.3, $\sum_n c(n)\xi_4^i(n) < \infty$ almost surely. Consider now the following system of ODEs:

$$\dot{\beta}(t) = -\Gamma\big(\nabla^2 J\big(\theta(t)\big)\big)\beta(t) + \nabla J\big(\theta(t)\big), \quad \dot{\theta}(t) = 0. \tag{27}$$

As a consequence of the second ODE in (27), $\theta(t) := \theta$ (i.e., a time-invariant quantity). Now note that $g(\beta) := -\Gamma(\nabla^2 J(\theta))\beta + \nabla J(\theta)$ is Lipschitz continuous in β. Further, $g_\infty(\beta) := \lim_{r\to\infty} \frac{g(r\beta)}{r} = -\Gamma(\nabla^2 J(\theta))\beta$. Since $\Gamma(\nabla^2 J(\theta))$ is a positive definite matrix, the ODE $\dot{\beta}(t) = g_\infty(\beta(t))$ has the origin as its unique globally asymptotically stable equilibrium. Thus, assumptions (A1) and (A2) of [23] hold true. From Theorem 2.1 of [23], $\sup_n \|\beta(n)\| < \infty$ almost surely and the claim follows from Theorem 2.2 of [23]. □

Finally, we consider the recursion (11). The ODE associated with (11) is

$$\dot{\theta}(t) = \hat{\Pi}\big(-\Gamma\big(\nabla^2 J\big(\theta(t)\big)\big)^{-1}\nabla J\big(\theta(t)\big)\big), \tag{28}$$

where the operator $\hat{\Pi}(\cdot)$ is defined as follows: for any bounded and continuous function $v : \mathbb{R} \to \mathbb{R}$,

$$\hat{\Pi}_i\big(v(y)\big) := \lim_{\eta\to 0}\bigg(\frac{\Pi_i(y + \eta v(y)) - \Pi_i(y)}{\eta}\bigg), \quad i = 1, \ldots, N.$$

Further, for $x = (x_1, \ldots, x_N)^T$, $\hat{\Pi}(x) := (\hat{\Pi}_1(x_1), \ldots, \hat{\Pi}_N(x_N))^T$.

Let $K := \{\theta \in C \mid \hat{\Pi}(-\Gamma(\nabla^2 J(\theta))^{-1}\nabla J(\theta)) = 0\}$. For given $\eta > 0$, let K^η denote the set of points in a η-neighbourhood of the set K, i.e., $K^\eta := \{\theta \in C \mid \|\theta - \theta_0\| < \eta,\ \theta_0 \in K\}$.

Theorem 4.1 *Under Assumptions 2.1–2.3, 3.1 and 3.2, given $\eta > 0$, there exists $\hat{\delta} > 0$ such that for all $\delta \in]0, \hat{\delta}[$, $\theta(n) \to K^\eta$, as $n \to \infty$, with probability one.*

Proof Note that (11) can be rewritten as

$$\theta(n+1) = \Pi\big(\theta(n) - a(n)\Gamma\big(\nabla^2 J\big(\theta(n)\big)\big)^{-1}\nabla J\big(\theta(n)\big) + a(n)\hat{\alpha}(n)\big), \tag{29}$$

where $\hat{\alpha}(n) = (\Gamma(\nabla^2 J(\theta(n)))^{-1}\nabla J(\theta(n)) - \beta(n))$. By Proposition 4.3, $\|\hat{\alpha}(n)\| \to 0$ with probability one as $n \to \infty$. Further, since $\theta(n) \in C$ (a compact set) for all n and by Assumptions 2.1 and 2.3, it follows that $\sup_n \|\Gamma(\nabla^2 J(\theta(n)))^{-1}\nabla J(\theta(n))\| < \infty$ almost surely. Also, $\sup_n \|\beta(n)\| < \infty$ almost surely as well (see proof of Proposition 4.3). It thus follows that $\sup_n \|\hat{\alpha}(n)\| < \infty$ almost surely as well. Now using a similar argument as in Theorem 5.3.1 of [24, pp. 191–196] (see also Theorem E.1 in [11, p. 298]), (29) can be seen to be a noisy Euler discretization of the ODE (28). It is easy to see that $J(\cdot)$ itself serves as an associated Lyapunov function for (28) since

$$\frac{dJ(\theta)}{d\theta} = \nabla J(\theta)^T\dot{\theta} = \nabla J(\theta)^T\hat{\Pi}\big(-\Gamma\big(\nabla^2 J(\theta)\big)^{-1}\nabla J(\theta)\big) \le 0.$$

Note that $J(\cdot)$ is continuous and hence uniformly bounded on the compact set C. Let $\mu = \sup_\theta J(\theta) < \infty$. Then $\{\theta \mid J(\theta) \le \mu\} = C$. The claim now follows from Lasalle's invariance theorem, see [25], also stated as Theorem 2.3 in [26, p. 76]. □

Remark 4.1 Let $R := \{\theta \in C \mid \hat{\Pi}(-\Gamma(\nabla^2 J(\theta))^{-1}\nabla J(\theta)) = -\Gamma(\nabla^2 J(\theta))^{-1}\nabla J(\theta)\}$. Note that $C^0 \subseteq R$. Further, for $\theta \in R$, $\frac{dJ(\theta)}{dt} < 0$ if $\nabla J(\theta) \neq 0$. For $\theta \in R \cap K$, $\nabla J(\theta) = 0$ since $\Gamma(\nabla^2 J(\theta))^{-1}$ is positive definite and symmetric. In addition, there may be spurious fixed points within K that would however lie in ∂C (the boundary of C), see [26, p. 79]. Now note that $S := \{\theta \in C \mid \nabla J(\theta) = 0\}$ constitutes the set of all Kuhn–Tucker points which includes local minima as well as unstable equilibria. Any stochastic approximation scheme in general may get trapped in an unstable equilibrium (see [20, Chap. 4.3]). In most practical scenarios (as with our experiments), because of the inherent randomness in the parameter updates, stochastic approximation algorithms are seen to converge to stable equilibria. Finally, the choice of the parameter δ has a bearing on the performance of algorithm. A very low δ has the effect of increasing the variability in the estimates while a large δ leads to inaccuracies resulting from the bias terms, see Sect. 5 for detailed experiments with various values of δ.

4.2 Convergence Analysis of Algorithm 2

Recall that the recursions (6)–(8) for averaging the cost functions corresponding to different parameters, are the same for both Algorithms 1 and 2. Thus, the conclusions of Proposition 4.1 continue to hold.

Lemma 4.5 *We have* $\|M(n) - \Gamma(\nabla^2 J(\theta(n)))^{-1}\| \to 0$ *as* $n \to \infty$ *with probability one.*

Proof Recall that the update of $\check{M}(n)$ (cf. (14)) is obtained via the Hessian update (9), see the calculation leading up to (13). From Lemma 4.2, we have

$$\big\|H(n) - \nabla^2 J\big(\theta(n)\big)\big\| \to 0,$$

as $n \to \infty$ almost surely. The claim follows as a consequence of Assumption 2.3 and the facts that $\breve{M}(n) = H(n)^{-1}$ and $M(n) = \Gamma(\breve{M}(n))$. □

Consider now the recursion (15). The ODE associated with (15) is

$$\dot{\theta}(t) = \hat{\Pi}\big(-\Gamma\big(\nabla^2 J\big(\theta(t)\big)^{-1}\big)\nabla J\big(\theta(t)\big)\big), \tag{30}$$

with $\hat{\Pi}(\cdot)$ defined as before. Let $\breve{K} := \{\theta \in C \mid \nabla J(\theta)^T \hat{\Pi}(-\Gamma(\nabla^2 J(\theta)^{-1})\nabla J(\theta)) = 0\}$. For given $\eta > 0$, let $\breve{K}^\eta := \{\theta \in C \mid \|\theta - \theta_0\| < \eta,\ \theta_0 \in \breve{K}\}$.

Theorem 4.2 *Under Assumptions 2.1–2.3, 3.1 and 3.2, given $\eta > 0$, there exists $\hat{\delta} > 0$ such that for all $\delta \in]0, \hat{\delta}[$, $\theta(n) \to \breve{K}^\eta$, as $n \to \infty$, with probability one.*

Proof Follows in a similar manner as the proof of Theorem 4.1. □

Finally, the conclusions in Remark 4.1 continue to hold in the case of Algorithm 2 as well. This completes the convergence analysis of both algorithms.

5 Numerical Experiments

We test the performance of our algorithms on a problem of traffic light control. The problem here is to dynamically adapt the sign configurations, i.e., the specification of green lights for the signalled lanes at traffic junctions in road networks, so that the traffic flow is maximized in the long-run average sense. This problem has been considered in detail in [15, 16], with the difference that the threshold parameter in our case has a higher dimension as we consider feedback policies with more levels than those considered in the aforementioned references. The state s_n at instant n is the vector of queue lengths and elapsed times on the signalled lanes of the road network considered. Here, elapsed time on a lane refers to the time elapsed after the signal turned to red on that lane. By using thresholds, we obtain rough estimates of the congestion levels as well as the time for which any lane has not received the green light. It should be noted that obtaining precise queue length information is often not possible and one needs to work with coarse congestion levels.

The single-stage cost is designed to balance efficiency (reduce the queue lengths) and fairness (no lane receives the red signal for a prolonged period) and is given by

$$\begin{aligned} c(s_n) := \alpha_1 * \Bigg(\sum_{i \in I_p} \alpha_2 * \tilde{q}_i(n) + \sum_{i \notin I_p} \beta_2 * \tilde{q}_i(n)\Bigg) \\ + \beta_1 * \Bigg(\sum_{i \in I_p} \alpha_2 * \tilde{t}_i(n) + \sum_{i \notin I_p} \beta_2 * \tilde{t}_i(n)\Bigg), \end{aligned} \tag{31}$$

where $\alpha_i, \beta_i \geq 0$ and $\alpha_i + \beta_i = 1, i = 1, 2$ and $\alpha_2 > \beta_2$. Here, I_p denotes the set of high priority lanes. The cost function thus assigns higher weight to delays and elapsed times on high priority lanes over lanes with low priority. For instance, high

Table 1 Priority assignment for each lane in the TLC policy

(**a**) Queue priority value based on q_i

Condition	Queue priority value
$q_i(n) < L_1$	1
$L_1 \le q_i(n) < L_2$	2
$L_2 \le q_i(n) < L_3$	3
$L_3 \le q_i(n) < L_4$	4
$L_4 \le q_i(n) < L_5$	5
$q_i(n) \ge L_5$	6

(**b**) Elapsed priority value based on t_i

Condition	Elapsed priority value
$t_i(n) < T_1$	1
$T_1 \le t_i(n) < T_2$	2
$T_2 \le t_i(n) < T_3$	3
$t_i(n) \ge T_3$	4

priority lanes could be those on the main road, while low priority lanes could correspond to lanes on the side roads. The parameter on which the state depends is $\theta := (L_1, L_2, L_3, L_4, L_5, T_1, T_2, T_3)^{\mathsf{T}}$.[1] We use the following as the state feature that coarsely describes the state:

$$s_n(\theta) := \big(\tilde{q}_1(n), \ldots, \tilde{q}_K(n), \tilde{t}_1(n), \ldots, \tilde{t}_K(n)\big), \tag{32}$$

where

$$\tilde{q}_i(n) := \begin{cases} 0, & \text{if } q_i(n) < L_1 \\ 0.2, & \text{if } L_1 \le q_i(n) \le L_2 \\ 0.4, & \text{if } L_2 \le q_i(n) \le L_3 \\ 0.6, & \text{if } L_3 \le q_i(n) \le L_4 \\ 0.8, & \text{if } L_4 \le q_i(n) \le L_5 \\ 1, & \text{if } q_i(n) > L_5 \end{cases} \quad \text{and} \quad \tilde{t}_i(n) := \begin{cases} 0, & \text{if } t_i(n) \le T_1 \\ 0.33, & \text{if } T_1 \le t_i(n) \le T_2 \\ 0.66, & \text{if } T_2 \le t_i(n) \le T_3 \\ 1, & \text{if } t_i(n) > T_3. \end{cases} \tag{33}$$

Here $L_1 < L_2 < L_3 < L_4 < L_5$ and similarly $T_1 < T_2 < T_3$. Further, $q_i(n)$ and $t_i(n)$ are the queue length on lane i and the time elapsed since the signal turned red on lane i, respectively, at instant n. At each time epoch, a decision on which sign configuration (from the set of feasible sign configurations in the given state) to switch next is made based on a priority assignment scheme described in Table 1.

[1] Note that, unlike [15, 16], we include more thresholds in deciding the congestion level on a lane in the network.

A traffic light control algorithm based on graded thresholds was proposed in [16]. This algorithm, called PTLC, decided on the sign configuration by first assigning a priority to each lane and then choosing a sign configuration that maximized the sum of priorities of all the signalled lanes, among all possible sign configurations. We adapt the PTLC algorithm to incorporate more thresholds. The priority for a lane i is simply the product of the queue priority value and the elapsed priority value calculated from Tables 1(a) and 1(b) based on the queue length q_i and elapsed time t_i on lane i. The lanes in the sign configuration that receives the highest priority value are switched to green at the next decision epoch.

The choice of thresholds is crucial as it depends on the road network considered as well as the traffic dynamics. Thus, any given choice of thresholds need not be optimal for all network settings. We combine the two algorithms proposed in Sect. 3 with the PTLC feedback scheme using multi-timescale stochastic approximation as described below.

- On the faster timescale, we obtain cost estimates Z^{--}, Z^{++} and Z^{o} by simulating the traffic with the sign configurations chosen by the PTLC algorithm and the threshold parameter θ^{--}, θ^{++} and θ^{o}, respectively.
- On the slower timescale, we use these quantities to estimate the gradient of the long-run average cost $J(\theta)$ and tune the threshold parameter θ in the descent direction using a Newton step.

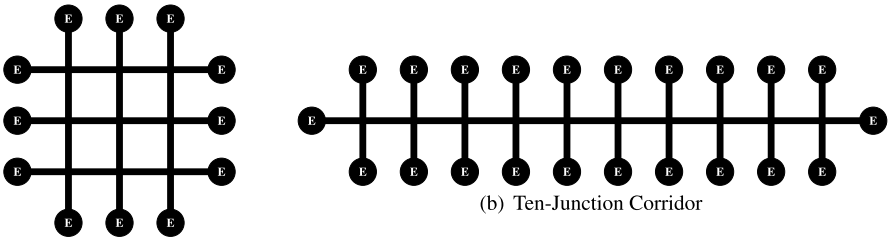
The step-size sequences were chosen according to $a(n) = \frac{1}{n}$, $c(n) = \frac{1}{n^{0.75}}$, and $b(n) = \frac{1}{n^{0.66}}$, $n \geq 1$, respectively. Further, we use symmetric, ± 1-valued Bernoulli random variables that are independent and identically distributed (with $p = \frac{1}{2}$) for perturbing the parameter θ in our experiments. The aforementioned choices of the step-sizes and the perturbations are seen to satisfy Assumptions 3.1 and 3.2.

Henceforth, we shall refer to the combination of PTLC with Algorithm 1 as PTLC-H, while that of PTLC with Algorithm 2 will be referred to as PTLC-W. Further, for the sake of comparison, we also implement the Newton-based 3SA and 4SA algorithms of [14] that have been found to perform well in practice. In fact, these algorithms (3SA and 4SA) have been found in [14] to show the best results there. We shall refer to the combination of these algorithms with PTLC as PTLC-4SA and PTLC-3SA, respectively.

For projecting the Hessian matrix $H(n)$ onto the space of positive definite and symmetric matrices, we first perform an eigenvalue decomposition of $H(n)$, project each eigenvalue onto $[\eta, \frac{1}{\eta}]$, where $0 < \eta < 1$ is a small number and reconstruct $H(n)$ using these projected eigenvalues. We set $\eta = 0.15$ in our experiments. The matrix $\check{M}$ in Algorithm 2 is projected in a similar fashion. The requirements in Assumption 2.3 are seen to be met with this choice of the projection operator.

We used the open source 'Green Light District (GLD)' software for our simulations. We conducted our experiments on the following road networks:

(i) a 3×3-grid network (Fig. 1a),
(ii) a corridor with ten junctions (Fig. 1b), and
(iii) a network modelled after 9 junctions around the Indian Institute of Science campus in Bangalore. A Google map for this (last) network can be found in Fig. 1(c) of [17].



(a) 3x3-Grid Network

(b) Ten-Junction Corridor

Fig. 1 Road Networks used for our experiments

Table 2 Comparison of PTLC algorithms using $\hat{J}(\theta)$

	3×3-Grid network	IISc network	Ten-junction corridor
PTLC-4SA	1662.66 ± 4.27	154260.56 ± 387.05	2342.85 ± 39.73
PTLC-3SA	1896.72 ± 11.14	163523.28 ± 294.03	2543.01 ± 75.23
PTLC-H	1491.03 ± 3.63	$\mathbf{128351.90 \pm 190.46}$	$\mathbf{1569.15 \pm 1.33}$
PTLC-W	$\mathbf{967.49 \pm 2.69}$	133785.42 ± 189.96	1731.02 ± 20.76

The various parameters for the simulation of traffic using GLD are set as specified in Sect. VI.B of [16].

All the experiments are run in two phases—a tuning phase followed by a converged run phase. In the tuning phase, we run each algorithm for 500 iterations, where each iteration involves three simulations—one with the nominal parameter θ and the other two with the perturbed parameters $\theta + \delta\Delta + \delta\Delta$ and $\theta - \delta\Delta - \delta\hat{\Delta}$. The length of each iteration for a particular choice of parameter is 100. After the completion of the tuning phase, we freeze the parameter and run ten independent simulations with this (converged) choice of the parameter for 10,000 samples. This latter step where the simulations are conducted for the converged value of the parameter constitutes for converged run phase. The results reported in the following section are averages obtained over these ten simulation runs. The parameter δ in all the algorithms is set to 0.2. This choice of δ is motivated by a sensitivity study conducted with various values of δ, the results of which can be seen in Table 3 of [17].

5.1 Results

For the purpose of evaluating the TLC algorithms, we use $\hat{J}(\theta)$ and the average trip waiting time (ATWT) as the performance metrics. Here $\hat{J}(\theta)$ is the empirical average of the cost function $c(\cdot)$, i.e., $\hat{J}(\theta) = \frac{1}{T}\sum_{m=1}^{T} c(s_m)$. For our experiments, we set $T = 10{,}000$. Table 2 presents the values of $\hat{J}(\theta)$ obtained for the TLC algorithms on all the road networks considered. We observe that our algorithms perform better than both PTLC-4SA and PTLC-3SA (cf. [14]) on all the road networks considered, which can possibly be attributed to the new Hessian estimate in our algorithms. Further, unlike the 4SA algorithm of [14] which requires four simulations, both our algorithms

require only three simulations each and exhibit overall better performance at reduced cost.

Figure 2(a) presents the values of the ATWT performance obtained for the TLC algorithms studied on a 3 × 3-grid network; we observe that our algorithms result in waiting times that are either comparable (as in the PTLC-H case) or significantly lower (as in the PTLC-W case) than the PTLC-4SA and PTLC-3SA algorithms. The ATWT results of the TLC algorithms on the other two road networks exhibited a similar trend. Figure 2(b) presents the convergence behaviour of the parameter component L_4 (that was arbitrarily chosen here) for the PTLC-H and PTLC-W algorithms on a 3×3-grid network. Our algorithms are seen to converge during the simulation run and exhibit a short transient phase.

We also performed experiments where PTLC was run with 32 different fixed values of the θ-parameter vector on the 3 × 3-grid network and the results of this experiment are shown in Fig. 2(c). We show the values of $\hat{J}(\theta)$ for the PTLC algorithm in each case and compare these with the solutions obtained using PTLC-W. While we label θ as $1, 2, \ldots, 32$ on the x-axis in the figure, the precise values for each of the 32 (multi-dimensional) parameters θ are described in Table 4 of [17]. We observe that PTLC-W significantly outperforms the fixed θ counterparts for most parameter choices. Further, among the choices of the parameters that performed better, we observe that PTLC-W results in a performance that is very close to the fixed parameter counterpart.

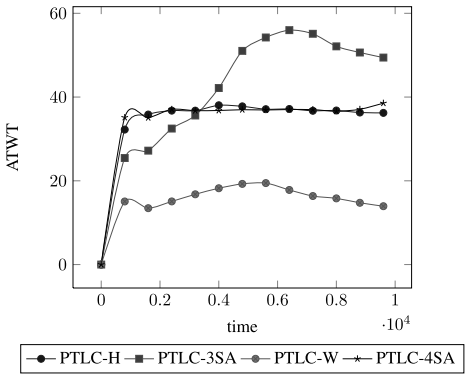
It may be noted that our algorithms are local search techniques and hence are not guaranteed to find the global optima. However, from our experiments, we observe that incorporating our Newton-based schemes results in performance enhancements for the PTLC algorithm on all the road network settings considered. Further, in comparison to the 4SA and 3SA algorithms of [14], our algorithms exhibit a superior overall performance.

6 Conclusions

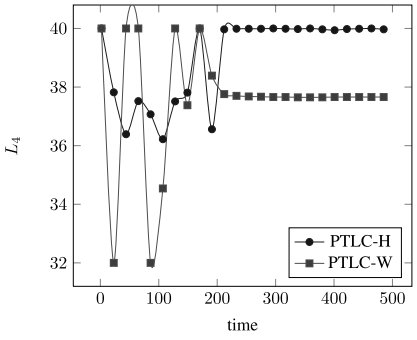
We presented in this paper a balanced three-simulation Hessian estimator based on the simultaneous perturbation technique. Using this Hessian estimator, we also presented two Newton-based algorithms that work around the normally tedious procedure of inverting the Hessian. We gave proofs of convergence for both our algorithms. Numerical experiments on various settings of road traffic control illustrate that our tuning algorithms in the setting of priority based traffic light control perform significantly better than both the 3SA and 4SA algorithms of [14], as well as fixed parameter algorithms, over a wide range of setting parameters. More experiments on other applications must however be tried to test the superiority of our algorithms over 3SA and 4SA.

While our algorithms require less computation when compared with other Newton-based approaches, they do have higher computational requirements than simple gradient search schemes. It would thus be interesting to develop, in the future, quasi-Newton algorithms for simulation optimization and study their performance characteristics as quasi-Newton algorithms are known to have lower computational

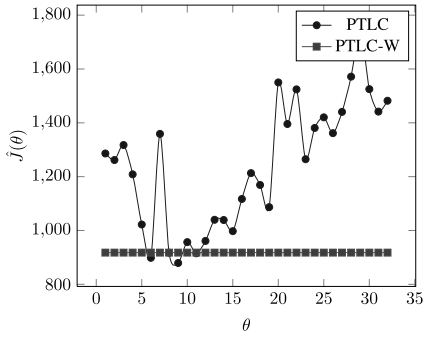
Fig. 2 Convergence and performance plots of PTLC algorithms on 3 × 3-grid network



(a) Comparison of PTLC algorithms using ATWT



(b) Convergence of θ - illustration using L_4 for PTLC-H and PTLC-W algorithms



(c) Comparison of $\hat{J}(\theta)$ for different (fixed) values of θ in PTLC with PTLC-W

requirements than pure Newton methods. Another possible direction would be to develop similar techniques for feature updates in reinforcement learning applications [11].

References

1. Chong, E.K.P., Ramadge, P.J.: Optimization of queues using an infinitesimal perturbation analysis-based stochastic algorithm with general update times. SIAM J. Control Optim. **31**(3), 698–732 (1993)
2. Ho, Y.C., Cao, X.R.: Perturbation Analysis of Discrete Event Dynamical Systems. Kluwer, Boston (1991)
3. Andradóttir, S.: Optimization of the transient and steady-state behavior of discrete event systems. Manag. Sci. **42**(5), 717–737 (1996)
4. Kiefer, E., Wolfowitz, J.: Stochastic estimation of the maximum of a regression function. Ann. Math. Stat. **23**, 462–466 (1952)
5. Spall, J.C.: Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans. Autom. Control **37**(3), 332–341 (1992)
6. Spall, J.C.: A one-measurement form of simultaneous perturbation stochastic approximation. Automatica **33**, 109–112 (1997)
7. Bhatnagar, S., Fu, M.C., Marcus, S.I., Bhatnagar, S.: Two-timescale algorithms for simulation optimization of hidden Markov models. IIE Trans. **33**(3), 245–258 (2001)
8. Bhatnagar, S., Fu, M.C., Marcus, S.I., Wang, I.: Two-timescale simultaneous perturbation stochastic approximation using deterministic perturbation sequences. ACM Trans. Model. Comput. Simul. **13**(2), 180–209 (2003)
9. Bhatnagar, S., Borkar, V.S.: Multiscale chaotic SPSA and smoothed functional algorithms for simulation optimization. Simulation **79**(10), 568–580 (2003)
10. Bhatnagar, S.: Adaptive Newton-based smoothed functional algorithms for simulation optimization. ACM Trans. Model. Comput. Simul. **18**(1), 2:1–2:35 (2007)
11. Bhatnagar, S., Prasad, H.L., Prashanth, L.A.: Stochastic Recursive Algorithms for Optimization: Simultaneous Perturbation Methods. Lecture Notes in Control and Information Sciences. Springer, London (2013)
12. Fabian, V.: Stochastic approximation. In: Rustagi, J.J. (ed.) Optimizing Methods in Statistics, pp. 439–470. Academic Press, New York (1971)
13. Spall, J.C.: Adaptive stochastic approximation by the simultaneous perturbation method. IEEE Trans. Autom. Control **45**, 1839–1853 (2000)
14. Bhatnagar, S.: Adaptive multivariate three-timescale stochastic approximation algorithms for simulation based optimization. ACM Trans. Model. Comput. Simul. **15**(1), 74–107 (2005)
15. Prashanth, L.A., Bhatnagar, S.: Reinforcement learning with function approximation for traffic signal control. IEEE Trans. Intell. Transp. Syst. **12**(2), 412–421 (2011)
16. Prashanth, L.A., Bhatnagar, S.: Threshold tuning using stochastic optimization for graded signal control. IEEE Trans. Veh. Technol. **61**(9), 3865–3880 (2012)
17. Bhatnagar, S., Prashanth, L.A.: Simultaneous perturbation Newton algorithms for simulation optimization. Technical report, Stochastic Systems Lab., IISc, (2013). http://stochastic.csa.iisc.ernet.in/www/research/files/IISc-CSA-SSL-TR-2013-4.pdf
18. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific, Belmont (1999)
19. Zhu, X., Spall, J.C.: A modified second-order SPSA optimization algorithm for finite samples. Int. J. Adapt. Control Signal Process. **16**, 397–409 (2002)
20. Borkar, V.S.: Stochastic Approximation: A Dynamical Systems Viewpoint. Cambridge University Press, Cambridge (2008)
21. Borkar, V.S.: Probability Theory: An Advanced Course. Springer, New York (1995)
22. Hirsch, M.W.: Convergent activation dynamics in continuous time networks. Neural Netw. **2**, 331–349 (1989)
23. Borkar, V.S., Meyn, S.P.: The O.D.E. method for convergence of stochastic approximation and reinforcement learning. SIAM J. Control Optim. **38**(2), 447–469 (2000)
24. Kushner, H.J., Clark, D.S.: Stochastic Approximation Methods for Constrained and Unconstrained Systems. Springer, New York (1978)
25. Lasalle, J.P., Lefschetz, S.: Stability by Liapunov's Direct Method with Applications. Academic Press, New York (1961)
26. Kushner, H.J., Yin, G.G.: Stochastic Approximation Algorithms and Applications. Springer, New York (1997)