Available online at www.sciencedirect.com
ScienceDirect
Automatica 43 (2007) 817–830
automatica
www.elsevier.com/locate/automatica

Simulation-based optimal sensor scheduling with application to observer trajectory planning[☆]

Sumeetpal S. Singh[a,*], Nikolaos Kantas[a], Ba-Ngu Vo[b], Arnaud Doucet[c,d], Robin J. Evans[b]

[a]*Signal Processing Group, Cambridge University Engineering Department, Trumpington Street, Cambridge CB2 1PZ, UK*
[b]*Department of Electrical and Electronic Engineering, University of Melbourne, Vic. 3010, Australia*
[c]*Department of Computer Science, University of British Columbia, Canada*
[d]*Department of Statistics, University of British Columbia, Canada*

Received 29 March 2005; received in revised form 30 May 2006; accepted 21 November 2006

Abstract

The sensor scheduling problem can be formulated as a controlled hidden Markov model and this paper solves the problem when the state, observation and action spaces are continuous. This general case is important as it is the natural framework for many applications. The aim is to minimise the variance of the estimation error of the hidden state w.r.t. the action sequence. We present a novel simulation-based method that uses a stochastic gradient algorithm to find optimal actions.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Sequential Monte Carlo; Particle filter; Stochastic approximation; Stochastic control; Sensor scheduling

1. Introduction

Consider the following continuous state hidden Markov model (HMM):

$$X_{n+1} = f(X_n, A_{n+1}, W_n), \qquad Y_n = g(X_n, A_n, V_n), \tag{1}$$

where $X_n \in \mathbb{R}^{d_x}$ is the hidden system state, $Y_n \in \mathbb{R}^{d_y}$ the observation of the state and W_n and V_n are i.i.d. noise terms. Unlike the classical HMM model, the evolution of the state and observation processes depends on an input parameter $A_n \in \mathbb{R}^{d_a}$, which is the control or action. In HMM models, one is primarily concerned with the problem of estimating the hidden state, which is achieved by propagating the posterior distribution (or filtering density) $\pi_n(x)\,\mathrm{d}x = \mathbf{P}(X_n \in \mathrm{d}x | A_{1:n}, Y_{1:n})$. By a judicious choice of action sequence $\{A_n\}$, the evolution of the state and observation processes can be 'steered' in order to yield filtering densities that give more accurate estimates of the state process. This problem is also known in the literature as the sensor scheduling problem.

Sensor scheduling has been a topic of interest to the target tracking community for the some years now (Hernandez, Kirubarajan, & Bar-Shalom, 2004; Kershaw & Evans, 1994; Logothetis, Isaksson, & Evans, 1997; Meier, Perschon, & Dressler, 1967; Singh, Kantas, Vo, & Doucet, 2005; Tremois & Le Cadre, 1999). The classical setting is the problem of tracking a manoeuvring target over N epochs. Here X_n denotes the state of the target at epoch n, Y_n the observation provided by the sensor and A_n some parameter of the sensor that may be adjusted to improve the 'quality' of the observation. For example, consider a non-moving platform with a finite number of sensors, where each has different characteristics. In this case A_n denotes the choice of sensor to be used at epoch n (Lee, Teo, & Lim, 2001; Singh, Vo, Doucet, & Evans, 2004). Alternatively, there may be only one sensor and A_n could denote some tunable parameter of the sensor, as in the waveform selection problem (Kershaw & Evans, 1994), or in the case of directing an electronically scanned aperture (ESA) radar (Blackman & Popoli, 1999). In contrast, consider the scenario in which a

[☆] This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Derong Liu under the direction of Editor Miroslav Krstic.
[*] Corresponding author.
E-mail addresses: sss40@eng.cam.ac.uk (S.S. Singh), nk234@eng.cam.ac.uk (N. Kantas), b.vo@ee.mu.oz.au (B. Vo), r.evans@ee.mu.oz.au (A. Doucet).

0005-1098/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2006.11.019

moving platform (or observer) is to be adaptively manoeuvred to optimise the tracking performance of a manoeuvring target. In this setting, A_n denotes the position of the observer at epoch n and the problem is termed the optimal *observer trajectory planning* (OTP) problem (Hernandez et al., 2004; Logothetis et al., 1997; Tremois & Le Cadre, 1999). In all these sensor scheduling problems, a measure of tracking performance is the mean squared tracking error over the N epochs,

$$\mathbf{E}\left\{\sum_{n=1}^{N}(\psi(X_n)-\langle\pi_n,\psi\rangle)^2\right\}, \tag{2}$$

where ψ: $\mathbb{R}^{d_x}\to\mathbb{R}$ is a suitable *test* function that emphasises the component (or components) of interest of the state vector we wish to track. The aim is to minimise (2) w.r.t. the choice of actions $\{A_1,\ldots,A_N\}$.

When the dynamics of the state and the observation processes are both linear and Gaussian, then the optimal solution to the sensor scheduling problem (2) (when ψ gives a quadratic cost) can be computed off line (Meier et al., 1967); this is not surprising given that the Kalman filter covariance is also independent of the actual realisation of observations. In the general setting studied in this paper, the dynamics can be both non-linear and non-Gaussian, which means that the filtering density π_n, and integration w.r.t. it, cannot be evaluated in closed form. Hence, the tracking error performance criterion itself does not admit a closed-form expression. To further complicate matters, the actions sought are continuous valued, i.e., vectors in $\mathbb{R}^{d_a}$.

To address the complications to do with the non-linear and non-Gaussian dynamics, one could linearise the state and observation model (as in Logothetis et al. (1997)), i.e., using the extended Kalman filter to propagate the filtering density π_n. However, dealing with the tracking error performance criterion directly is the exception rather than the rule. The majority of works (Hernandez et al., 2004; Paris & Le Cadre, 2002; Tremois & Le Cadre, 1999, and references therein), while aiming to minimise mean squared tracking error, do so indirectly by defining a lower bound to the tracking error criterion and minimising the lower bound instead. The bound in question is the posterior Cramer–Rao lower bound (PCRLB), which is the inverse of the Fisher information matrix (FIM). This approach hinges on the ability to propagate recursively the FIM in closed form by a Ricatti-type equation for the non-linear and non-Gaussian filtering problem. Unfortunately, the recursion for the FIM involves evaluating the expectation of certain derivatives of the transition probability density of the state dynamics, as well as the expectation of certain derivatives of the observation likelihood (see (3) and (4) below). As these quantities cannot be evaluated in general except for the linear and Gaussian case, this assumption is either invoked or the authors resort to simulation-based approximations.

As for the complications due to continuous valued actions, the approach in the literature is to discretise $\mathbb{R}^{d_a}$ to a grid. There have also been studies where the continuous state HMM (1) is approximated by a discrete state HMM, and the latter solved using dynamic programming (Tremois & Le Cadre, 1999).

The aim of this paper is to solve the sensor scheduling problem with continuous action space directly, and not a surrogate problem defined through the PCRLB or otherwise. We make no assumptions of linearity or Gaussianity for analytic convenience, nor do we discretise the state, observation or action space. We avoid these restrictive modelling assumptions on the continuous state HMM by recourse to methods based on computer simulation (simulation for short). As the action policy derived will be a function of the filtering density, we will employ simulation to approximate the posterior density by a finite sum of weighted point–mass distributions (Doucet, De Freitas, & Gordon, 2001). The main advantage of simulation over other numerical integration methods is that it is typically very easy to implement. Furthermore, it follows the strong law of large numbers (Del Moral, 2004) and there is much literature on its efficient implementation for approximating posterior densities (Doucet et al., 2001).

In order to solve for the optimal sequence of continuous valued actions, we will use an iterative stochastic gradient algorithm. We derive the gradient of the performance criterion w.r.t. the action trajectory and demonstrate how low-variance estimates of it may be obtained using *control variate* (CV) (Glynn & Szechtman, 2002) techniques. One major advantage of a stochastic gradient-based method is that theoretical guarantees are easily obtained. Under suitable regularity assumptions, one can guarantee convergence to a local optimum of the performance criterion, while it is difficult to make similar assertions about the quality of the solutions obtained by other approximate methods proposed in the literature for sensor scheduling.

As an instance of the sensor scheduling problem, we study the OTD problem for a bearings-only application. We state theoretical results concerning the convergence of the observer trajectory identified by our simulation-based algorithm. Handling multiple observers simultaneously is easy in our proposed framework, and numerical results are presented for cooperating observers tracking a manoeuvring target.

The organisation of this paper is as follows. In Section 2, we formulate the optimal sensor scheduling problem. We also summarise some key points concerning several methods in the literature that may be used to solve this problem. In Section 3, we derive the gradient of the performance criterion being optimised, and detail the use of simulation and variance reduction techniques for its estimation. In Section 4.2, we present the main algorithm of the paper, which is a two time-scale stochastic gradient algorithm for solving the sensor scheduling problem. General convergence results for this algorithm are presented in the Appendix. In Section 5, we formulate the OTD problem as an instance of the sensor scheduling problem and apply the convergence results to this application. Numerical examples are presented in Section 6, and concluding remarks are presented in Section 7. All proofs appear in the technical report (Singh et al., 2005), which is available online or may be obtained by e-mailing any of the authors.

Notation. The notation that is used in the paper is now outlined. The norm of a scalar, vector or matrix is denoted by $|\cdot|$. For a vector b, $|b|$ denotes the vector 2-norm

$\sqrt{\sum_i |b(i)|^2}$. For a matrix A, $|A|$ denotes the matrix 2-norm, $\max_{b:|b|\neq 0} |Ab|/|b|$. For convenience, we also denote a vector $b \in \mathbb{R}^n$ by $b = [b(i)]_{i=1,\dots,n}$, or the ith component of a vector by $[b]_i$. For scalars $a_{j,i}$, $j = 1, \dots, m$, $i=1, \dots, n$, let $[[a_{j,i}]_{j=1,\dots,m}]_{i=1,\dots,n}$ denote the stacked vector $[a_{1,1}, \dots, a_{m,1}, \dots, a_{1,n}, \dots, a_{m,n}]^{\mathrm{T}}$. For a vector b, let $\mathrm{diag}(b)$ denote the diagonal matrix formed from b. For a function $f : \mathbb{R}^n \to \mathbb{R}$ with arguments $z \in \mathbb{R}^n$, we denote $(\partial f/\partial z(i))(z)$ by $\nabla_{z(i)} f(z)$ and $\nabla f(z) = [\nabla_{z(1)} f(z), \dots, \nabla_{z(n)} f(z)]^{\mathrm{T}}$. For the vector-valued function $F = [F_1, \dots, F_n]^{\mathrm{T}} : \mathbb{R}^n \to \mathbb{R}^n$, let ∇F denote the matrix $[\nabla F_1, \dots, \nabla F_n]$. For real-valued integrable functions f and g, let $\langle f, g\rangle$ denote $\int f(x)g(x)\,\mathrm{d}x$.

2. Problem formulation

At time n, let X_n and Y_n be random vectors that model the d_x-dimensional state and its d_y-dimensional observation, respectively. Suppose that an action $A_n \in \mathbb{R}^{d_a}$ is applied at time n. The state $\{X_n\}_{n\geqslant 0}$ is an unobserved Markov process with initial distribution and transition law given by

$$X_0 \sim \pi_0, \quad X_{n+1} \sim p(\cdot|X_n, A_{n+1}). \tag{3}$$

(The symbol '$\sim$' means distributed according to.) The observation process $\{Y_n\}_{n\geqslant 1}$ is generated according to the state- and action-dependent probability density

$$Y_n \sim q(\cdot|X_n, A_n). \tag{4}$$

Given the sequence of actions $a_{1:n} := \{a_1, \dots, a_n\}$ and measurements $y_{1:n} := \{y_1, \dots, y_n\}$, the *filtering density* at time n is denoted by π_n (or $\pi_n^{(y_{1:n},a_{1:n})}$ to emphasise the dependence on $y_{1:n}$ and $a_{1:n}$,) and satisfies the *Bayes* recursion

$$\pi_n(x) = \frac{q(y_n|x, a_n)\int p(x|x', a_n)\pi_{n-1}(x')\,\mathrm{d}x'}{\iint q(y_n|x, a_n)p(x|x', a_n)\pi_{n-1}(x')\,\mathrm{d}x'\,\mathrm{d}x}. \tag{5}$$

Consider a suitable *test* function $\psi : \mathbb{R}^{d_x} \to \mathbb{R}$ where, for example, ψ could pick out a component of interest of the state vector we wish to estimate. The optimal sensor scheduling problem is to solve

$$\min_{A_{1:N}\in\Theta_A} J(A_{1:N}) = \mathbf{E}_{(\pi_0,A_{1:N})}\left\{\sum_{n=1}^{N} \lambda^{N-n}(\psi(X_n) - \langle\pi_n, \psi\rangle)^2\right\}, \tag{6}$$

where $\mathbf{E}_{(\pi_0,A_{1:N})}$ denotes expectation w.r.t. the random variables $(X_{0:N}, Y_{1:N})$ which are distributed according to the law specified by $(\pi_0, A_{1:N})$. For any $1\leqslant n\leqslant N$ and integrable function $h : (\mathbb{R}^{d_x})^n \times (\mathbb{R}^{d_a})^n \times (\mathbb{R}^{d_y})^n \to \mathbb{R}$,

$$\mathbf{E}_{(\pi_0,A_{1:n})}\{h(X_{1:n}, A_{1:n}, Y_{1:n})\} = \int h(x_{1:n}, A_{1:n}, y_{1:n}) \times \underbrace{\Pi_{i=1}^n q(y_i|x_i, A_i)p(x_i|x_{i-1}, A_i)\pi_0(x_0)\,\mathrm{d}x_{0:n}\,\mathrm{d}y_{1:n}}_{\mathbf{P}_{(\pi_0,A_{1:N})}}. \tag{7}$$

The set of actions $\Theta_A \subset (\mathbb{R}^{d_a})^N$ is open and $\lambda \in [0, 1]$ is a discount factor to favour better tracking performance in the later epochs if so desired.

Feedback control: The sensor scheduling problem stated in (6) is an open-loop stochastic control problem. In order to utilise feedback in implementation, we will use the *open-loop feedback control* (OLFC) approach, which is described as follows (see Bertsekas, 1995 for a detailed account).

Path constraints: For trajectory planning application, the sequence of actions $A^*_{1:N}$ determined by solving (6) above may not be realisable due to motion constraints on the observer. For example, the sequence of possible accelerations in a linear motion model (see Section 5) may not be able to realise the desired sequence of positions $A^*_{1:N}$. Alternatively, we may seek a sequence of positions that belong to some parametric class, e.g., an observer doing a constant velocity turn where the turn rate is to be determined. In some cases, we may summarise observer motion constraints through a bounded mapping

$$F : (\mathbb{R}^{d_u})^N \to (\mathbb{R}^{d_a})^N, \tag{8}$$

where the actions $A_{1:N} = F(U_1, \dots, U_N)$. For example, $U_{1:N}$ could describe the sequence of accelerations of an observer; see Section 5 for more details. We would then solve problem (6) subject to the equality constraint

$$A_{1:N} = F(U_{1:N}), \quad U_{1:N} \in (\mathbb{R}^{d_u})^N, \tag{9}$$

i.e., Θ_A now corresponds to the range of the function F.

Simulation- and gradient-based methods: We do not have a closed-form expression for J because the filtering density π_n and integration w.r.t. it cannot be evaluated in closed-form in our general setting. To evaluate $J(A_{1:N})$, we could revert to state-space discretisation (see Hernandez-Lerma, 1989 for issues on discretising general state-space HMMs). One could discretise $\mathbb{R}^{d_x}$, $\mathbb{R}^{d_y}$ and derive the corresponding state evolution and observation laws, i.e., (3) and (4), for the approximating discrete problem. We may then calculate the approximation to $J(A_{1:N})$ for any choice of actions. This approach has its drawbacks though. Firstly, assuming that $\mathbb{R}^{d_x}$ and $\mathbb{R}^{d_y}$ are discretised to finite sets of cardinality L_x and L_y, respectively, then the multiple integral in (6) (cf. (7)) is converted to a sum over $(L_y)^N \times (L_x)^{N+1}$ terms, which is computationally prohibitive. Thus, we would be limited to a coarse discretisation and a small horizon N at best. Secondly, it is not obvious how to choose the grid in $\mathbb{R}^{d_x}$ and $\mathbb{R}^{d_y}$, since for accuracy of the approximation, the grid should be finer in the regions where density in (7) has more mass. In Tremois and Le Cadre (1999), the HMM is discretised and a closed-loop formulation of problem (6) is solved. A closed-loop formulation of (6) is known as a partially observed Markov decision process (POMDP). However, solving a POMDP exactly is computationally very demanding, specifically PSPACE-hard, and various approximation schemes that trade-off accuracy and speed have been devised (Hauskrecht, 2000).

We propose to use simulation with stochastic approximation (SA) to minimise $J(A_{1:N})$ when Θ_A is an open (i.e., continuous) set without resorting to discretising $\mathbb{R}^{d_x}$, $\mathbb{R}^{d_y}$ or Θ_A. SA is a gradient descent algorithm that only requires noisy estimates of the cost function gradient, i.e.,

$$A_{1:N,k+1} = A_{1:N,k} - \alpha_k(\nabla J(A_{1:N})|_{A_{1:N}=A_{1:N,k}} + \text{noise}), \quad (10)$$

where $k \geqslant 0$ and $\nabla J(A_{1:N})$ denotes the gradient of J w.r.t. $A_{1:N}$. The step-size α_k is a non-increasing positive sequence tending to zero. In Section 3, we derive the gradient ∇J. Once again, we do not have a closed form expression for ∇J for the same reasons as in J; the filtering density π_n and integration w.r.t. it cannot be evaluated in closed form in our general setting. We will show instead how one may obtain an estimate of ∇J, namely $\widehat{\nabla J}$. The noise in (10) arises precisely because we use $\widehat{\nabla J}$ instead of ∇J. The smaller the variance of the noise is, the more quickly (10) converges to a minimising action sequence. This motivates us to consider techniques such as CV (Glynn & Szechtman, 2002) to reduce the noise variance. We propose an *adaptive* CV method to reduce the variance of $\widehat{\nabla J}$ by coupling with (10) a second SA iteration that estimates the optimal control variates for $\widehat{\nabla J}$. We then demonstrate in numerical examples that the variance of $\widehat{\nabla J}$ is reduced by several orders of magnitude and that the convergence of (10) to the minimising solution is accelerated. In Appendix A, we address the convergence of the proposed method.

One major advantage of a gradient-based method is that theoretical guarantees are easily obtained. Under suitable assumptions on the noise in (10), one can guarantee that $A_{1:N,k}$ eventually converges to a local minimiser of J, while it is difficult to make similar assertions about the quality of the solution obtained by other approximate methods proposed in the literature. However, the method we propose is not yet suitable for real-time applications where decisions need to be made in tens of seconds. Our main intention in this study is to propose a simulation-based method for non-linear and non-Gaussian systems that is both easy to implement and provably convergent. We wish to avoid restrictive modelling assumptions so that the proposed method is generally applicable. The conventional approach in the literature is to linearise the non-linear and non-Gaussian dynamics which is not always straightforward to implement from a numerical stability point. Also, one cannot decrease the error in the approximation of the original problem—note that linearising is an approximation. The same is true for methods based on the PCRLB. In the method we propose, however, one merely increases the number of particles or samples L, which could not be simpler. However, there is a computational cost to pay. There is currently work being done on the fast implementation of particle filters but we have not investigated this aspect of the problem.

3. The cost gradient and its simulation-based approximation

In this section, we derive the gradient of the cost function (6) w.r.t. $A_{1:N}$. We then propose a suitable simulation-based approximation for optimising with SA. Because problem (6) is solved for a fixed initial state distribution π_0, henceforth, we omit reference to π_0 in the notation for $\mathbf{E}_{(\pi_0,A_{1:N})}$ and denote the probability w.r.t. which this expectation is taken by $\mathbf{P}_{A_{1:N}}$.

Keeping in mind that $(\psi(X_n) - \langle\pi_n, \psi\rangle)^2$ is a function of the form $h(X_{1:n}, A_{1:n}, Y_{1:n})$, (7) implies

$$\mathbf{E}_{A_{1:N}}\{(\psi(X_n) - \langle\pi_n, \psi\rangle)^2\} = \mathbf{E}_{A_{1:n}}\{(\psi(X_n) - \langle\pi_n, \psi\rangle)^2\}.$$

For $l > n$, $\nabla_{A_l}\mathbf{E}_{A_{1:N}}\{(\psi(X_n) - \langle\pi_n, \psi\rangle)^2\} = 0$. For $l \leqslant n$, using (7), $\nabla_{A_l}\mathbf{E}_{A_{1:N}}\{(\psi(X_n) - \langle\pi_n, \psi\rangle)^2\}$ is

$$\int (\psi(x_n) - \langle\pi_n^{(y_{1:n},A_{1:n})}, \psi\rangle)^2 \nabla_{A_l}\mathbf{P}_{A_{1:N}}(x_{0:n}, y_{1:n})\,\mathrm{d}x_{0:n}\,\mathrm{d}y_{1:n}$$
$$+ \int \nabla_{A_l}[(\psi(x_n) - \langle\pi_n^{(y_{1:n},A_{1:n})}, \psi\rangle)^2]$$
$$\times \mathbf{P}_{A_{1:N}}(x_{0:n}, y_{1:n})\,\mathrm{d}x_{0:n}\,\mathrm{d}y_{1:n}.$$

The first term of the gradient may be written as

$$\mathbf{E}_{A_{1:N}}\left\{(\psi(X_n) - \langle\pi_n, \psi\rangle)^2 \times \left[\frac{\nabla_{A_l}q(Y_l|X_l, A_l)}{q(Y_l|X_l, A_l)} + \frac{\nabla_{A_l}p(X_l|X_{l-1}, A_l)}{p(X_l|X_{l-1}, A_l)}\right]\right\}$$

and the second term is, omitting the factor -2,

$$\mathbf{E}_{A_{1:N}}\{(\psi(X_n) - \langle\pi_n^{(Y_{1:n},A_{1:n})}, \psi\rangle)\nabla_{A_l}\langle\pi_n^{(Y_{1:n},A_{1:n})}, \psi\rangle\},$$

which evaluates to zero upon conditioning on $Y_{1:n}$. It follows from the above derivation that to obtain an unbiased estimator of $\nabla_{A_l}J(A_{1:N})$ for a given $A_{1:N}$, one samples a realisation of states and observations $(Y_{1:N}, X_{0:N}) \sim \mathbf{P}_{A_{1:N}}$ and forms the following estimate:

$$\widehat{\nabla_{A_l}J}(A_{1:N}) = \sum_{n=l}^{N}\lambda^{N-n}\mathbf{E}_{A_{1:N}}\left\{(\psi(X_n) - \langle\pi_n, \psi\rangle)^2 \times \left[\frac{\nabla_{A_l}q(Y_l|X_l, A_l)}{q(Y_l|X_l, A_l)} + \frac{\nabla_{A_l}p(X_l|X_{l-1}, A_l)}{p(X_l|X_{l-1}, A_l)}\right]\middle| Y_{1:n}\right\}, \quad (11)$$

where we have added the conditioning on $Y_{1:n}$ as it leads to a lower variance gradient estimate.[1] In sensor scheduling applications concerning target tracking, the state process X_n is the state of the target to be tracked and often evolves independent of the action. Henceforth, we assume this independence for simplicity in presentation, i.e., $p(X_n|X_{n-1})$, and remark that the work may also be extended to the more general case of state evolution and action dependence.[2] Define the vector-valued

[1] The variance is reduced since, for two jointly distributed random variables X and Y, $\mathbf{var}(\mathbf{E}(X|Y)) = \mathbf{var}(X) - \mathbf{E}(\mathbf{var}(X|Y))$, and $\mathbf{E}(\mathbf{var}(X|Y)) > 0$.

[2] In methods that use the PCRLB (Hernandez et al., 2004; Paris & Le Cadre, 2002; Tremois & Le Cadre, 1999), even after assuming that the state process evolves independent of the actions, one still needs to evaluate the expectation of derivatives of $\ln p(X_n|X_{n-1})$ w.r.t. X_n and X_{n-1}, while this is clearly not needed in (11).

function called the *score* (Pflug, 1996),

$$S(y,x,a) := \frac{\nabla_a q(y|x,a)}{q(y|x,a)} \in \mathbb{R}^{d_a}. \tag{12}$$

We may also write $\widehat{\nabla_{A_l} J}(A_{1:N})$ in (11) as

$$\sum_{n=l}^{N} \lambda^{N-n}\{\langle \pi_{0:n}, \psi^2(\cdot)S(Y_l,\cdot,A_l)\rangle + \langle \pi_n,\psi\rangle^2 \langle \pi_{0:n}, S(Y_l,\cdot,A_l)\rangle - 2\langle \pi_n,\psi\rangle\langle \pi_{0:n}, \psi(\cdot)S(Y_l,\cdot,A_l)\rangle\}. \tag{13}$$

To implement (11), we see that we require both the marginal π_n and the full posterior $\pi_{0:n}$ for all N epochs, i.e., for $1 \leqslant n \leqslant N$. We propose to approximate these densities using a mixture Dirac delta-masses,

$$\hat{\pi}_{0:n}(x_{0:n}) := \sum_{j=1}^{L} w_n^{(j)} \delta_{X_{0:n}^{(j)}}(x_{0:n}), \tag{14}$$

where $\delta_{X_{0:n}^{(j)}}$ denotes the Dirac delta-mass located at $X_{0:n}^{(j)}$ and the *importance weights* $\{w_n^{(j)}\}_{j=1}^{L}$ are non-negative scalars that sum to one. The approximation to π_n, namely $\hat{\pi}_n$, follows by marginalising $\hat{\pi}_{0:n}$, which is nothing more than dropping $X_{0:n-1}^{(j)}$ in (14). There are a number of ways to define such a point–mass approximation. For example, the simplest scheme would be to sample L independent state trajectory realisations $\{X_{0:n}^{(j)}\}_{j=1}^{L}$ from $(\Pi_{i=1}^{n} p(x_i|x_{i-1}))\pi_0(x_0)$. The importance weights would then be

$$w_n^{(j)} := \frac{\Pi_{i=1}^{n} q(Y_i|X_i^{(j)},A_i)}{\sum_{j=1}^{L} \Pi_{i=1}^{n} q(Y_i|X_i^{(j)},A_i)}. \tag{15}$$

For any integrable function h, $\int h(x_{0:n})\hat{\pi}_{0:n}(x_{0:n})\,\mathrm{d}x_{0:n}$ converges to $\int h(x_{0:n})\pi_{0:n}(x_{0:n})\,\mathrm{d}x_{0:n}$ as $L \to \infty$ (see Doucet, De Freitas et al., 2001, Chap. 2 for a precise statement of the mode of convergence). Practically though, we would prefer a small sample size L and this simple scheme of sampling from the state transition model can result in the majority of the importance weights $w_n^{(j)}$ being very small. There are a number of remedies proposed for this in the sequential Monte Carlo, also known as particle filtering (PF), literature (Doucet, De Freitas et al., 2001, Chap. 1.3.2). For example, the importance sampling step can be designed to minimise the conditional variance of the importance weights by sampling $\{X_{0:n}^{(j)}\}_{j=1}^{L}$ from a Markov transition density that takes the observations into account, i.e., $X_n^{(j)}|X_{n-1}^{(j)} \sim k(x_n|X_{n-1}^{(j)},Y_n)$. We emphasise that standard techniques from the sequential Monte Carlo literature can be adopted in constructing an approximation of the form (14) to the full posterior but we do not study this issue in detail here.

We summarise the discussion thus far with the following algorithm.

Simulation-based sensor scheduling procedure:
0. Initialisation: Choose $A_{1:N,0} \in \Theta_A$, step-size $\{\alpha_k\}_{k\geqslant 1}$, $\alpha_k \downarrow 0$, $\sum_k \alpha_k = \infty$, PF sample size L
For $k \geqslant 0$, iterate
1. Sample $(X_{0:N}, Y_{1:N}) \sim P_{A_{1:N,k}}$
2. Generate the particle filter $\hat{\pi}_{0:N}$ according to (15) or a more sophisticated scheme (Doucet, Gordon, & Krishnamurthy, 2001)
3. Substitute $(X_{0:N}, Y_{1:N})$, $A_{1:N,k}$ and $\hat{\pi}_{0:N}$ into (13) to obtain $\widehat{\nabla J}(A_{1:N,k})$:

$$\widehat{\nabla_{A_l} J}(A_{1:N,k}) = \sum_{n=l}^{N} \lambda^{N-n}\{\langle \hat{\pi}_{0:n}, \psi_n^2(\cdot)S(Y_l,\cdot,A_{l,k})\rangle + \langle \hat{\pi}_n,\psi_n\rangle^2 \langle \hat{\pi}_{0:n}, S(Y_l,\cdot,A_{l,k})\rangle - 2\langle \hat{\pi}_n,\psi\rangle\langle \hat{\pi}_{0:n}, \psi_n(\cdot)S(Y_l,\cdot,A_{l,k})\rangle\} \tag{16}$$

4. Update trajectory: $A_{1:N,k+1} = A_{1:N,k} - \alpha_k \widehat{\nabla J}(A_{1:N,k})$
5. Set $k = k+1$ and repeat

One may use a constant step-size $\alpha_k = \alpha$ as was done in the numerical implementation; see Section 6. In implementation we found that the variance of the gradient estimate (16) was large. The reason is that, for a large horizon N, we are approximating high-dimensional integrals using simulation and, more so, with a moderate sample size L. We propose a remedy in Section 4.1.

Computation complexity: The particle filter $\hat{\pi}_{0:N}$ can be implemented at a cost of $O(LN)$ (Doucet, De Freitas et al., 2001) with or without resampling (as in (15)). This cost dominates the cost of sampling $(X_{0:N}, Y_{1:N})$ from $P_{A_{1:N,k}}$. Thus, the total cost per iteration k of the simulation-based sensor scheduling procedure is still order $O(LN)$.

4. A verifiably convergent particle implementation

Implementing the algorithm detailed in Section 3 with the gradient estimate (16) is straightforward. However, to prove its convergence we would not be able to use standard SA results. Even though (16) is a noisy estimate of $\nabla_{A_l} J(A_{1:N})$, the noise is not zero-mean due to the bias of the simulation-based approximations to π_n and $\pi_{0:n}$. To assert convergence of (10) to a minima of J, we would have to gradually increase the number of samples L to remove the bias. (Similar conditions are required for convergence of SA driven by sample averages, Pflug, 1996.) While this is fine theoretically, it is infeasible in practice as the computational complexity of the SA recursion increases with each iteration. In this section we propose an alternate implementation whose convergence can be established for a finite number of particles L. Moreover, in Section 5 we show that the proposed implementation applied to the standard OTP problem with bearings-only observations converges.

To simplify the presentation, we will only focus on the simple scheme of sampling from the state transition model as in (15) and not the more sophisticated PF presented in the

Appendix. To emphasise the dependence of $\hat{\pi}_{0:n}$ on the realisation of observations $Y_{1:n}$ and the sequence of actions $A_{1:n}$, we should use the notation $\hat{\pi}_{0:n}^{(Y_{1:n},A_{1:n})}$. However, we often do not do so in order to unclutter the expressions. **The reader is reminded that $\hat{\pi}_{0:n}$ should always be regarded as a function of** $(Y_{1:n}, A_{1:n})$. Henceforth, we fix the set of L state trajectory samples $\{X_{0:N}^{(j)}\}_{j=1}^{L}$, i.e., they are sampled once at the start and reused throughout to form $\hat{\pi}_{0:n}$.

By a conditioning argument, $J(A_{1:N})$ can be written as $\sum_{n=1}^{N} \lambda^{N-n} \mathbf{E}_{A_{1:N}}\{\langle \pi_n, \psi^2\rangle - \langle \pi_n, \psi\rangle^2\}$ and we form the following approximation to J:

$$\hat{J}(A_{1:N}) = \sum_{n=1}^{N} \lambda^{N-n} \mathbf{E}_{A_{1:N}}\{\langle \hat{\pi}_n, \psi^2\rangle - \langle \hat{\pi}_n, \psi\rangle^2\}. \quad (17)$$

Since the error in the approximation $\hat{\pi}_{0:n}$ diminishes as the sample size L increases, we would expect $\hat{J}$ to be a good approximation to J for sufficiently large L. We will then derive an unbiased estimate of the gradient of $\hat{J}$ in a similar manner to J above and minimise $\hat{J}$ via SA. This approach can be analysed and we show that, under suitable assumptions, SA converges to a local minima of $\hat{J}$ almost surely.

In the same way as gradient of J was derived in (11) we have

$$\begin{aligned}
&\nabla_{A_l} \hat{J}(A_{1:N}) \\
&\quad = \sum_{n=1}^{N} \lambda^{N-n} \nabla_{A_l} \mathbf{E}_{A_{1:N}}\{\langle \hat{\pi}_n, \psi^2\rangle - \langle \hat{\pi}_n, \psi\rangle^2\} \\
&\quad = \mathbf{E}_{A_{1:N}}\left\{\sum_{n=l}^{N} \lambda^{N-n}(\langle \hat{\pi}_n, \psi^2\rangle - \langle \hat{\pi}_n, \psi\rangle^2) S(Y_l, X_l, A_l)\right\} \quad (18) \\
&\qquad + \mathbf{E}_{A_{1:N}}\left\{\sum_{n=l}^{N} \lambda^{N-n}(\nabla_{A_l}\langle \hat{\pi}_n, \psi^2\rangle - 2\langle \hat{\pi}_n, \psi\rangle \nabla_{A_l}\langle \hat{\pi}_n, \psi\rangle)\right\}, \quad (19)
\end{aligned}$$

where $\nabla_{A_l}\langle \hat{\pi}_n, \psi\rangle$ evaluates to[3]

$$\langle \hat{\pi}_{0:n}, \psi(\cdot) S(Y_l, \cdot, A_l)\rangle - \langle \hat{\pi}_n, \psi\rangle \langle \hat{\pi}_{0:n}, S(Y_l, \cdot, A_l)\rangle.$$

It is now straightforward to obtain a simulation-based approximation of $\widehat{\nabla J}(A_{1:N})$. For a given $A_{1:N}$, one samples a realisation of states and observations $(Y_{1:N}, X_{0:N}) \sim \mathbf{P}_{A_{1:N}}$ and forms the following unbiased estimate of $\nabla_{A_l}\hat{J}(A_{1:N})$, for $l = 1, \ldots, N$:

$$\begin{aligned}
&S(Y_l, X_l, A_l) \sum_{n=l}^{N} \lambda^{N-n}(\langle \hat{\pi}_n, \psi^2\rangle - \langle \hat{\pi}_n, \psi\rangle^2) \\
&\quad + \sum_{n=l}^{N} \lambda^{N-n}(\nabla_{A_l}\langle \hat{\pi}_n, \psi^2\rangle - 2\langle \hat{\pi}_n, \psi\rangle \nabla_{A_l}\langle \hat{\pi}_n, \psi\rangle). \quad (20)
\end{aligned}$$

[3] It is possible to compute $\nabla_{A_l}\langle \hat{\pi}_n, \psi\rangle$ when resampling is employed to construct $\hat{\pi}_n$; see Poyiadjis, Singh, and Doucet (2002) for details.

4.1. Variance reduction by Control Variates

In implementation, we found that the variance of the gradient estimate (20) (or (16)) was quite large. This is because we are approximating high-dimensional integrals using simulation and, more so, with a moderate sample size L. Naturally, it would be possible to reduce the variance by simply increasing the number of samples. As we do not wish to do so, our aim is to extract the most accurate estimates of the quantities of interest for a given set of samples. Control Variates are widely used to reduce the variance in simulation-based approximations. The method involves collecting additional statistics from the samples and is very simple to implement. Recently, it has been shown that other popular variance reduction techniques such as conditional Monte Carlo, antithetics, rotation sampling, stratification can be viewed as various implementations of this method (Glynn & Szechtman, 2002). We now describe how control variates may be implemented for our problem.

For a random variable W, consider the problem of estimating $\mathbf{E}(W)$ when we have access to a zero-mean random variable Z correlated with W. Rather than using a realisation of W as an unbiased estimate, we use $W - bZ$ where b is a constant. The estimator $W - bZ$ is also unbiased. Furthermore, the function of b

$$\mathbf{var}(W - bZ) = \mathbf{var}(W) - 2b\,\mathbf{cov}(W, Z) + b^2\,\mathbf{var}(Z) \quad (21)$$

is convex and is minimised at $b^* = \mathbf{cov}(W, Z)/\mathbf{var}(Z)$, which implies that the variance of the estimate $W - b^*Z$ of $\mathbf{E}(W)$ is less than the variance of the estimate W. The random variable Z is referred to as the control variate (CV) and we call b the CV constant (Glynn & Szechtman, 2002).

In the context of the gradient estimate in (20), we found in implementation that reducing the variance of the estimate of (18) was sufficient. The score in (18) is zero-mean, i.e., $\mathbf{E}_{A_{1:N}}\{S(Y_l, X_l, A_l)\} = 0$, and we use it as the CV. Doing so yields the following unbiased estimator of $\nabla_{A_l}\hat{J}$ instead of (20):

$$\begin{aligned}
&\operatorname{diag}(S(Y_l, X_l, A_l))\left(\sum_{n=l}^{N} \lambda^{N-n}(\langle \hat{\pi}_n, \psi^2\rangle - \langle \hat{\pi}_n, \psi\rangle^2)\mathbf{1} - b_l\right) \\
&\quad + \sum_{n=l}^{N} \lambda^{N-n}(\nabla_{A_l}\langle \hat{\pi}_n, \psi^2\rangle - 2\langle \hat{\pi}_n, \psi\rangle \nabla_{A_l}\langle \hat{\pi}_n, \psi\rangle), \quad (22)
\end{aligned}$$

where $\mathbf{1} \in \mathbb{R}^{d_a}$ and the CV constant (vector) $b_l \in \mathbb{R}^{d_a}$ is to be determined in order to minimise the variance of the estimate. Noting that the optimal CV constant is a solution of the minimisation problem (21), we may employ the following SA algorithm to converge to it:

$$\begin{aligned}
b_l \longleftarrow\ & b_l - \beta \operatorname{diag}(S(Y_l, X_l, A_l))(\operatorname{diag}(S(Y_l, X_l, A_l))b_l \\
& - \sum_{n=l}^{N} \lambda^{N-n}(\langle \hat{\pi}_n, \psi^2\rangle - \langle \hat{\pi}_n, \psi\rangle^2)\mathbf{1}), \quad (23)
\end{aligned}$$

where β is the step-size. Under suitable assumptions stated in Appendix A, we will have b_l converging to

$$\mathbf{E}_{A_{1:N}}\{\mathrm{diag}\,(S(Y_l, X_l, A_l))^2\}^{-1}\mathbf{E}_{A_{1:N}}\left\{\mathrm{diag}(S(Y_l, X_l, A_l))^2 \times \sum_{n=l}^{N} \lambda^{N-n}(\langle\hat{\pi}_n, \psi^2\rangle - \langle\hat{\pi}_n, \psi\rangle^2)\mathbf{1}\right\}. \quad (24)$$

The same approach applies when minimising the variance of the gradient estimate (16) with CV.

4.2. The main algorithm

We now state the main algorithm of the paper whose convergence we subsequently prove. It is a two time-scale SA algorithm to minimise $\hat{J}$ using the reduced variance estimate of $\widehat{\nabla J}$ given by (22) and (23). We do so for the case with action path constraints as specified in (9). We can also derive a similar two time-scale version of the algorithm presented in Section 3.

Solving problem (6) with (9) is equivalent to minimising $\hat{J} \circ F$ (which is the composite function $\hat{J}(F(\cdot))$) over $(\mathbb{R}^{d_u})^N$. The appropriate modification to (10) for this case is

$$U_{1:N,k+1} = U_{1:N,k} - \alpha_k(\widehat{\nabla J} \circ F(U_{1:N})|_{U_{1:N}=U_{1:N,k}} + \text{noise}),$$

where $\widehat{\nabla J} \circ F(U_{1:N}) = \nabla F(U_{1:N})\widehat{\nabla J}(F(U_{1:N}))$.

We introduce the following functions to make the presentation of the main algorithm concise. For each $A_{1:N}$, define the functions $h_{i,A_{1:N}} : (\mathbb{R}^{d_x})^{N+1} \times (\mathbb{R}^{d_y})^N \to (\mathbb{R}^{d_a})^N$, $i = 1, 2$ as follows:

$$h_{1,A_{1:N}}(X_{0:N}, Y_{1:N}) := \left[S(Y_l, X_l, A_l) \sum_{n=l}^{N} \lambda^{N-n}(\langle\hat{\pi}_n, \psi^2\rangle - \langle\hat{\pi}_n, \psi\rangle^2) \right]_{l=1,\ldots,N}, \quad (25)$$

$$h_{2,A_{1:N}}(X_{0:N}, Y_{1:N}) := \left[\sum_{n=l}^{N} \lambda^{N-n}(\nabla_{A_l}\langle\hat{\pi}_n, \psi^2\rangle - 2\langle\hat{\pi}_n, \psi\rangle\nabla_{A_l}\langle\hat{\pi}_n, \psi\rangle) \right]_{l=1,\ldots,N}. \quad (26)$$

Note that

$$\widehat{\nabla J}(A_{1:N}) = \mathbf{E}_{A_{1:N}}\{h_{1,A_{1:N}}(X_{0:N}, Y_{1:N}) + h_{2,A_{1:N}}(X_{0:N}, Y_{1:N})\} \in (\mathbb{R}^{d_a})^N.$$

For technical reasons concerning the convergence of the two time-scale SA algorithm below, we introduce the positive scalar-valued function $\Gamma : (\mathbb{R}^{d_a})^N \to (0, \infty)$,

$$\Gamma(b) := (1 + |b|)^{-1}C, \quad (27)$$

where C is a positive constant. The function Γ is needed to ensure that the CV constants remain bounded almost surely (more details in Appendix A). However, we set $\Gamma(b) = 1$ in implementation.

The two time-scale SA algorithm for solving the sensor scheduling problem:
For conciseness, let $\theta = U_{1:N}$, $\tilde{\theta} = A_{1:N}(=F(\theta)),\omega= (X_{0:N}, Y_{1:N})$.

$$\theta_{k+1} = \theta_k - \alpha_{k+1}\Gamma(b_k)\nabla F(\theta_k)(h_{1,\tilde{\theta}_k}(\omega_{k+1}) + h_{2,\tilde{\theta}_k}(\omega_{k+1}) - S_{\tilde{\theta}_k}(\omega_{k+1})b_k), \quad (28)$$

$$b_{k+1} = b_k - \beta_{k+1}S^2_{\tilde{\theta}_k}(\omega_{k+1})b_k + \beta_{k+1}S_{\tilde{\theta}_k}(\omega_{k+1})h_{1,\tilde{\theta}_k}(\omega_{k+1}), \quad (29)$$

$$\omega_{k+1} \sim \mathbf{P}_{\tilde{\theta}_k}, \quad \tilde{\theta}_k = F(\theta_k), \quad k \geqslant 0, \quad (30)$$

Note that $\theta_k = U_{1:N,k}$, $\tilde{\theta}_k = A_{1:N,k}$, $\omega_{k+1}= (X_{0:N,k+1}, Y_{1:N,k+1})$ and

$$S_{A_{1:N,k}}(X_{0:N,k+1,}Y_{1:N,k+1}) = \mathrm{diag}([S(Y_{l,k+1}, X_{l,k+1}, A_{l,k})]_{l=1,\ldots,N}). \quad (31)$$

As for the algorithm presented in Section 3, the cost of sampling ω_{k+1} and computing $\hat{\pi}_{0:N}$ is $O(NL)$. The only difference is that we sample the state trajectories that form the approximate posterior density $\hat{\pi}_{0:N}$ (which also gives us $\{\hat{\pi}_n\}_{n=0}^N$) at the start and do not change them thereafter. Also, no resampling is employed. The storage requirement does, however, increase. Computing $h_{2,\tilde{\theta}_k}$ is the most expensive step, specifically $O(N^2L)$, and subsumes all other costs, matrix multiplications included. Thus the total cost of the algorithm is $O(N^2L)$. Note that the cost is still linear in L, which is the most important point since the horizon is typically very small compared to the number of state trajectory samples L.

Assumption 1. The step-size sequences $\{\alpha_k\}$ and $\{\beta_k\}$ are non-negative, sum to infinity, are squared summable and for some $p > 0$ satisfy $\sum_k (\alpha_k/\beta_k)^p < \infty$.

Typically, the step-sizes are

$$\alpha_k = k^{-\alpha}, \quad \beta_k = k^{-\beta}, \quad (32)$$

where $\alpha > \beta > 0.5$. Thus, $\sum_k (\alpha_k/\beta_k)^p < \infty$ may only be satisfied for a large positive p. Since α_k tends to zero more quickly than β_k, the recursion for actions (28) is said to evolve on a slower time-scale than that for the CV constants (29). By having $U_{1:N,k}$ evolve more slowly than b_k, we allow b_k to 'track' the optimal CV constants, which depend on the point at which the gradient $\widehat{\nabla J}$ is evaluated (see (24)). In the Appendix, we will establish the convergence of algorithm (28)–(30) for the choice of step-sizes in Assumption 1. However, in the numerical example presented in Section 6, we set function $\Gamma(b) = 1$, use constant step-sizes $\alpha_k = \alpha$ and $\beta_k = \beta$ and still demonstrate convergence. For SA in general, decreasing step-sizes are essential for almost sure convergence. If fixed step-sizes are used, then we may still have convergence but now the iterates 'oscillate' about their limiting values with variance proportional to the step-size.

The convergence of a two time-scale SA algorithm related to (28)–(30) was studied in Konda and Tsitsiklis (2003). We may write the slow time-scale process in a more general form than (28) as $\theta_{k+1} = \theta_k + \alpha_{k+1} H_{k+1}$. If the parameter θ_k does not change, say $\theta_k = \theta$ for all k, the process $\{b_k\}$ would converge to some $\bar{b}(\theta)$. When θ_k varies slowly, we would like the process $\{b_k\}$ to track $\bar{b}(\theta_k)$. Under certain regularity assumptions on the process $\{H_k\}$ (Konda & Tsitsiklis, 2003), it can be shown that this would be the case when θ_k did change. As for the convergence of $\{\theta_k\}$, we may use the line of proof in Bertsekas and Tsitsiklis (2000) to show $\lim\inf_k |\nabla(\hat{J}\circ F)(\theta_k)| = 0$. Details are in the Appendix.

5. Application to Observer Trajectory Planning (OTP)

In OTP, we wish to track a manoeuvring target for N epochs. At epoch n, X_n denotes the state of the target, A_n the position of the observer and Y_n the partial observation of the target state, i.e., $Y_n = g(X_n, A_n, V_n)$, where V_n is measurement noise. Typically, the observer has its own motion model and we let X_n^{o} denote state of the observer. The observer state descriptor usually includes its position and therefore A_n corresponds to certain components of X_n^{o}. The aim of OTP is to adaptively manoeuvre the observer to optimise the tracking performance of the target.

In this section, we formalise the OTP problem for a bearings-only observation model as an instance of the sensor scheduling problem in Section 2. We also give results concerning convergence of algorithm (28)–(30) for this application. As we show below, while most existing work in the literature concerns OTP for one observer only, our proposed framework can handle multiple observers simultaneously. There are some convergence issues though, as we point out in the numerical examples in Section 6. Adding more observers can result in an increased number of local minima of the cost function J while gradient-based algorithms are only guaranteed to converge to a local minimum. Also, J can be increasingly flat at the minima, which means varying the trajectory of the observers at any minima will result in only small changes to J. In practice, this can slow down the convergence of SA.

We do not need to specify the target model explicitly. Our only concern is that we can sample from the model. Manouvring targets are often modelled as a *jump Markov linear system* (JMLS) (Doucet, Gordon et al., 2001). The state of the target is comprised of continuous and discrete valued variables, i.e., $X_n = [r_{x,n}, v_{x,n}, r_{y,n}, v_{y,n}, \xi_n]^{\mathrm{T}} \in \mathbb{R}^4 \times \Xi$, where $(r_{x,n}, r_{y,n})$ denotes the target's (Cartesian) coordinates at time n, $(v_{x,n}, v_{y,n})$ denotes the target velocity in the x and y directions and ξ_n denotes the mode of the target, which belongs to a finite set Ξ. The target switches discontinuously, as indicated by ξ_n, between constant velocity manoeuvres. In Section 6, we consider a manoeuvring target in the examples.

As indicated in (9), we require an observer model of the form $A_{1:N} = F(U_{1:N})$, where we exert control on the observer positions $A_{1:N}$ through the variables $U_{1:N}$. For instance, the accelerations of the observer could be determined from the input $U_{1:N}$, which will in turn determine the observer trajectory. The convergence results of Propositions 2 and 3 below do not depend on the specific form of F but only that this function is sufficiently regular. We now give an example of F which is adopted in the numerical section.

Example 1. Let the state of the observer be $X_n^{\mathrm{o}} = [r_{x,n}^{\mathrm{o}}, v_{x,n}^{\mathrm{o}}, r_{y,n}^{\mathrm{o}}, v_{y,n}^{\mathrm{o}}]^{\mathrm{T}}$, with $A_n = [r_{x,n}^{\mathrm{o}}, r_{y,n}^{\mathrm{o}}]^{\mathrm{T}}$. Assume a kinematic model for the evolution of the state,

$$X_{n+1}^{\mathrm{o}} = \underbrace{\begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{=:G} X_n^{\mathrm{o}} + \underbrace{\begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix}}_{=:H} C \operatorname{atan}(U_{n+1}), \tag{33}$$

where the initial state X_0^{o} is fixed, T is the sampling interval, and $U_{n+1} \in \mathbb{R}^2$ determines the acceleration in the x and y directions. We have included the function arctan and the positive diagonal matrix C. The function arctan and its first two derivatives are bounded. Also, arctan is linear around zero and makes a nice choice of saturating function for the acceleration; naturally the acceleration cannot be unbounded. The matrix C alters the saturation behaviour of the acceleration. The observer trajectory is completely determined once X_0^{o} and $U_{1:N}$ are given,

$$A_n = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \left(G^n X_0^{\mathrm{o}} + \sum_{i=1}^{n} G^{n-i} H C \operatorname{atan}(U_i) \right). \tag{34}$$

The function F in (8) is now implicitly defined by (34).

In the bearings-only model, the observation process $\{Y_n\}_{n\geqslant 0}$ $(\subset \mathbb{R})$ is generated according to

$$Y_n = \operatorname{atan}\left(\frac{r_{x,n} - A_n(1)}{r_{y,n} - A_n(2)} \right) + V_n, \tag{35}$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_Y^2)$. In our simulation-based framework, we require the observation process density to be known and differentiable w.r.t. A_n. The bearings-only case is one such example. To present the convergence results of Proposition 2 and 3 below, we will assume that the x and y position of the target corresponds to the first and third component of the state descriptor X_n,

$$X_n = [r_{x,n}, \cdot, r_{y,n}, \ldots]^{\mathrm{T}}, \tag{36}$$

which is usual convention in the literature. The score $S(y,x,a)^{\mathrm{T}}$ is then given by (12)

$$\sigma_Y^{-2}\frac{y-\operatorname{atan}((x(1)-a(1))/(x(3)-a(2)))}{1+((x(1)-a(1))/(x(3)-a(2)))^2}\times\left[\frac{-1}{x(3)-a(2)},\frac{x(1)-a(1)}{(x(3)-a(2))^2}\right].$$

For the case of multiple observers, say p of them, let the position of observer l at epoch n be denoted by $A_n^{(l)}$. Also, assume that each observer measures a bearings angle according to (35) independent of the other observers, i.e., observer l receives the measurement $Y_n^{(l)}$ generated according to model (35) based on its own position $A_n^{(l)}$. We stack these observations together as a vector-valued observation $Y_n=[Y_n^{(1)},\ldots,Y_n^{(p)}]^{\mathrm{T}}$. Likewise, we stack the positions to form the effective position $A_n=[(A_n^{(1)})^{\mathrm{T}},\ldots,(A_n^{(p)})^{\mathrm{T}}]^{\mathrm{T}}$. The observation density for this multiple observer case is $q(Y_n|X_n,A_n)=q(Y_n^{(1)}|X_n,A_n^{(1)})\times\cdots\times q(Y_n^{(p)}|X_n,A_n^{(p)})$. It is apparent that we are now effectively in the original single observer setting and may proceed to solve the multiple OTP problem as above. Note that we are now optimising the tracking performance criterion over the space of possible trajectories for p observers, which implies that the observers cooperate. In the examples of Section 6, we study the two observer case.

5.1. Convergence for bearings-only tracking

For the bearings-only observation model, we have the following sufficient conditions for the convergence of the slow and fast time-scale. The sufficient conditions (37)–(40) are only restrictions on the target state transition model, and the range of the function F that is used to map the sequence $U_{1:N}$ (which could be accelerations) to the observer positions for all N epochs. The proof of this proposition involves verifying the assumptions of Lemmas 4 and 5 in the Appendix. All proofs appear in the technical report (Singh et al., 2005), which is available online or may be obtained by e-mailing any of the authors. The following result concerns the cost function (17) with $\lambda=0$ and can be generalised to any $\lambda\in[0,1]$.

Proposition 2. *Consider algorithm* (28)–(30) *for the bearings-only observation model* (35). *Suppose the following assumptions hold*:

$$\sup_{A_{1:N}\in\operatorname{range}(F)}\mathbf{E}_{A_{1:N}}\{|X_n(3)-A_n(2)|^{-p}\}<\infty,\quad 1\leqslant n\leqslant N,\quad p>0, \tag{37}$$

$$\sup_{A_{1:N}\in\operatorname{range}(F)}\max_l|X_n^{(l)}(3)-A_n(2)|^{-1}<\infty,\quad 1\leqslant n\leqslant N, \tag{38}$$

$$\inf_{\substack{1\leqslant n\leqslant N\\ A_{1:N}\in\operatorname{range}(F)}}\mathbf{E}_{A_{1:N}}\left\{\frac{1/(X_n(3)-A_n(2))^2}{[1+((X_n(1)-A_n(1))/(X_n(3)-A_n(2)))^2]^2}\right\}>0, \tag{39}$$

$$\inf_{\substack{1\leqslant n\leqslant N\\ A_{1:N}\in\operatorname{range}(F)}}\mathbf{E}_{A_{1:N}}\left\{\frac{[(X_n(1)-A_n(1))/(X_n(3)-A_n(2))^2]^2}{[1+((X_n(1)-A_n(1))/(X_n(3)-A_n(2)))^2]^2}\right\}>0. \tag{40}$$

Then, almost surely, $\sup_k|b_k|<\infty$ *and*

$$\lim_k|b_k-\overline{S^2}(A_{1:N,k})^{-1}\overline{S\times h_1}(A_{1:N,k})|=0.$$

Furthermore, if F has bounded second order derivatives then, almost surely, $\lim\inf_k|\nabla(\hat{J}\circ F)(U_{1:N,k})|=0$.

Recall that the expectation operator $\mathbf{E}_{A_{1:N}}\{\cdot\}$ above is an abbreviation for $\mathbf{E}_{(X_{0:N},Y_{1:N})\sim\mathbf{P}_{A_{1:N}}}\{\cdot\}$. Condition (38) relates to the samples used to approximate the posterior density in (14)–(15). Also, the first and third component of the target state is its x and y component, respectively. Note that the proposition does not limit the specific form of function F that relates inputs $U_{1:N}$ to actions $A_{1:N}$. It only restricts range(F) and requires F to be sufficiently regular as specified by the last assumption concerning bounded second order derivatives. For F defined implicitly by (34), this assumption is satisfied.

The next result gives the conditions under which (37)–(40) hold. This result basically says that if the support of $X_{0:N}$ and the range of function F do not intersect, then the assumptions hold and we have the desired convergence of two time-scale SA for OTP. It is interesting to note that the scenario in which the support of $X_{0:N}$ and the range of F do not intersect is a standard setting studied by previous works on OTP for bearings-only observations (see references in the Introduction) and hence the conditions of Proposition 2 are not restrictive for the application.

Proposition 3. *Write the mapping* $F:R^{2N}\to R^{2N}$ *as* $F=[F_{1,1},F_{1,2},\ldots,F_{N,1},F_{N,2}]^{\mathrm{T}}$. (*Note that* $A_n(j)=F_{n,j}(U_{1:N})$.) *Suppose that the density of* $X_{0:N}$, $f(x_{0:N})$, *has a compact support* $\mathscr{K}_f\subset R^{4(N+1)}$. *Furthermore, suppose that for each* $1\leqslant n\leqslant N$, *the compact set* $\mathscr{K}_{f,n}:=\{x_n(3)|x_{0:N}\in\mathscr{K}_f\}$ *does not intersect with the closure of the set* range($F_{n,2}$), *i.e., there exists a compact set* $\mathscr{K}_{A,n}$ *such that* range$(F_{n,2})\subset\mathscr{K}_{A,n}$, *and* $\mathscr{K}_{f,n}\cap\mathscr{K}_{A,n}=\varnothing$. *Then, conditions* (37)–(40) *are satisfied.*

6. Numerical example

The aim of this section is to demonstrate the utility of the proposed simulation-based algorithm for the OTP problem. We demonstrate various convergence aspects of the algorithm and solve for the optimal open-loop observer trajectory under a variety of tracking scenarios, namely, with a single observer and cooperating observers. OLFC is implemented for the cooperating observers.

All examples below for OLFC concern a manoeuvring target where the target follows a linear Gaussian model between manoeuvres. However, when solving for the open-loop trajectory we assume a linear Gaussian model $X_{n+1}=GX_n+HW_n$ with

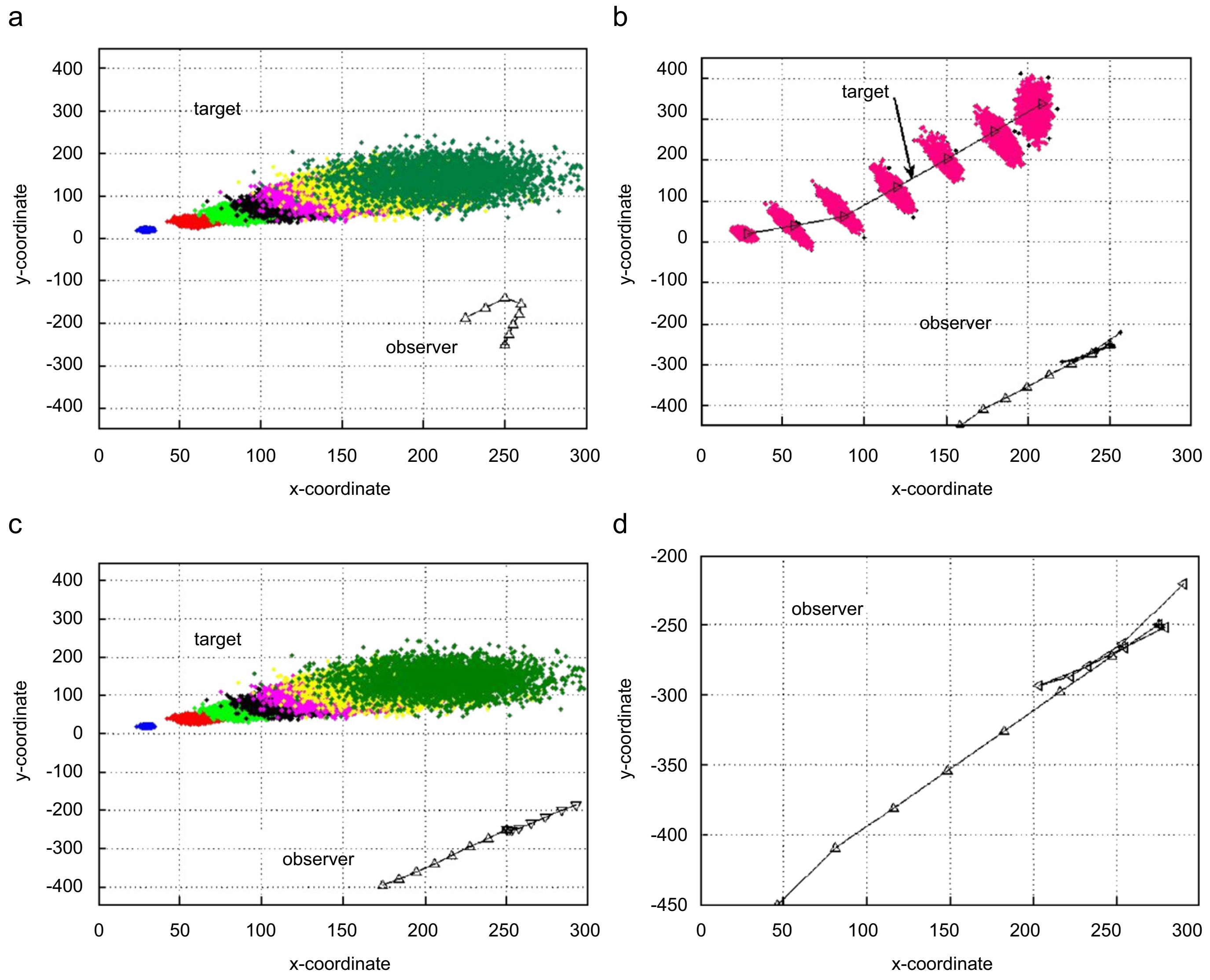

Fig. 1. (a) Single observer open-loop trajectory. Particle clouds are trajectory samples from the target's dynamical model in (3). Shown are many trajectory samples $(X_1, X_2, \ldots, X_N)$ where the various X_1 samples are the blue cloud, the red cloud is X_2, lime is X_3, black X_4 and so on. Target moving northeast and observer moves southwest while doing a hook-turn. (b) Two cooperating observers open-loop trajectory; one moving northeast and the other southwest. Particle cloud as in part (a). (c) Two cooperating observers OLFC trajectory with a manoeuvring target. Particle cloud are samples from the target's filtering density $\{\widehat{\pi}_n\}_{n=0}^N$. (d) Magnification of the OLFC trajectory of the observers. All slow observers commence from coordinate $(250, -250)$. The open-loop trajectories of (a) and (b) took several hours to compute with a 2.8 GHz Pentium 4 CPU.

increased acceleration noise to account for the unmodelled manoeuvres. This is more robust in practice as one usually knows little about the target's precise model. The particle cloud in Fig. 1(a) are independent samples from the target's dynamic model (3) to help visualise it. The target starts at $(0, 0)$ and moves northeast. Shown in Fig. 1(c) is the actual manoeuvring target when the OLFC trajectory is constructed. The manoeuvring target also starts at $(0, 0)$ and moves northeast. For the manoeuvring target, at time $t = 3$, the velocity of the target in the y direction is increased to induce the manoeuvre. Note that the target manoeuvre was intentionally chosen to be far more drastic than that predicted by its model. This was done to contrast the constructed open-loop and OLFC trajectories. In all the examples below, the problem horizon N is 7 and we choose function ψ to be $\psi(X_n) = w_1 X_n(1) + w_2 X_n(3)$ where weights $w_1, w_2 \in [0, 1]$ are selected to trade-off accuracy in tracking the x and y coordinates. The setting for this example is a manoeuvring target that is to be tracked by a single observer and two cooperating observers. The observer motion model is given by as in Ex. 1 Fig. 1(a) shows the optimal open-loop trajectory of one slow observer, and Fig. 1(b) that of two cooperating slow observers. All observers commence at coordinate $(250, -250)$. Note that a single slow observer is obliged to do more manoeuvres to improve the tracking performance since it is significantly more constrained in motion. Fig. 1(c) shows the OLFC trajectory obtained for two cooperating slow observers. A more detailed plot of the OLFC trajectory is shown in Fig. 1(d). Fig. 1(c) also shows the actual target manoeuvre and the particle cloud surrounding it are samples from the filtering density at each time, which is implemented using a particle filter. We note that the OLFC trajectory performs more manoeuvresthan the equivalent open-loop one in order to respond to the actual manoeuvrs of the target. Also, the open-loop trajectory of a single observer differs greatly from that of two cooperating observers. Fig. 2 shows a running plot of the performance criterion, as the (open-loop) trajectories are computed by algorithm (28)–(30), for the single and two cooperating slow observers of Fig. 1(a) and (b). The cost was estimated using Monte Carlo simulation. As Fig. 2 indicates, two observers cooperating during tracking outperforms one. In Fig. 3, the convergence of the trajectory $A_{1:N,k}$ computed using algorithm (28)–(30) for the two slow cooperating observers of Fig. 1(b) is shown. Note that although it converges slowly, there is little gain in performance beyond iteration 1.5×10^6 as indicated by the running cost plot in Fig. 2.

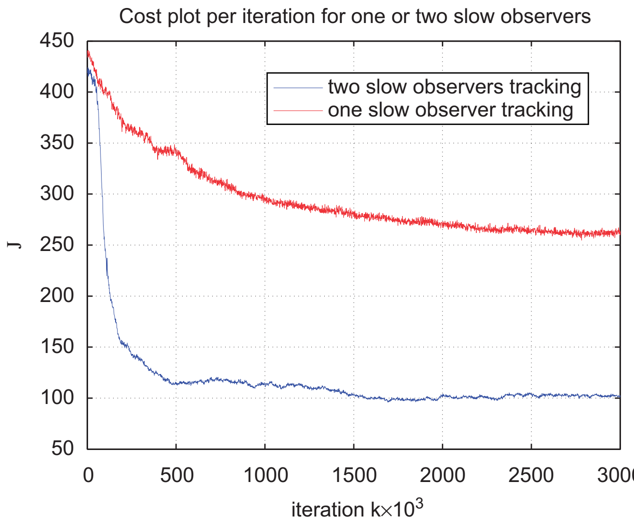

Fig. 2. Plot of performance criterion as actions are iterated by algorithm (28)–(30), for the single and two cooperating observers of Fig. 1(a) and (b). Performance criterion was estimated using Monte Carlo simulation.

6.1. Variance reduction

Here we illustrate the importance of using the control CV variance reduction scheme. In algorithm (28)–(30), we do not update the actions, i.e., $A_{1:N,k} = A_{1:N,0}$ for all k. In Fig. 4(a) we show the gradient estimates without the CV while in Fig. 4(b) we show the gradient estimate as the CV iterated by (29) converges. The convergence of the CV iterated by (29) is shown in Fig. 4(c). Note the significant variance reduction achieved which will speed up the convergence of the actions iterated by (28). The plots only show the gradient of the performance criterion w.r.t. x coordinate of the action at time 1, i.e., $A_1(1)$. The effect of the CV on the remaining components of the performance criterion gradient is similar.

7. Conclusion

In this paper we proposed a novel simulation-based method to solve the sensor scheduling problem for the case in which the state, observation and action spaces are continuous valued vectors. This general continuous state-space case is important as it is the natural framework for many applications, like OTP. We avoided restrictive modelling assumptions on the

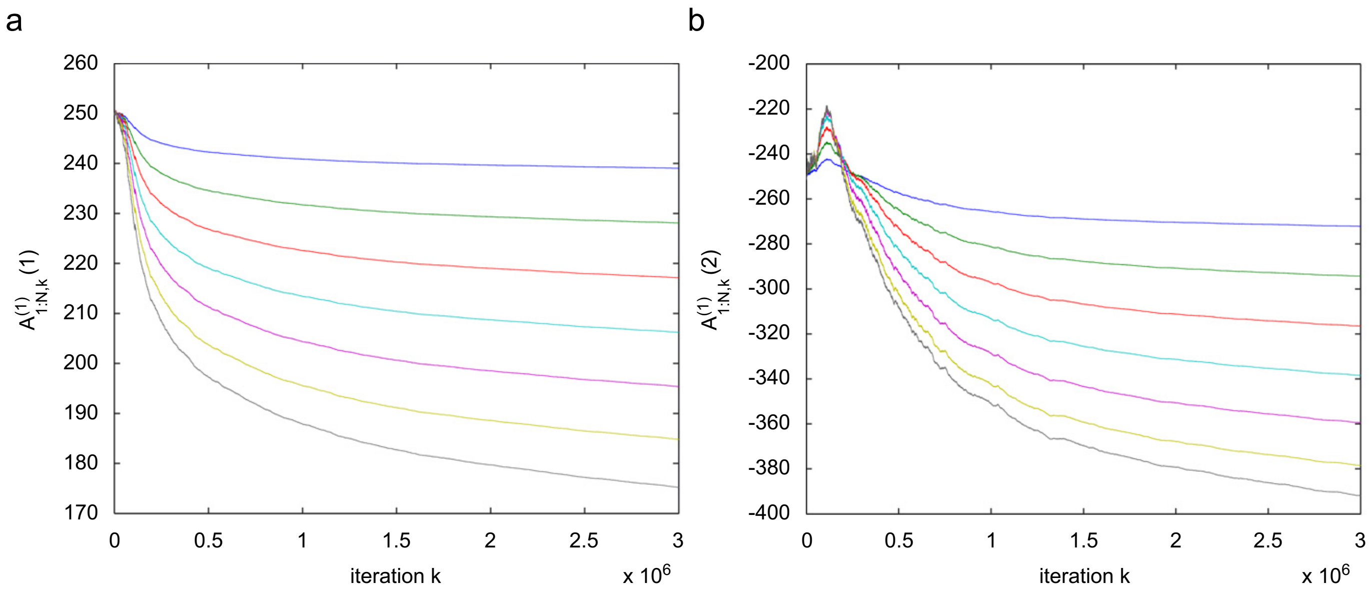

Fig. 3. The convergence of the trajectories $A_{1:N,k}$ computed using algorithm (28)–(30) for the two cooperating observers of Fig. 1(b) is shown. (a) shows the convergence of the x coordinate of the trajectory for observer 1. The blue line is the x position for epoch 1, green for epoch 2, red for epoch 3, cyan for epoch 4 and so on. There are seven epochs in total. (b) shows the convergence of the y coordinate of the trajectory for observer 1. The blue line is the y position for epoch 1, green for epoch 2, red for epoch 3, cyan for epoch 4 and so on. Plots for observer 2's trajectory computed using Algorithm (28)–(30) look similar. (a) Observer 1, x-axis trajectory. (b) Observer 1, y-axis trajectory.

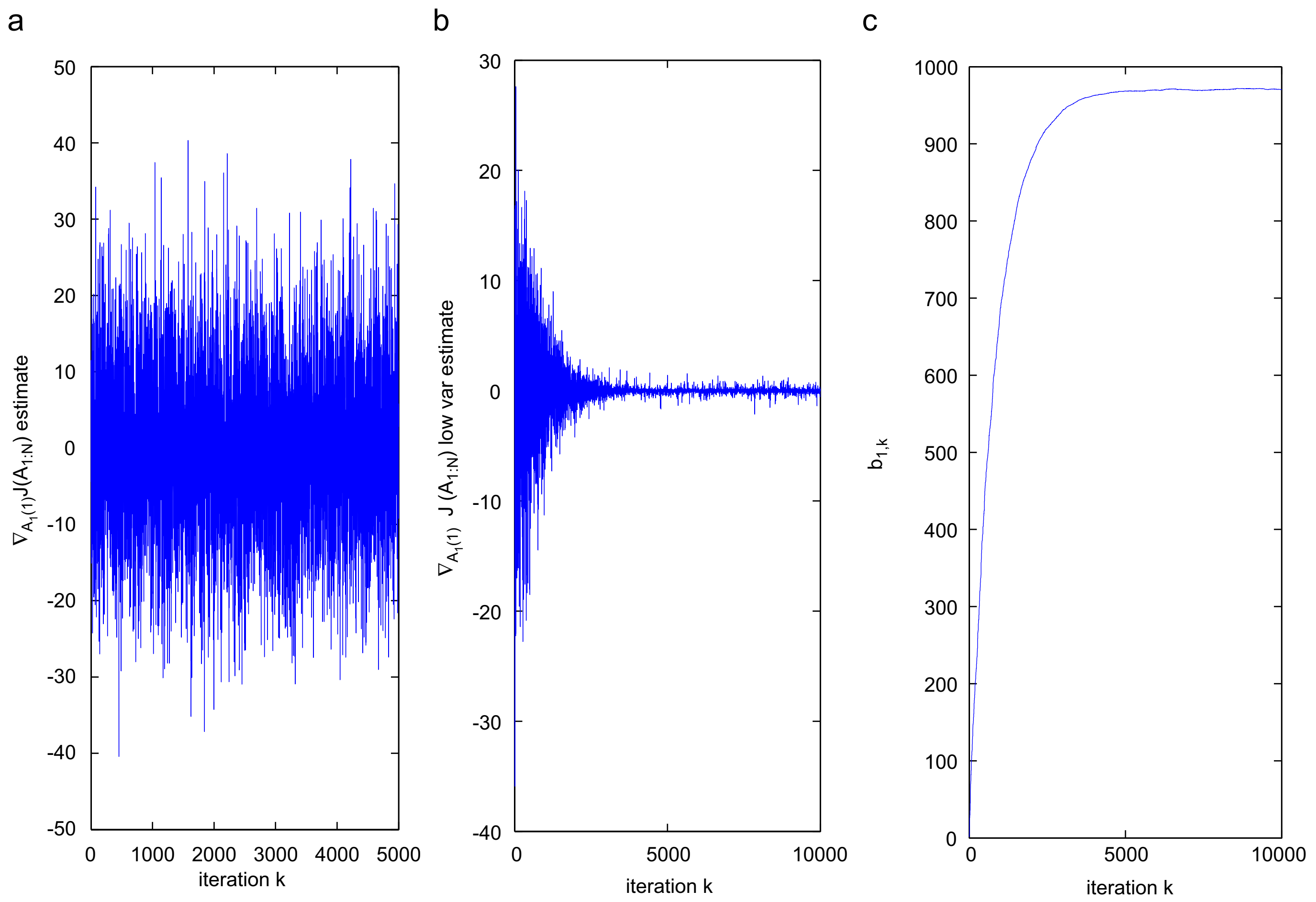

Fig. 4. Variance reduction by control variate: (a) shows gradient estimate without control variate. (b) shows the gradient estimate variance decreasing as the control variate in (c) converges.

continuous state HMM, such as a linear and (or) Gaussian system, by recourse to simulation-based methods. This paper solved the sensor scheduling problem with continuous action space directly, and not a surrogate problem defined through the PCRLB or otherwise, which is the approach adopted in other works. The novel simulation-based method presented used a two time-scale SA algorithm to find the optimal actions. We presented general convergence results for convergence to a local minima of the cost function.

As an application we studied the OTP problem with bearings-only measurements. We established that in the standard scenario where the observer and the target are well separated, our simulation-based algorithm converged to a local minimum of the cost function. In future work we plan to apply our simulation-based method to related problems in robotics and sensor networks.

Appendix A. Convergence

In this section, we study the convergence of the two time-scale SA algorithm (28)–(30). We will first present a result asserting the convergence (or tracking performance) of CV constants iteration (29), in the sense that b_k will follow the optimal CV constants given in (24). We then present a result that concerns the convergence of the iterates $A_{1:N,k}$ to the minimiser of $\hat{J}$. The convergence results presented below will be then specialised to the OTP problem in Section 5.

The convergence of a two time-scale SA algorithm related to (28)–(30) was studied in Konda and Tsitsiklis (2003). The key to studying convergence is to write the slow time-scale process in a more general form than (28) as

$$\theta_{k+1} = \theta_k + \alpha_{k+1} H_{k+1}.$$

If the parameter θ_k does not change, say $\theta_k = \theta$ for all k, the process $\{b_k\}$ would converge to some $\bar{b}(\theta)$. When θ_k varies slowly, we would like the process $\{b_k\}$ to 'track' $\bar{b}(\theta_k)$. Under certain regularity assumptions on the process $\{H_k\}$, it can be shown that this would be the case (Konda & Tsitsiklis, 2003). As for the convergence of $\{\theta_k\}$, we will use the line of proof in Bertsekas and Tsitsiklis (2000) to show $\lim\inf_k |\nabla(\hat{J} \circ F)(\theta_k)| = 0$. Details now follow.

Let $\theta \in \mathbb{R}^d$ and the mapping

$$F = [F_1, \ldots, F_d]^{\mathrm{T}} : \mathbb{R}^d \to \mathbb{R}^d$$

be bounded with bounded partial derivatives, i.e.,

$$\sup_{\theta\in\mathbb{R}^d} |F(\theta)| < \infty, \tag{A.1}$$

$$\sup_{\theta\in\mathbb{R}^d} |\nabla F(\theta)| = |[\nabla F_1(\theta), \ldots, \nabla F_d(\theta)]| < \infty. \tag{A.2}$$

Note that $\nabla F_i = [\partial F_i/\partial\theta_1, \ldots, \partial F_i/\partial\theta_d]^{\mathrm{T}}$ and that implicit in the definition of F is the assumption that $\mathbb{R}^{d_a} = \mathbb{R}^{d_u}$. Denote the range of the function F by range(F). For algorithm (28)–(30), define the σ-algebra

$$\mathscr{F}_k = \sigma\{\theta_0\}, \quad \mathscr{F}_k = \sigma\{\theta_0, \omega_1, \ldots, \omega_k\}, \quad k \geqslant 1.$$

We have θ_k, b_k being $\mathscr{F}_k$-measurable. Let $\mathbf{E}_k(\cdot)$ denote $\mathbf{E}(\cdot|\mathscr{F}_k)$. For each $\theta\in\mathbb{R}^d$, define

$$\bar{h}_i(\theta) = \mathbf{E}_{\omega\sim\mathbf{P}_\theta}(h_{i,\theta}(\omega)), \quad i = 1, 2, \tag{A.3}$$

$$h_\theta(\omega) = h_{1,\theta}(\omega) + h_{2,\theta}(\omega), \tag{A.4}$$

$$\bar{h}(\theta) = \bar{h}_1(\theta) + \bar{h}_2(\theta). \tag{A.5}$$

A.1. Convergence of the fast time-scale

The assumptions below to establish the convergence of process $\{b_k\}$ are essentially the same as in Konda and Tsitsiklis (2003), with some omissions. These are due to the Markov structure of ω_{k+1} in Konda and Tsitsiklis (2003), i.e., ω_{k+1} depends on θ_k and ω_k, while in our case, there is no dependence on ω_k.

Assumption 2. Define the following functions:

$$\overline{S^2}(\theta) = \mathbf{E}_{\omega\sim\mathbf{P}_\theta}(S_\theta^2(\omega)),$$
$$\overline{S\times h_1}(\theta) = \mathbf{E}_{\omega\sim\mathbf{P}_\theta}(S_\theta(\omega)h_{1,\theta}(\omega)).$$

(a) There exists some constant C such that for all $\theta\in$ range(F), we have $\max(|\overline{S^2}(\theta)|, \overline{S\times h_1}(\theta)|) \leqslant C$.
(b) There exists some constant C such that for all $\theta, \theta' \in$ range(F), we have

$$\max(|\overline{S^2}(\theta) - \overline{S^2}(\theta')|, |\overline{S\times h_1}(\theta) - \overline{S\times h_1}(\theta')|) \leqslant C|\theta-\theta'|.$$

(c) For each $p > 0$, there exists a constant $C_p > 0$ such that

$$\sup_k \mathbf{E}(|S^2_{\tilde{\theta}_k}(\omega_{k+1})|^p) < C_p,$$

$$\sup_k \mathbf{E}(|S_{\tilde{\theta}_k}(\omega_{k+1})h_{1,\tilde{\theta}_k}(\omega_{k+1})|^p) < C_p.$$

Assumption 3. Rewriting the iteration for θ_k in (28) as $\theta_{k+1} = \theta_k + \alpha_{k+1}H_{k+1}$, we require $\sup_k \mathbf{E}(|H_k|^p) < \infty$ for all $p > 0$.

Assumption 4 (*Uniform positive definiteness*). The re exists some constant $a > 0$ such that for all $b\in\mathbb{R}^d$, $\theta\in$ range(F),

$$b^{\mathrm{T}}\overline{S^2}(\theta)b \geqslant a|b|^2.$$

The following result follows from Konda and Tsitsiklis (2003, Theorem 7).

Lemma 4 (*Konda and Tsitsiklis, 2003, Theorem 7*). *Consider algorithm* (28)–(30) *under Assumptions* 1–4. *Then, almost surely*, $\sup_k |b_k| < \infty$ *and* $\lim_k |b_k - \overline{S^2}(\tilde{\theta}_k)^{-1}\overline{S\times h_1}(\tilde{\theta}_k)| = 0$.

A.2. Convergence of the slow time-scale

We now present a result concerning the convergence of $\widehat{\nabla J}(\theta_k)$. The proof of this result can be established using the same ideas in Bertsekas and Tsitsiklis (2000) and we refer the reader to the technical report (Singh et al., 2005), which is available online, for details.

Lemma 5. *Consider the recursion for* θ_k *in* (28) *re-written as* $\theta_{k+1} = \theta_k + \alpha_{k+1}H_{k+1}$ *where*

$$H_{k+1} = -\Gamma(b_k)\nabla F(\theta_k)\bar{h}(\theta_k) + W_{k+1},$$

and noise term

$$\begin{aligned} W_{k+1} = &-\Gamma(b_k)\nabla F(\theta_k)(h_{1,F(\theta_k)}(\omega_{k+1}) - \bar{h}_1(F(\theta_k)))\\ &-\Gamma(b_k)\nabla F(\theta_k)(h_{2,F(\theta_k)}(\omega_{k+1}) - \bar{h}_2(F(\theta_k)))\\ &+\Gamma(b_k)\nabla F(\theta_k)S_{F(\theta_k)}(\omega_{k+1})b_k\end{aligned}$$

satisfies $\mathbf{E}_k(W_{k+1}) = 0$. *Assume*:

(a) $\nabla F(\theta)\bar{h}(F(\theta))$ $(= \nabla(\hat{J}\circ F)(\theta))$ *satisfies* $\sup_\theta|\nabla(\hat{J}\circ F)(\theta)| < \infty$, *and, for some constant* C *and all* θ, θ',

$$|\nabla(\hat{J}\circ F)(\theta) - \nabla(\hat{J}\circ F)(\theta')| < C|\theta-\theta'|.$$

(b) *That* $\sup_k \mathbf{E}(|H_k|^2) < \infty$ *and almost surely*,

$$\sup_k \mathbf{E}\{|\nabla F(\theta_k)(h_{1,F(\theta_k)}(\omega_{k+1}) - \bar{h}_1(F(\theta_k)))|^2|\mathscr{F}_k\} < \infty, \tag{A.6}$$

$$\sup_k \mathbf{E}\{|\nabla F(\theta_k)(h_{2,F(\theta_k)}(\omega_{k+1}) - \bar{h}_2(F(\theta_k)))|^2|\mathscr{F}_k\} < \infty, \tag{A.7}$$

$$\sup_k \mathbf{E}\{|\nabla F(\theta_k)S_{F(\theta_k)}(\omega_{k+1})|^2|\mathscr{F}_k\} < \infty. \tag{A.8}$$

(c) $\sup_k|b_k| < \infty$ *almost surely. Then, if* $\hat{J}\circ F$ *is bounded below*,

$$\lim_k \inf |\nabla(\hat{J}\circ F)(\theta_k)| = 0 \quad \textit{almost surely}.$$

References

Bertsekas, D. P. (1995). *Dynamic programming and optimal control*. Belmont: Athena Scientific.

Bertsekas, D. P., & Tsitsiklis, J. N. (2000). Gradient convergence in gradient methods with errors. *SIAM Journal on Optimization*, *10*(3), 627–642.

Blackman, S., & Popoli, R. (1999). *Modern tracking systems*. Artech House.

Del Moral, P. (2004). *Feynman–Kac formulae: Genealogical and interacting particle systems with applications*. New York: Springer.

Doucet, A., De Freitas, J. F. G., & Gordon, N. J. (2001). *Sequential Monte Carlo methods in practice*. New York: Springer.

Doucet, A., Gordon, N. J., & Krishnamurthy, V. (2001). Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, *49*(3), 613–624.

Glynn, P. W., & Szechtman, R. (2002). Some new perspectives on the method of control variates. In: K. T. Fang, F. J. Hickernell, & H. Niederreiter (Eds.), *Monte Carlo and quasi-Monte Carlo methods 2000* (pp. 27–49). Berlin: Springer.

Hauskrecht, M. (2000). Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, *13*, 33–94.

Hernandez, M. L., Kirubarajan, T., & Bar-Shalom, Y. (2004). Multisensor resource deployment using posterior Cramer–Rao bounds. *IEEE Transactions on Aerospace and Electronic and Systems*, *40*(2),

Hernandez-Lerma, O. (1989). *Adaptive Markov control processes*. New York: Springer.

Kershaw, D. J., & Evans, R. J. (1994). Optimal waveform selection for target tracking. *IEEE Transactions on Information Theory*, *40*(5), 1536–1550.

Konda, V. R., & Tsitsiklis, J. N. (2003). Linear stochastic approximation driven by slowly varying Markov chains. *Systems and Control Letters*, *50*(2), 95–102.

Lee, H. W. J., Teo, K. L., & Lim, E. B. (2001). Sensor scheduling in continuous time. *Automatica*, *37*, 2017–2023.

Logothetis, A., Isaksson, A., & Evans, R. J. (1997). An information theoretic approach to observer path design for bearings-only tracking. *IEEE conference on decision control* (pp. 3132–3137).

Meier, L., Perschon, J., & Dressler, R. M. (1967). Optimal control of measurement systems. *IEEE Transactions on Automatic Control*, *12*(5), 528–536.

Paris, S., & Le Cadre, J. P. (2002). Planification for terrain-aided navigation. In *Proceedings of the international conference on information fusion* (pp. 1007–1014). Annapolis, MD.

Pflug, G. Ch. (1996). *Optimization of stochastic models: The interface between simulation and optimization*. Boston: Kluwer Academic Publishers.

Poyiadjis, G., Singh, S. S., & Doucet, A. (2006). Particle filter as a controlled Markov chain for on-line parameter estimation in general state space models. In *Proceedings of the international conference on acoustics, speech, and signal process* (ICASSP).

Singh, S. S., Kantas, N., Vo, B., Doucet, A., & Evans R. (2005). *On the convergence of a stochastic optimisation algorithm for optimal observer trajectory planning*. Technical Report CUED/F-INFENG/TR 522, Department of Engineering, University of Cambridge. URL: ⟨www-sigproc.eng.cam.ac.uk/∼nk234/⟩.

Singh, S. S., Vo, B., Doucet, A., & Evans, R. (2004). Variance reduction for Monte Carlo implementation of adaptive sensor management. In *Proceedings of the international conference on information fusion* (pp. 901–908). Stockholm, Sweden.

Tremois, O., & Le Cadre, J. P. (1999). Optimal observer trajectory in bearings-only tracking for manoeuvring sources. *IEE Proceedings, Radar, Sonar Navigation*, *146*(1), 31–39.

Sumeetpal S. Singh received the B.E. (with first-class honours) and Ph.D. degrees from the University of Melbourne, Melbourne, Australia, in 1997 and 2002, respectively. From 1998 to 1999, he was a Communications Engineer in industry. From 2002 to 2004, he was a Research Fellow in the Department of Electrical and Electronic Engineering, University of Melbourne. He joined the Engineering Department of Cambridge University in 2004 as a Research Associate and is currently a University Lecturer in Engineering statistics. His research interests include Monte Carlo methods for estimation and control.

Nikolaos Kantas was born in Athens, Greece, in 1981. He completed his undergraduate education in Cambridge University Engineering Department, where he is currently working towards a Ph.D. degree on Monte Carlo methods for estimation and control.

Ba-Ngu Vo was born in 1970 in Saigon Vietnam. He received his Bachelor degrees jointly in Science and Electrical Engineering with first-class honours at the University of Western Australia in 1994, and a Ph.D. at Curtin University of Technology in 1997. Since 2000, he has been with the Department of Electrical and Electronic Engineering at the University of Melbourne where he is currently an Associate Professor. Dr. Vo's research interests include optimisation, signal processing and stochastic geometry.

Arnaud Doucet was born in France in 1970. He received the Ph.D. in Information Engineering from University Paris XI (Orsay) in 1997. From 1998 to 2000, he was a Research Associate in Cambridge University Engineering Department. From 2001 to 2002, he was Senior Lecturer in the Department of Electrical Engineering of Melbourne University. From 2002 to 2005, he was University Lecturer in Cambridge University Engineering Department. Since 2005, he is Associate Professor and Canada Research Chair in the Department of Statistics and the Department of Computer Science of the University of British Columbia. His main research interests are Bayesian statistics and Monte Carlo methods.

Robin Evans was born in Melbourne, Australia, in 1947. After completing a B.E. degree in Electrical Engineering at the University of Melbourne in 1969, he worked as a Radar Systems Engineering Officer with the Royal Australian Airforce. He completed a Ph.D. in 1975 at the University of Newcastle followed by postdoctoral studies at the Laboratory for Information and Decision Systems, MIT, USA, and the Control and Management Department, Cambridge University, UK. He has held various academic positions including Head of the Department of Electrical and Computer Engineering at the University of Newcastle, Head of the Department of Electrical and Electronic Engineering at the University of Melbourne, Research Leader for the Cooperative Centre for Sensor Signal and Information Processing and Director of the Centre for Networked Decision Systems. He is currently Director of the Victoria Research Laboratory of National ICT Australia. His research has ranged across many areas including theory and applications in industrial control, radar systems, signal processing and telecommunications. He is a Fellow of the Australian Academy of Science, a Fellow of the Australian Academy of Technological Sciences and Engineering and a Fellow of the Institution of Electrical and Electronic Engineers, USA.