Parameter Estimation of State Space Models Using Particle Importance Sampling

Yuxiong Gao
University of Manchester
Manchester, UK

Wentao Li
University of Manchester
Manchester, UK

Rong Chen
Rutgers University
New Jersey, USA

Abstract

State-space models have been used in many applications, including econometrics, engineering, medical research, etc. The maximum likelihood estimation (MLE) of the static parameter of general state-space models is not straightforward because the likelihood function is intractable. It is popular to use the sequential Monte Carlo(SMC) method to perform gradient ascent optimisation in either offline or online fashion. One problem with existing online SMC methods for MLE is that the score estimators are inconsistent, i.e. the bias does not vanish with increasing particle size. In this paper, two SMC algorithms are proposed based on an importance sampling weight function to use each set of generated particles more efficiently. The first one is an offline algorithm that locally approximates the likelihood function using importance sampling, where the locality is adapted by the effective sample size (ESS). The second one is a semi-online algorithm that has a computational cost linear in the particle size and uses score estimators that are consistent. We study its consistency and asymptotic normality. Their computational superiority is illustrated in numerical studies for long time series.

1 INTRODUCTION

State-space models are a class of stochastic models in which the observations $\{y_t\}_{t=1}^T$ are based on unobservable latent state variables $\{x_t\}_{t=1}^T$ through some probabilistic dependence. This class of models has been extensively studied due to its wide applications in science and engineering. In this paper, we mainly consider the state-space models in discrete time and have Markov property:

$$x_t|x_{0:t-1} \sim f_\theta(\cdot|x_{t-1}),\ y_t|x_{0:t}, y_{0:t-1} \sim g_\theta(\cdot|x_t), \quad (1)$$

where $t \in \mathbb{Z}^+$, $x_{0:t}$ refers to $x_0, \ldots, x_t$, θ is the parameter vector, $f_\theta(x_t|x_{t-1})$ and $g_\theta(y_t|x_t)$ are probability densities. The question of interest here is the true parameter $\theta^\star$. Since $\{x_t\}_{t=1}^T$ are not observable, the likelihood function $p_\theta(y_{0:T})$ is intractable in the sense that a direct attempt to calculate the likelihood function may involve high-dimensional integration over $\{x_t\}_{t=1}^T$, except for the linear Gaussian state-space model (Kalman, 1960). Sequential Monte Carlo (SMC) is a popular class of methods that approximate the likelihood by generating Monte Carlo samples, so-called 'particles', for the latent variables and assigning them with weights corresponding to the sequential importance sampling with resampling (Andrieu et al., 2005). This work focuses on the MLE of the true parameter value $\theta^\star$.

Given a parameter value θ, the SMC method provides particles targeting the conditional distribution $p_\theta(x_{0:T}|y_{0:T})$ (Kantas et al., 2015) and the estimated likelihood (Douc and Moulines, 2008). It is well-known that the estimated likelihood function is not continuous due to the independent Monte Carlo noise on each SMC run and the discreteness of the resampling step (Doucet et al., 2023). To apply gradient-based optimizations, the score function is often estimated using Fisher's identity when the gradients of the state and observation densities can be evaluated easily, as given below,

$$\nabla_\theta l(\theta) = \int \nabla_\theta \log p_\theta(x_{0:T}, y_{0:T}) p_\theta\left(x_{0:T}|y_{0:T}\right)\, dx_{0:T}, \quad (2)$$

where the log-likelihood function $l(\theta) \triangleq \log p_\theta(y_{0:T})$. Given particles from $p_\theta\left(x_{0:T}|y_{0:T}\right)$, (2) can be estimated consistently and the steepest gradient ascent (SGA) can be used. Alternatively, (2) can be estimated using a recursive expression without needing to store the full paths of particles in the cost of $O(N^2)$ computational complexity (Poyiadjis et al., 2011). However,

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

both methods require a new set of particles to be generated for different θ, and the computational cost is sensitive to the optimization scheme. For example, particles may be repeatedly generated for similar parameter values if the step size is small, or for similar likelihood values in regions where parameter values are close to unidentifiable.

Sometimes it is more efficient to update θ when processing the data on the fly, e.g. when the early part of the observed series is informative about the parameter. Based on the decomposition $p_\theta(y_{0:T}) = \prod_{t=1}^{T} p_\theta(y_t|y_{0:t-1})p_\theta(y_0)$ and the ergodicity assumption on $y_{0:T}$, by iterating the following update from $t = 0$ to $t = T - 1$,

$$\theta_{t+1} = \theta_t + \gamma_t \nabla_\theta \log p_\theta(y_t|y_{0:t-1})|_{\theta=\theta_t}, \tag{3}$$

it is expected that θ_T will have the same asymptotic behaviour as the MLE. The challenge is that consistent estimation of the conditional score function $\nabla_\theta \log p_\theta(y_t|y_{0:t-1})$ using SMC requires particles simulated with the parameter value θ_t different at each time t, and regenerating particles at every t is contrary to the purpose of using (3) for the online update. Existing online SMC implementations of (3) bypass this problem by propagating the particles under the 'time-varying' parameter $\theta_{0:n}$ (Poyiadjis et al., 2006, 2011; Nemeth et al., 2016). For finite state-space hidden Markov models, it was shown that such an algorithm converges toward $\theta^\star$ under regularity conditions (LeGland and Mével, 1997). For general state-space models, however, there is no theoretical support and it contains a non-vanishing bias as N increases. It is also sensitive to the initialisation of θ which needs to be close to the true parameter $\theta^\star$. This is illustrated numerically in Section 4.1.

This paper proposes to recycle generated particles for performing multiple updates in SGA using (2) and a novel semi-online SMC algorithm that performs (3) under varying parameter values with consistent estimators, exploiting the fact that for $x_{0:t} \sim p_{\theta_0}(x_{0:t}|y_{0:t})$, $x_{0:t}$ weighted with the importance sampling weight, defined as

$$a_{\theta_0}(\theta, x_{0:t}) \triangleq p_\theta(x_{0:t}, y_{0:t})/p_{\theta_0}(x_{0:t}, y_{0:t}), \tag{4}$$

follows the density $p_\theta(x_{0:t}|y_{0:t})$. Numerical experiments show that the new methods significantly improve the standard SGA by up to one order of magnitude, and are competitive to existing state-of-the-art algorithms. There are three main contributions. First, for the offline implementation, the score function in a neighbourhood around θ_0 is approximated using particles weighted with $a_{\theta_0}(\theta, x_{0:t})$, and the neighbourhood is determined by thresholding ESS of $a_{\theta_0}(\theta, x_{0:t})$ to avoid the poor approximation when θ is far way from θ_0. Second, a combination of the online gradient ascent and a SMC algorithm that renews part of the particle path is proposed, which provides a consistent estimator of the conditional score function and has a computational complexity of $O(N)$. Specifically, at the kth iteration, the weights of particles targeting $p_{\theta_{k-1}}(x_{0:k}|y_{0:k})$ are multiplied with $a_{\theta_{k-1}}(\theta_k, x_{0:k})$, so that the weighted particles are 'retargeted' to the new parameter value. A renewing step is introduced in which, if the quality of the particles is below a certain threshold, an SMC algorithm is run to regenerate particles targeting $p_{\theta_k}(x_{0:k}|y_{0:k})$. Third, the consistency and asymptotic normality of the conditional score estimator as N goes to infinity is stated, which provide justification for the form of the thresholding ESS in the renewing step.

Notations Capital letters such as X, Y represent random variables and lowercase x, y their corresponding values. The latent variable is denoted by $X \in \mathcal{X}$ and the observation by $Y \in \mathcal{Y}$. The sequence of latent variables $x_t, \ldots, x_{t'}$ is denoted by $x_{t:t'}$, and similarly the observations $y_t, \ldots, y_{t'}$ by $y_{t:t'}$, where $t' \geq t$ are non-negative integers. The static parameter $\theta \in \mathbb{R}^p$ in (1) is of interest. The density of the initial latent variable X_0 at value x_0 is denoted by $f_\theta(x_0)$. For any function h of θ we denote its gradient with respect to its argument θ as $\nabla h(\theta)$, the i^{th} coordinate of the gradient as $\nabla_i h(\theta)$, and the value of the gradient at θ_0 as $\nabla h(\theta)|_{\theta=\theta_0}$ or $\nabla h(\theta_0)$.

1.1 Maximum Likelihood Estimation of State-space Models Using SMC

At each time point t, for a given θ and $t_0 < t$, the conditional likelihood $p_\theta(y_{(t_0+1):t} \mid y_{0:t_0})$ can be written by the Markovian structure as follows,

$$\frac{p_\theta(y_{0:t})}{p_\theta(y_{0:t_0})} = \int p_\theta(x_{(t_0+1):t}, y_{(t_0+1):t}|x_{t_0}) p_\theta(x_{t_0}|y_{0:t_0}) dx_{t_0:t}, \tag{5}$$

where $p_\theta(x_{(t_0+1):t}, y_{(t_0+1):t}|x_{t_0})$ can be factorised as $\prod_{j=t_0}^{t-1} f_\theta(x_{j+1}|x_j) g_\theta(y_{j+1}|x_{j+1})$. An online gradient ascent to approximately maximise $p_\theta(y_{0:T})$ using the SMC method on (3) is given in Algorithm 1. If the step size $a_t \equiv 0$, the parameter value is kept fixed and Algorithm 1 reduces to the vanilla SMC algorithm which can be used for offline MLE with (2).

The ESS above is defined as $ESS(w_{1:N}) \triangleq 1/\sum_{i=1}^{N} \overline{w}_i^2$ for a set of importance weights $w_{1:N}$, where $\overline{w}_i$ is the normalised weights defined as $w_i/\sum_{j=1}^{N} w_j$ (Martino et al., 2017). Algorithm 1 is taken by the literature which propose estimators of $\nabla \log p_\theta(y_{t+1}|y_{0:t})$ for the online updates (Chopin et al., 2020; Nemeth et al., 2016; Poyiadjis et al., 2011). When $\theta_t \equiv \theta$, at each time t, the weighted particles $(x_{0:t}^{(i)}, w_t^{(i)})$ and $(\widetilde{x}_{0:t}^{(i)}, \widetilde{w}_t^{(i)})$ approx-

Algorithm 1 Online gradient ascent / vanilla SMC

Assume that the initial value θ_0, the threshold $r_2 \in (0,1)$ and the step sizes $\{\gamma_t\}_{n=0}^T$ are given. At $t=0$, the proposal distribution is $q_0(x_0)$. Let $\widetilde{w}_{-1}^{(i)} = 1$, $i = 1, \cdots, N$.

For $t = 0$ to $T-1$:

Propagation & Weighting: For $i = 1, \cdots, N$, sample $x_t^{(i)} \sim q_t(x_t|\widetilde{x}_{0:t-1}^{(i)})$, calculate $u_t^{(i)} = \frac{p_{\theta_t}(x_t^{(i)}, y_t|\widetilde{x}_{0:t-1}^{(i)})}{q_t(x_t^{(i)}|\widetilde{x}_{0:t-1}^{(i)})}$ and let $w_t^{(i)} = u_t^{(i)}\widetilde{w}_{t-1}^{(i)}$, $x_{0:t}^{(i)} = (\widetilde{x}_{0:t-1}^{(i)}, x_t^{(i)})$.

Conditional GA: Set $\theta_{t+1} = \theta_t + \gamma_t \widehat{\nabla} \log p_\theta(y_t|y_{0:t-1})|_{\theta=\theta_t}$.

Resampling: If $\mathrm{ESS}(w_t^{1:N})/N \leq r_2$, resample $\{x_{0:t}^{(i)}\}_{i=1}^N$ with weights $\{w_t^{(i)}\}_{i=1}^N$ to get $\{\widetilde{x}_{0:t}^{(i)}\}_{i=1}^N$ and set all $\widetilde{w}_t^{(i)} = 1$. Otherwise set $\{(\widetilde{x}_{0:t}^{(i)}, \widetilde{w}_t^{(i)})\}_{i=1}^N = \{(x_{0:t}^{(i)}, w_t^{(i)})\}_{i=1}^N$.

End For

imately follow the density $p_\theta(x_{0:t}|y_{0:t})$, $i = 1, \cdots, N$. Therefore (5) can also be estimated asymptotically unbiasedly using the particles, and $l(\theta)$ can be estimated by $\widetilde{l}(\theta) \triangleq \Sigma_{j=1}^{k+1} \log(N^{-1}\Sigma_{i=1}^N w_{t_j}^{(i)})$ where $t_1, \cdots, t_{k+1}$ are the times when resampling is performed.

However, when θ_t is updated online, the particles instead follow the density $p_{\theta_{0:t}}(x_{0:t}|y_{0:t})$ where the parameter value of the state and observation densities varies over t. For example, at time $t=1$, the particle $x_{1:2}^{(i)}$ weighted with $w_2^{(i)}$ targets the density proportional to $p_{\theta_1}(x_2, y_2|x_1)p_{\theta_0}(x_1|y_1)$, which means the marginal density of weighted $x_2^{(i)}$ is biased towards the filtering density of x_2 at the parameter value θ_1. Thus, the conditional score cannot be estimated consistently and the estimated MLE contains a non-vanishing bias. Algorithm 1 seems to perform well empirically in the literature, but requires the initial parameter θ_0 to be close to the true value which is usually found using the a short segment of the data.

On the other hand, the development of Bayesian inference methods for the static parameter is very active, including both online and offline methods (Carvalho et al., 2010; Andrieu et al., 2010; Chopin et al., 2013; Rosato et al., 2022). One method of updating the posterior distribution of θ online is to approximate the likelihood at the proposed θ_t using that at θ_{t-1} by controlling $\|\theta_t - \theta_{t-1}\|$ to be small (Crisan and Míguez, 2018; Crisan and Miguez, 2017; Pérez-Vieites et al., 2018). It is not in our purpose to promote MLE over Bayesian inference of the static parameter, but to provide an alternative to existing offline and online SMC MLE estimators for users who prefer an alternative to choosing a prior distribution for θ and tuning and assessing the mixing Markov chain sampler when performing Bayesian inference.

2 NEW ALGORITHMS

2.1 Adaptive Gradient Ascent Using ESS

Following the importance sampling form below,

$$\frac{p_\theta(y_{0:T})}{p_{\theta_0}(y_{0:T})} = \int a_{\theta_0}(\theta, x_{0:T}) p_{\theta_0}(x_{0:T}|y_{0:T})\, dx_{0:T}, \quad (6)$$

the Fisher's identity (2) can be extended to give the score estimator

$$\widehat{\nabla} l_{\theta_0}(\theta) \triangleq \frac{\sum_{i=1}^N \nabla \log p_\theta(x_{0:T}^{(i)}, y_{0:T}) a_{\theta_0}(\theta, x_{0:T}^{(i)}) w_T^{(i)}}{\sum_{i=1}^N a_{\theta_0}(\theta, x_{0:T}^{(i)}) w_T^{(i)}}.$$

The approximation is accurate when θ is close to θ_0, but unreliable when θ is far away, because particles approximating the distribution p_{θ_0} may not cover the high-density region of p_θ. It is natural to use $\widehat{\nabla} l_{\theta_0}(\theta)$ in the neighbourhood of θ_0 adaptively determined by the quality of the importance weights. Based on this we introduce an offline algorithm in Algorithm 2 that generalises the SGA .

Algorithm 2 Adaptive gradient ascent with particle importance sampling (adaptGA-PIS)

Assume that the initial value θ_0, a threshold $r \in (0,1)$ and the step sizes $\{\gamma_n\}_{n=0}^I$ are given.

For $n = 0$ to the maximum iterations I:

1. Run an SMC algorithm from $t=0$ to T with parameter θ_n to obtain the weighted particles $\{(x_{0:T}^{(i)}, w_T^{(i)})\}_{i=1}^N$. Let $\theta_{n,1} = \theta_n$ and $k = 1$.
2. **While** $\mathrm{ESS}(\{a_{\theta_n}(\theta_{n,k}, x_{0:T}^{(i)})\}_{i=1}^N) > rN$ and $\theta_{n,k}$ does not converge: Let $\theta_{n,k+1} = \theta_{n,k} + \gamma_n \widehat{\nabla} l_{\theta_n}(\theta)|_{\theta=\theta_{n,k}}$, where $\widehat{\nabla} l_{\theta_n}(\theta)$ estimates $\nabla l(\theta)$ using (6), and $k = k+1$.
3. Let $\theta_{n+1} = \theta_{n,k}$. Stop if θ_n converges, otherwise go to step 1.

End For

One commonly used SMC algorithm is given in the appendix. The value of ESS is equal to 1 when the weights are all equal and close to 0 when a few weights dominate the others, which, in our context, corresponds to $\theta = \theta_0$ and θ is far away from θ_0. We provide more details on the choice of step sizes γ_n and the tuning for r in Appendix D.

If in step 2 ESS is removed from the stopping rules, the algorithm reduces to the Monte Carlo maximum likelihood (MCML) (Geyer and Thompson, 1992) or the SGA if step 2 ends at $k=2$. Compared to MCML,

the unreliable score estimates are avoided. Compared to the SGA, the additional computational cost of Algorithm 2 is negligible when the cost of evaluating $\{a_{\theta_n}(\theta, x_{0:T}^{(i)})\}_{i=1}^N$ is negligible compared to that of generating particles. In this case, Algorithm 2 is computationally more efficient since one set of particles is used for multiple gradient ascent steps, particularly when small step sizes are used, e.g. in the vicinity of MLE.

When T is large, intuitively it is sufficient that $\theta_{n,k} - \theta_n = O(T^{-1/2})$ for the ESS to be away from 0 by noting the following: in the expression of $\widehat{\nabla} l_{\theta_n}(\theta)$, by the central limit theorem over the weighted particles, the normalised importance weight can be expanded as follows,

$$\begin{aligned}\frac{a_{\theta_n}(\theta, x_{0:T})}{\sum_{i=1}^N a_{\theta_n}(\theta, x_{0:T}^{(i)}) w_T^{(i)}} &= \frac{p_\theta(x_{0:T}|y_{0:T})}{p_{\theta_n}(x_{0:T}|y_{0:T})}\{1 + O_p\left(1/\sqrt{N}\right)\} \\ &= \exp\left[\left\{\nabla_\theta \log p_{\dot\theta}(x_{0:T}, y_{0:T}) - \nabla_\theta \log p_{\dot\theta}(y_{0:T})\right\}(\theta - \theta_n)\right] \\ &\cdot \{1 + O_p\left(1/\sqrt{N}\right)\}, \end{aligned} \tag{7}$$

which is in the order of $\exp\{O_p(\sqrt{T}(\theta - \theta_n))\}$, since the two score functions have the form of summation and are in order of $\sqrt{T}$ under standard regularity conditions for the state-space model. The second equality of (7) holds by the Taylor expansion and $\dot\theta$ is an intermediate value between θ_n and θ. Since the gradient and Hessian of the log-likelihood function typically has the order $O(\sqrt{T})$ and $O(T)$ respectively (Durbin and Koopman, 2012; Hamilton, 2020), one set of particles can be recycled for multiple times within the range of a Newton-Raphson iteration. This is illustrated in the numerical experiments with $T = 10000$.

It is challenging to use the particle-based SGA if some parameters are hardly identifiable, because $l(\theta)$ is nonconcave and flat in some region of the parameter space. The SGA may repeatedly generate particles at parameter values where the likelihood changes little, so its computational cost is sensitive to the initialisation and the choice of step size. In contrast, Algorithm 2 renews the particles less in such regions as the joint density $p_\theta(x_{0:T}, y_{0:T})$ is not sensitive to the change of θ, hence is more robust to the choice of step size by adapting to the curvatures. See Figure 2 for an illustration.

2.2 Semi-online Gradient Ascent Controlled by ESS

In Algorithm 2, each gradient ascent step is conditional on the entire sequence of data. If the data is long or the early part of the data is informative about the parameter, it may be cheaper to update the parameter value using (3). At each iteration which is also time t, to obtain particles following the filtering density at the latest parameter value, we propose to adjust the particle weight with $a_{\theta_t}(\theta_{t+1}, x_{0:t})$, which gives Algorithm 3.

Algorithm 3 Semi-online gradient ascent with particle importance sampling (semiGA-PIS)

Assume that the initial value θ_0, thresholds $r_1, r_2 \in (0,1)$ and the step sizes $\{\gamma_t\}_{t=0}^T$ are given. At $t = 0$, the proposal distribution is $q_0(x_0)$. Let $\widetilde{w}_{-1}^{(i)} = 1$, $i = 1, \cdots, N$.

For $t = 0$ to $T - 1$:

Propagation & Weighting: Same as that in Algorithm 1.

Conditional GA:
Set $\theta_{t+1} = \theta_t + \gamma_t \widehat{\nabla} \log p_\theta(y_t|y_{0:t-1})|_{\theta=\theta t}$.

Retarget: Multiply $w_t^{(i)}$ by $a_{\theta_t}(\theta_{t+1}, x_{0:t}^{(i)})$ to obtain $\widetilde{w}_t^{(i)}$, $i = 1, \cdots, N$.

Renewing:
If $\sum_{k=t-K+1}^t \mathrm{ESS}(a_{\theta_k}(\theta_{k+1}, x_{0:t}^{1:N}))/N \le r_1$, run an SMC algorithm to obtain a new set of particles targeting $p_{\theta_{t+1}}(x_{0:t}|y_{0:t})$, denoted by $\{\widetilde{x}_{0:t}^{(i)}\}_{i=1}^N$. Set all $\widetilde{w}_t^{(i)} = 1$.

Resampling: If $\mathrm{ESS}(\widetilde{w}_t^{1:N})/N \le r_2$, resample $\{x_{0:t}^{(i)}\}_{i=1}^N$ with weights $\{\widetilde{w}_t^{(i)}\}_{i=1}^N$ to get $\{\widetilde{x}_{0:t}^{(i)}\}_{i=1}^N$ and set all $\widetilde{w}_t^{(i)} = 1$. Otherwise set $\{\widetilde{x}_{0:t}^{(i)}\}_{i=1}^N = \{x_{0:t}^{(i)}\}_{i=1}^N$.

End For

The conditional GA step is the same as that in Algorithm 1 and the same conditional score estimator therein can be used. In our experiment, both algorithms use the difference $\widehat{\nabla} \log p_\theta(y_{0:t}) - \widehat{\nabla} \log p_\theta(y_{0:(t-1)})$ where each partial score is estimated using (2). Compared to Algorithm 1, it adds the retarget step which gives $\{x_{0:t}^{(i)}, \widetilde{w}_t^{(i)}\}_{i=1}^N$ approximately following $p_{\theta_{t+1}}(x_{0:t}|y_{0:t})$. Thus, at each iteration, the propagation and weighting step is conditional on particles from the filtering density having the correct parameter value. Unlike Algorithm 1, the new algorithm gives a consistent estimator of $\nabla \log p_\theta(y_t|y_{0:t-1})|_{\theta=\theta t}$ as $N \to \infty$, whereas that in Algorithm 1 is inconsistent. For the tuning of r_1 see Appendix D for more details.

At iteration t, if θ_{t+1} is very different from some earlier parameter values θ_k, $k < t+1$, the particles may suffer from particle degeneracy. Because $x_k^{(i)}$ generated with θ_k when conditioned on $y_{0:k}$ may be in tail areas of $p_{\theta_{t+1}}(x_k|y_{0:t})$, when the information of later observations are included. The renewing step is introduced to regenerate the entire particle trajectory up to the current iteration if the particles degenerate severely due to the changes in parameter values. This degeneracy is different from that caused by the weighting step and is monitored by the quality of $a_{\theta_t}(\theta_{t+1}, x_{0:t})$. To see this,

note that when θ_k is fixed, there is still particle degeneracy but no regeneration is performed. This can also be seen in (8). The ESS of $a_{\theta_t}(\theta_{t+1}, x_{0:t}^{1:N})$ measures the variation of changes of $p_{\theta_t}(x_{0:t}^{(i)}, y_{0:t})$ when θ_t is changed to θ_{t+1} and controls the resulting additional variance that is added to the estimator $\widehat{\nabla}\log p_\theta(y_{t+1}|y_{0:t})|_{\theta=\theta_{t+1}}$. When particles are in tail areas of $p_{\theta_t}(x_{0:t}|y_{0:t})$, the change is likely to vary greatly and trigger the renewal of particles. Between two renewals, early part of the particles usually degenerate under repeated resampling. Denote t_0 as the largest index where $x_{0:t_0}^{(i)}$ have an identical value denoted by $x_{0:t_0}^{(0)}$. If $t_0 > 0$, the ESS of $a_{\theta_t}(\theta_{t+1}, x_{0:t}^{1:N})$ measures the variation of changes of $p_{\theta_t}(x_{(t_0+1):t}^{(i)}, y_{(t_0+1):t}|x_{0:t_0}^{(0)}, y_{0:t_0})$ instead. When the particles are in tail areas, the skewness can still be detected with sufficient diversity on the dimensions $x_{(t_0+1):t}$ despite the coalescence on the dimensions $x_{0:t_0}$. In the renewing step, the average of the most recent several ESS values is used for stability purpose.

Both Algorithm 1 and the online part of Algorithm 3 cost $O(TN)$ for length-T data. The new algorithm is less sensitive to the initial value θ_0 than Algorithm 1 and gives the consistent estimator by using the offline renewal with the additional cost $O(\sum_j^d t_j N)$ if it occurs at time $t_1, \cdots, t_d$, hence it is identified as 'semi-online'. However, it does not mean Algorithm 3 has an overall cost of $O(T^2N)$, since the renewal steps are not necessarily a recurrent or regular process proportional to iteration T. The frequency depends on r_1 and the sensitivity of the joint density to the change of the parameter value. The renewal occurs relatively frequently in the early iterations where the gradients are of a considerable scale and only on an occasional basis when θ is in close proximity to the MLE where the gradient is relatively small. Algorithm 1 has the following benefits. First, similarly to Algorithm 2, the frequency of regenerating the particle trajectories is adaptive to the likelihood surface, but in an online manner, by being less frequent in the region where the joint density is flat. Second, Algorithm 3 does not need a pilot phase for initialisation which Algorithm 1 needs. Third, if $p_\theta(x_{0:t}, y_{0:t})$ depends on the latent states through a fixed d-dimensional summary statistics as that of models considered in Fearnhead (2002), only a $d \times N$ matrix needs to be stored and the $O(tN)$ cost at iteration t can be avoided, preventing the total cost from being $O(T^2N)$. These are illustrated in the numerical studies with large T values.

3 ASYMPTOTIC PROPERTY

For Algorithm 2, since the weighted particles are from some existing SMC algorithm, $\widehat{\nabla} l_{\theta_n}(\theta)$ satisfies the corresponding central limit theorem (Chopin, 2004), and has a finite asymptotic variance with the order $O(N^{-1})$ given that ESS is away from 0. Here, we focus on Algorithm 3. Since the stability of the conditional gradient ascent update is determined by the variance of $\widehat{\nabla}\log p_\theta(y_t|y_{0:t-1})$, below the asymptotic variance of estimator using $\{(x_{0:t}^{(i)}, w_t^{(i)})\}_{i=1}^N$ is given to highlight the impact of the time-varying θ_t and justify that it can be controlled using the averaged ESSs in the renewing step.

Suppose multinomial resampling is used at every step, and the renewal does not occur until time t. Denote $p_{\theta_k}(\cdot|y_{0:t})$ by $\pi_{\theta_k}^t(\cdot)$.

Theorem 1. *For a test function f and the initialisation of particles satisfying certain regularity conditions(see assumption B in the supplementary materials), conditional on $\theta_1, \cdots, \theta_t$, as $N \to \infty$ we have:*

$$\sqrt{N}\left(\frac{\Sigma_{i=1}^N w_t(x_{0:t}^{(i)}) f(x_{0:t}^{(i)})}{\Sigma_{i=1}^N w_t(x_{0:t}^{(i)})} - \mathbb{E}(\pi_{\theta_t}^t(f))\right) \xrightarrow{D} N(0, V_t(f)),$$

where $V_t(f) = \int \frac{\pi_{\theta_t}^t(x_0)^2}{q_0(x_0)} \cdot$

$$\left(\int f(x_{0:t})\pi_{\theta_t}^t(x_{1:t}|x_0)dx_{1:t} - \mathbb{E}_{\pi_{\theta_t}^t}(f(x_{0:t}))\right)^2 dx_0$$

$$+ \Sigma_{k=1}^t \int \frac{\pi_{\theta_t}^t(x_{0:k})^2}{q_k(x_k|x_{0:k-1})\pi_{\theta_k}^{k-1}(x_{0:k-1})} \cdot$$

$$\left(\int f(x_{0:t})\pi_{\theta_t}^t(x_{k+1:t}|x_{0:k})dx_{k+1:t} - \mathbb{E}_{\pi_{\theta_t}^t}(f(x_{0:t}))\right)^2 dx_{0:k}.$$

As a result, in Algorithm 3, as $N \to \infty$,

$$\widehat{\nabla}\log p_\theta(y_t|y_{0:(t-1)}) \xrightarrow{P} \nabla \log p_\theta(y_t|y_{0:(t-1)}).$$

The complete statement is presented in the supplements. The theorem extends the standard asymptotic variance result (Chopin, 2004) in that each importance sampling variance term contains the time-indexed parameter in both the target densities $\pi_{\theta_t}^t(x_{0:k})$ and the proposal densities $q_k(x_k|x_{0:k-1})\pi_{\theta_k}^{k-1}(x_{0:k-1})$ over $k = 0, \cdots, t$. The variance increases as θ_t increasingly differs from θ_k with k fixed as t increases, since the importance weight in the kth term is proportional to

$$\frac{p_{\theta_k}(x_k, y_k|x_{0:k-1}, y_{0:k-1})}{q_k(x_k|x_{0:k-1})} \cdot \frac{p_{\theta_t}(x_{0:k}, y_{0:t})}{p_{\theta_k}(x_{0:k}, y_{0:k})}. \quad (8)$$

The increase in variance comes from the second term.

4 NUMERICAL EXPERIMENTS

This section presents a comparative analysis of the performance of *adaptGA-PIS* and *semiGA-PIS* using the state-of-the-art (SoTA) algorithms from Poyiadjis

et al. (2011), denoted by P-offline and P-online, as the benchmarks. We also compare their performance with *Naive SGA* and *Fisher SGA*, which are the SGA algorithms using particles from the vanilla SMC in Algorithm 1. The difference is that the former estimates the score function using finite difference ignoring the discontinuity of the SMC likelihood, and the latter uses the Fisher's identity (2). The semiGA-PIS algorithm is compared with Algorithm 1, named *vanilla onlineGA*, to illustrate its robustness. The algorithms are tested on three models: the noisy auto-regressive(AR) model, the stochastic volatility (SV) model and the Poisson auto-regressive(PAR) model.

The root mean square errors (RMSE) with respect to the MLE are reported. In the plots, the markers indicate the times at which values of the estimated parameters are output. For all algorithms, the multinomial resampling is performed at every step, i.e. $r_2 = 1$. The step sizes of the online algorithm are γ_t multiplied by the data length T so that the gradient updates in all algorithms have a similar scale. Relaxing r_1 to some reasonable value such as 0.5 can further reduce the computational cost of Algorithm 3. The algorithms are compared with the same CPU times which are measured using the *time.perf_counter*() command in Python. The choices of learning rates in all models based on the guidelines from Spall (1998): γ_t is in the form of $\frac{c_1}{(A+t)^\alpha}$ with $\alpha = 1$. The joint choice of c_1, A are based on the magnitude of the gradient and the scale of the parameter. The learning rates are reasonable and fair in the sense that their scales are sufficient for multiple algorithms to converge. More details including raw data and Python code can be found in the supplementary materials.

4.1 Auto-regressive Model of Order One with Noise

Consider the univariate AR(1) model with added normal noise, given below,

$$X_{t+1} = \phi X_t + \sigma_x \eta_t,\ Y_t = X_t + \sigma_y \xi_t, \quad t = 0, \dots, T, \tag{9}$$

where η_t, ξ_t are independent standard Gaussian noise, X_0 has mean 0 and follows the stationary distribution of X_t. The parameter of interest is $\theta \triangleq (\phi, \sigma_x, \sigma_y)$ and its MLE can be computed by the Kalman filter. The data length is chosen to be a moderately large value $T = 10000$ which is challenging to using the importance sampling weights. The particle size $N = 1000$ for all algorithms, and for Algorithm 3 we set $r_1 = 0.5$.

First, we compare all algorithms except Algorithm 1. The results are presented below:

The table shows that our proposed semi-online algorithm has the lowest RMSE among all algorithms. The

Table 1: RMSE ratios for the AR(1) experiment from 20 replications averaged over all parameters. For all algorithms $N = 1000$ except P-offline and P-online which costs $O(N^2)$, while for the two we choose $N = 30$ due to the computational cost restriction. The initial parameters $\theta_0 = (0.5, 0.5, 0.7)$, $r = 0.2$ for Algorithm 2 and $r_1 = 0.5$ for Algorithm 3. RMSEs are calculated with respect to $\theta_{MLE} = (0.66825, 0.73900, 0.95750)$ estimated by the Kalman filter. The CPU time is 130 seconds.

Algorithms	ϕ	σ_x	σ_y
P-offline $O(N^2)$	4.106	1.514	1.945
P-online $O(N^2)$	1.000	1.000	1.000
Naive SGA	6.900	6.393	10.13
Fisher SGA	5.556	5.522	9.29
AdaptGA-PIS	1.461	1.539	1.830
semiGA-PIS	0.568	0.595	0.612

algorithm performed 393 renewing steps in total over the 20 replications. The P-online has the second lowest RMSE. Our proposed adaptGA-PIS algorithm performs the best among all offline algorithms, maintaining a lower RMSE with a given computational budget, followed by P-offline. The naive SGA performs the worst due to the discontinuity of the SMC likelihood $\widetilde{l}(\theta)$, and although the Fisher SGA solves this issue, it does not have much improvement.

Second it is shown that Algorithm 3 is more robust than Algorithm 1. The following model is tested where a degenerating trend term is added to Y_t:

$$X_{t+1} = \phi X_t + \sigma_x \eta_t,\ Y_t = 3\phi^t + X_t + \sigma_y \xi_t, \quad t = 0, \dots, T. \tag{10}$$

In model (10), the early observations are highly informative about θ by the trend term. The result is in Figure 1.

In both plots, Algorithm 1 detects the correct gradient in early updates but eventually fails due to the bias in the particle trajectory. In the right plot although the initial value is fairly close to ϕ_*, it fails significantly because the wrong ϕ also appears in the observation equation and is more misleading. Furthermore, the experiment is run for 20 replications. We counted the 'failure' of Algorithm 1, defined by that $\theta_{end} \notin (0.6, 1.3)$ or is a *nan* value. Among the 20 replications, Algorithm 1 has 5 failures for model (9) and 16 for (10), and Algorithm 3 has no failure. Similarly, for model (9), the RMSE of Algorithm 3 over Algorithm 1 at the end of the iterations is $0.1129/0.2544 = 0.44$, and that for model (10) is $0.0140/0.3250 = 0.043$ due to the many failures of Algorithm 1. More results are in the supplementary materials. The results above show that the negative impact of the inconsistent conditional score estimator in Algorithm 1 can be significant when the parameter value trajectory drifts away from $\theta^\star$. In

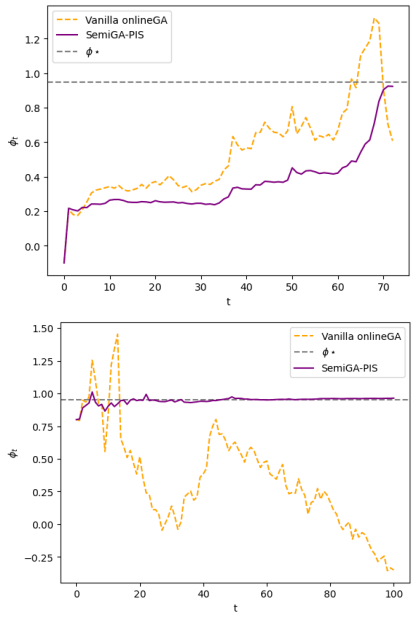

Figure 1: Trajectory plots for one replication of model (9) (Left) and model (10) (Right), with initial values -0.1 and 0.8 respectively. Both models have $\phi_\star = 0.95$ and $\sigma_x = 0.5$ and $\sigma_y = 0.5$.

practice, it is preferable to use batch data to perform offline algorithms for initialisation, but the pilot stage introduces additional computational cost, and for high-dimensional parameters, the initialisation may still be far from the MLE.

4.2 Stochastic Volatility Model

The SV model is a popular class of models to capture the stylised fact of volatility clustering in financial time series. Consider the following univariate SV model from Sandmann and Koopman (1998):

$$\begin{aligned} X_{t+1} &= \phi X_t + \sigma_x \eta_t, \ t = 0, \dots, T-1, \\ Y_t &= \sigma_y e^{\frac{X_t}{2}} \xi_t, \ t = 0, \dots, T, \end{aligned} \tag{11}$$

where X_0 has mean 0 and η_t, ξ_t follow $N(0,1)$. The parameter to estimate is $\theta = (\phi, \sigma_x, \sigma_y)$. Since the SV model is non-linear and non-Gaussian, the Kalman filter can not be applied. The RMSE is calculated with respect to the estimated value $\hat{\theta}_{MLE} = (0.896, 0.399, 0.243)$, which is obtained by running Fisher SGA with large values of N and iteration numbers initialised from the true parameter value. The RMSE ratios are shown in Table 2.

Table 2 shows that both new algorithms converge faster than the SoTA online algorithm in all parameters. The AdaptGA-PIS algorithm has the lowest RMSE in ϕ and σ_y, yet its estimation for σ_x is considerably worse than the SemiGA-PIS algorithms. The Naive SGA performs the worst, with the Fisher SGA improves moderately.

Table 2: RMSE ratios for the SV experiment from 50 replications averaged over all parameters.The true parameter $\theta^\star = (0.9, 0.40, 0.25)$ and $T = 10000$. For all algorithms $N = 1000$ except P-offline and P-online which costs $O(N^2)$, while for the two we choose $N = 30$ due to the computational cost restriction. The initial parameters $\theta_0 = (0.7, 0.25, 0.40)$, $r = 0.4$ for Algorithm 2 and $r_1 = 0.6$ for Algorithm 3. The semi-online algorithm performs 6665 renewing steps over the 50 replications. The CPU time is 3000 seconds. The results using $N = 400$ (and $N = 20$ for $O(N^2)$ the algorithms) are similar

Algorithms	ϕ	σ_x	σ_y
P-offline $O(N^2)$	2.377	2.043	0.222
P-online $O(N^2)$	1.000	1.000	1.000
Naive SGA	4.535	2.531	1.989
Fisher SGA	3.908	1.681	0.899
AdaptGA-PIS	0.165	0.798	0.039
SemiGA-PIS	0.226	0.268	0.064

The P-offline algorithm performs better than the Fisher SGA and Naive SGA overall with the $O(N^2)$ cost.

It is known that ϕ is difficult to identify when the other parameters have certain values (Chopin et al., 2020, Chapter 14). Below the behaviours of our algorithms in regions with flat likelihood surfaces stated back in Section 3 are demonstrated by the figures below. We noticed that indeed Algorithm 2 required fewer SMC runs to regenerate particles when moving on the flat surface, and when there are significant changes in likelihood the algorithm detects it and ensures the following iterations are in the right direction. Algorithm 3 has a similar behaviour and we can see clearly from the second plot that between the second renewing step and the third renewing step, the parameter is not moving in the right direction since the gradient estimates are based on the particles generated under the second renewing step. But the algorithm detected the issue and after the third renewing step the parameter started to update in the right gradient. This is another example where our algorithm is more robust than Algorithm 1, where there is no renewing step and in early iterations the gradient updates may suffer from the bad quality of the particles.

Similar to model (10), a model with a degenerating trend added to the mean volatility is tested. The RMSE of the semi-online algorithm significantly outperforms all others by up to one order of magnitude. Again it shows the benefit of the exact online updates when early observations are highly informative. Specific results are in the supplementary materials.

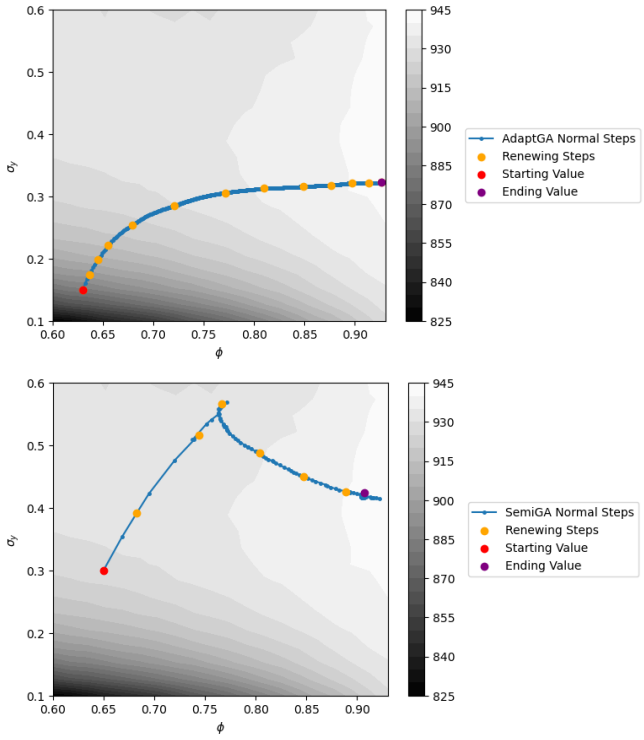

Figure 2: Trajectory plots for one replication of the algorithms run on the model (11). Here $T = 100$ and $\sigma_x = 0.4$ is known, so the parameter of interest is $\theta \triangleq (\phi, \sigma_y)$. The plot on the top is the trajectory for AdaptGA-PIS and the plot on the bottom is the trajectory for SemiGA-PIS. The initial values are $\theta_0 = (0.63, 0.15)$ and $\theta_0 = (0.65, 0.3)$ respectively. The MLE estimated by a sufficient number of Fisher SGA is $\theta^\star = (0.91, 0.39)$. Full settings can be found in the code.

4.3 Poisson Auto-regressive Model

Consider the real-world time series of 168 monthly counts of poliomyelitis in the United States from January 1970 to December 1983, introduced by Zeger (1988). The dataset is well studied using the Poisson Auto-regressive model of order 1(PAR(1)) with a deterministic trend function nonlinear in t, where there are eight unknown parameters $\theta = (\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \phi, \sigma_x)$ (Langrock, 2011). Our implementation of the SGA algorithm with 2000 iterations gives parameter estimates consistent with the literature and is used to calculate the RMSEs. Since P-offline and P-online have no benefits for short time series given the $O(N^2)$ cost, they are not compared here. Occasionally the naive SGA updates move to the wrong direction due to the discontinuity of $\widetilde{l}(\theta)$, and give invalid values for the parameters. The next least feasible random seed is used in such cases. The semi-online algorithm is used as a benchmark and the results are in Table 3.

The AdaptGA-PIS algorithm shows superior performance by having a low RMSE in all parameters. The SemiGA-PIS algorithm is inferior here due to the short data size ($T = 168$). This can be seen by that with $O(TN)$ computational cost, the semi-online algorithm moves T steps while the offline algorithms move 1 step, so a larger T may give more advantage of the semi-online algorithm over the offline algorithms.

Table 3: RMSE ratios for the PAR(1) experiment from 20 replications averaged over all parameters. For all algorithms $N = 3000$, the initial parameters $\theta_0 = (0.4, -3.8, 0.2, -.4, 0.5, -0.1, 0.7, \sqrt{0.4})$, $r = 0.6$ for Algorithm 2 and $r_1 = 0.6$ for Algorithm 3. The CPU time is 900 seconds.

Algorithms	μ_1	μ_2	μ_3	μ_4
Naive SGA	3.829	1.242	1.266	1.590
Fisher SGA	0.519	0.944	0.960	1.196
AdaptGA-PIS	0.674	0.8645	0.276	0.455
SemiGA-PIS	1.000	1.000	1.000	1.000
Algorithms	μ_5	μ_6	ϕ	σ
Naive SGA	2.1955	2.4885	1.7565	1.240
Fisher SGA	1.462	1.646	1.337	0.745
AdaptGA-PIS	0.258	0.383	0.6165	0.491
SemiGA-PIS	1.000	1.000	1.000	1.000

5 CONCLUSIONS

Two SMC algorithms are proposed to utilize the importance weight $a_{\theta_0}(\theta, x_{0:t})$ for gradient-based likelihood optimization. In offline gradient ascent the computational efficiency can be significantly improved by recycling generated particles for multiple updates where the approximation accuracy is maintained by thresholding the ESS. The semi-online algorithm performs parameter updates within the SMC steps and re-targets the particle mass using $a_{\theta_t}(\theta_{t+1}, x_{0:t})$ to achieve consistent estimation. The benefits of controlled particle quality in offline updates and the efficiency of online updates are combined by regenerating the particles when the averaged ESSs is low which approximately controls the asymptotic variance of the conditional score estimator. Both algorithms are robust to the choice of step size and flat likelihood surface, and show competitive performance to SoTA algorithms in numerical studies with long time series.

This work offers several interesting avenues for further research. One is to modify the structure of the semi-online algorithm with different control measures for the renewing step, which may be more computationally efficient than the ESS. Another is to use the semi-online algorithm for deterministic optimisation of criterion function which can be expressed in the form of a state-space model.

LIMITATIONS The two new algorithms are suitable for models where simulating particles is more expensive than evaluating the joint log-likelihood. For example, for each model in the numerical study, the joint likelihood depends on the latent states through a d-dimensional summary statics with d fixed as t increases, hence only a $d \times N$ matrix needs to be stored and updated to evaluate the ESSs as each algorithm iterates. This feature exists for a class of state-space model and is often utilized for MCMC moves in SMC algorithms (Fearnhead, 2002; Kantas et al., 2015). For other models, e.g. a $t \times N$ matrix needs to be stored and updated as t increases, the performance of two algorithms may drop and the memory requirement needs to be considered. The tuning of the ESS thresholds r_1, r_2 in the online algorithm is model-dependent and may require extra work. Theorem 1 considers a simplified case that the central limit theorem at time t is conditional on the trajectory of θ. Since the trajectory depends on the particle path, the unconditional result is more involved than Theorem 1.

Acknowledgements

We thank the reviewers and the Area Chair for their insightful comments.
We thank Nemeth Christopher for providing the R code for Poyiadjis' benchmark algorithms.
Yuxiong Gao was funded by the 'Dean's Doctoral Scholarship' from the University of Manchester.

References

Andrieu, C., Doucet, A., and Godsill, S. (2005). On sequential Monte Carlo sampling methods for Bayesian filtering. *Readings in Unobserved Components Models*, page 418.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 72(3):269–342.

Carvalho, C. M., Johannes, M. S., Lopes, H. F., and Polson, N. G. (2010). Particle learning and smoothing. *Statistical Science*, 25(1):88–106.

Chopin, N. (2004). Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *Annals of Statistics*, pages 2385–2411.

Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2013). Smc2: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 75(3):397–426.

Chopin, N., Papaspiliopoulos, O., et al. (2020). *An introduction to sequential Monte Carlo*. Springer.

Crisan, D. and Miguez, J. (2017). Uniform convergence over time of a nested particle filtering scheme for recursive parameter estimation in state-space markov models. *Advances in Applied Probability*, 49(4):1170–1200.

Crisan, D. and Míguez, J. (2018). Nested particle filters for online parameter estimation in discrete-time state-space markov models. *Bernoulli*, 24(4A):3039–3086.

Douc, R. and Moulines, É. (2008). Limit theorems for weighted samples with applications to sequential monte carlo methods. *Annals of Statistics*, 36(5):2344–2376.

Doucet, A. and Johansen, A. (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3.

Doucet, A., Moulines, E., and Thin, A. (2023). Differentiable samplers for deep latent variable models. *Philosophical Transactions of the Royal Society A*, 381(2247):20220147.

Durbin, J. and Koopman, S. J. (2012). *Time series analysis by state space methods*, volume 38. OUP Oxford.

Fearnhead, P. (2002). Markov chain monte carlo, sufficient statistics, and particle filters. *Journal of Computational and Graphical Statistics*, 11(4):848–862.

Geyer, C. J. and Thompson, E. A. (1992). Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 54(3):657–683.

Hamilton, J. D. (2020). *Time series analysis*. Princeton university press.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45.

Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J., and Chopin, N. (2015). On particle methods for parameter estimation in state-space models. *Statistical Science*, 30(3):328–351.

Langrock, R. (2011). Some applications of nonlinear and non-Gaussian state–space modelling by means of hidden Markov models. *Journal of Applied Statistics*, 38(12):2955–2970.

LeGland, F. and Mével, L. (1997). Recursive estimation in hidden Markov models. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 4, pages 3468–3473. IEEE.

Martino, L., Elvira, V., and Louzada, F. (2017). Effective sample size for importance sampling based on discrepancy measures. *Signal Processing*, 131:386–401.

Nemeth, C., Fearnhead, P., and Mihaylova, L. (2016). Particle approximations of the score and observed information matrix for parameter estimation in state–space models with linear computational cost. *Journal of Computational and Graphical Statistics*, 25(4):1138–1157.

Pérez-Vieites, S., Mariño, I. P., and Míguez, J. (2018). Probabilistic scheme for joint parameter estimation and state prediction in complex dynamical systems. *Physical Review E*, 98(6):063305.

Poyiadjis, G., Doucet, A., and Singh, S. S. (2011). Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80.

Poyiadjis, G., Singh, S. S., and Doucet, A. (2006). Gradient-free maximum likelihood parameter estimation with particle filters. In *2006 American Control Conference*. IEEE.

Rosato, C., Devlin, L., Beraud, V., Horridge, P., Schön, T. B., and Maskell, S. (2022). Efficient learning of the parameters of non-linear models using differentiable resampling in particle filters. *IEEE Transactions on Signal Processing*, 70:3676–3692.

Sandmann, G. and Koopman, S. J. (1998). Estimation of stochastic volatility models via Monte Carlo maximum likelihood. *Journal of Econometrics*, 87(2):271–301.

Spall, J. (1998). Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):817–823.

Zeger, S. L. (1988). A regression model for time series of counts. *Biometrika*, 75(4):621–629.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Yes]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Parameter Estimation in State Space Models Using Particle Importance Sampling: Supplementary Materials

A THE SMC FRAMEWORK

We have introduced the vanilla SMC as a case in Algorithm 1, here we provide an alternative expression which may be clearer for some readers: At each time t, the weighted particles$\{(\tilde{x}_{0:t}^{(I)}, \tilde{w}_t^{(i)})\}_{i=1}^N$ is an approximation of

Vanilla sequential Monte Carlo

Assume that f, g, θ, $y_{0:T}$ and the number of particles needed N are given, and a proposal distribution for generating new latent variables $q_\theta(x_t|x_{0:(t-1)})$ is available. For $t = 0$, the proposal distribution is $q_\theta(x_0)$), and we set all $\tilde{w}_{-1}^{(i)} = 1$, $i = 1, \cdots, N$

For $t = 0$ to T:

Propagation: Use the proposal distribution $q_\theta(x_t^{(i)}|\tilde{x}_{0:(t-1)}^{(i)})$ to sample the new latent variable, and let $x_{0:t}^{(i)} = (\tilde{x}_{0:(t-1)}^{(i)}, x_t^{(i)})$, $i = 1$ to N.

Reweightning: Compute $u_t^{(i)}$, the *incremental weight* for the particle, defined as:

$u_t^{(i)} \triangleq \frac{g_\theta(y_t|x_t) f_\theta(x_t^{(i)}|\tilde{x}_{t-1}^{(i)})}{q_\theta(x_t^{(i)}|\tilde{x}_{0:(t-1)}^{(i)})}$, $i = 1$ to N.

Set the particle $x_{0:t}^{(i)}$ with weight w_t^i where $w_t^{(i)} \triangleq u_t^{(i)} \tilde{w}_{t-1}^{(i)}$

Resampling: If the condition for resampling is satisfied, resample $\{x_{0:t}^{(i)}\}_{i=1}^N$ according to corresponding weights $\{w_t^{(i)}\}_{i=1}^N$ to obtain $\{\tilde{x}_{0:t}^{(i)}\}_{i=1}^N$ and set all $\{\tilde{w}_t^{(i)}\}_{i=1}^N$ equals 1; if the condition for resampling is not satisfied, set $\{\tilde{x}_{0:t}^{(i)}\}_{i=1}^N = \{x_{0:t}^{(i)}\}_{i=1}^N$ and $\{\tilde{w}_t^{(i)}\}_{i=1}^N = \{w_t^{(i)}\}_{i=1}^N$

End For

the samples from distribution $p_\theta(x_{0:t}|y_{0:t})$. Therefore, for any function $h(x_{0:t})$, we could estimate the expectation $E_\theta(h(x_{0:t})|y_{0:t})$ by $\frac{\Sigma_{i=1}^N h(\tilde{x}_{0:t}^{(i)})\tilde{w}_t^{(i)}}{\Sigma_{i=1}^N h(\tilde{w}_t^{(i)})}$. Suppose that the resampling happens at time $t_j, j = 1, \cdots, k$, and we define $t_{k+1} \triangleq T$. The estimate of the likelihood is given by: $\tilde{p}_\theta(y_{0:T}) = \Pi_{j=1}^{k+1}(N^{-1}\Sigma_{i=1}^N w_{t_j}^{(i)})$. It was shown that both $(\Sigma_{i=1}^N w_t^{(i)})/N$ and $\tilde{p}_\theta(y_{0:T})$ are unbiased estimators and are asymptotic normal with convergence rate $\sqrt{N}$ as $N \to \infty$(Kantas et al., 2015). We can then estimate the log-likelihood by the particles :

$$\tilde{l}(\theta) \triangleq \log(\tilde{p}_\theta(y_{0:T})) = \Sigma_{j=1}^{k+1} \log(N^{-1}\Sigma_{i=1}^N w_{t_j}^{(i)}), \tag{12}$$

which is also unbiased and asymptotic normal with convergence rate $\sqrt{N}$ as $N \to \infty$.

B MORE ON MONTE CARLO LIKELIHOOD FUNCTION

We mentioned in section 2 that in the smoothed likelihood approximation in equation (6), the importance sampling estimator is under the true distribution $p_{\theta_{true}}(x_{0:T}|y_{0:T})$ and the proposal distribution $p_{\theta_{last}}(x_{0:T}|y_{0:T})$. Where θ_{last} is the last parameter to generate the particles. We have made this claim, and we would like to briefly justify it using the toy example below.

Example To illustrate the levels of accuracy of the $\hat{l}_{\theta_0}(\theta)$ approximation, consider estimating the log-likelihood function of a first-order auto-regressive model (AR(1)). Following the notation in (9) and assume that $\sigma_x, sigma_y$ are known, so the parameter of interest $\theta \triangleq \phi$. The true parameter value is $\theta_{true} = 0.7$, $N = 1000, T = 5, \mu_0 = 0, \Sigma_0 = 10, V_t \sim N(0,1), W_t \sim N(0, 0.5^2)$. The usual random seed $s = 1$ is used. In this case, we could use the Kalman filter to obtain the true likelihood curve $l(\theta)$, and we also use independent SMC samples to pointwise approximate the likelihood values, which means numerous sets of particles are needed. We then use three different simulation parameters θ_0, each generating a set of particles, and use the likelihood ratio approximation to produce a smoothed likelihood curve estimate $\tilde{l}_{\theta_0}(\theta)$ for $l(\theta)$. The results are shown below.

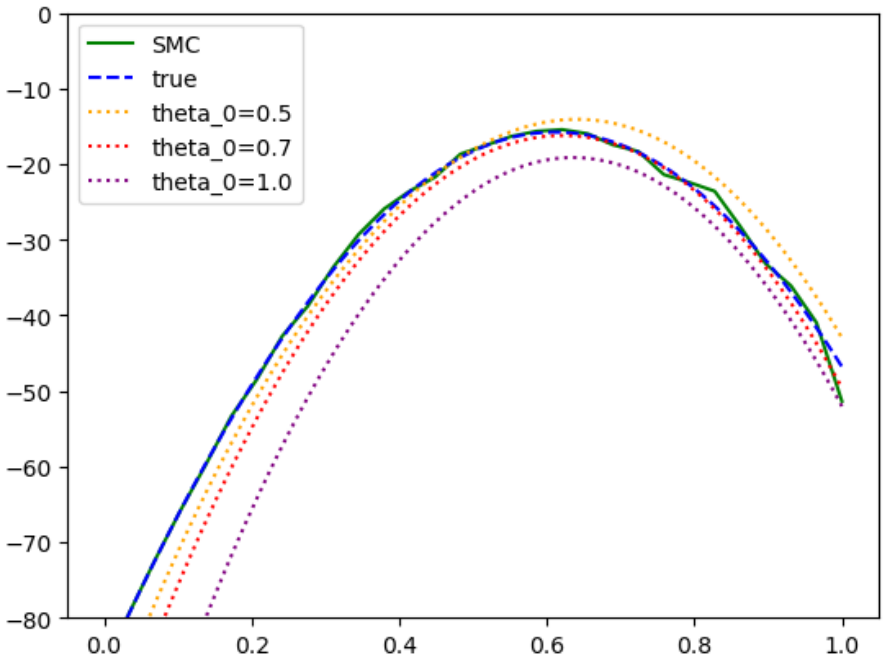

Figure 3: Likelihood approximation of an AR(1) model using independent SMC estimate and smoothed likelihood ratio estimates $\hat{l}_{\theta_0}(\theta)$ under different θ_0

We can see that the SMC approximation is close to the true likelihood curve, but several sets of particles are required and the approximation is not continuous due to the Monte Carlo noise. When $\theta_0 = 0.7 = \theta_{true}$, i.e. the true parameter value is used for the simulation, $\hat{l}_{\theta_{true}}(\theta)$ gives a very nice approximation to $l(\theta)$ using only one set of particles. If θ_0 is far from the true parameter value, i.e. for $\theta_0 = 0.5, 1$ both methods have good approximation accuracy for θ around the corresponding θ_0, but $\hat{l}_{\theta_0}$ becomes increasingly biased as θ is far from θ_0.

C ASSUMPTIONS and PROOF of THEOREM 1

This section provides proof for Theorem 1 in section 3 of the main text. We follow the framework of Douc and Moulines (2008) but the information here should be sufficient for justifying our theories.

Definition 1 Let $L^1(\mathcal{X}, \mu) \triangleq \{f : \mathcal{X} \mapsto \mathbb{R} | \int |f(x)|\mu(x)dx\} < \infty$, The set of weighted samples $\{x_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N$ is said to be consistent for the probability measure μ and the set $\mathcal{C} \subseteq L^1(\mathcal{X}, \mu)$ if for any $f \in \mathcal{C}$, as $N \to \infty$:

$$\Omega_t^{-1} \Sigma_{i=1}^N w_t^{(i)} f(x_{0:t}^{(i)}) \xrightarrow{\text{P}} E_\mu(f(x_{0:t})), \tag{13}$$

$$\Omega_t^{-1} \max_{1 \le i \le N} w_t^{(i)} \xrightarrow{\text{P}} 0, \text{where } \Omega_t \triangleq \Sigma_{i=1}^N w_t^{(i)}. \tag{14}$$

Definition 2 The set of weighted samples $\{x_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N$ is said to be asymptotically normal for $\{\mu, \gamma, A, W, \sigma\}$ if:

$$\sqrt{N}\Omega_t^{-1} \Sigma_{i=1}^N w_t^{(i)} (f(x_{0:t}^{(i)}) - E_\mu(f(x_{0:t})) \xrightarrow{\text{D}} N(0, \sigma^2(f)), \forall f \in A, \tag{15}$$

$$N\Omega_t^{-2}\Sigma_{i=1}^N (w_t^{(i)})^2 f(x_{0:t}^{(i)}) \xrightarrow{\mathrm{P}} \gamma(f), \forall f \in W, \tag{16}$$

$$\sqrt{N}\Omega_t^{-1} \max_{1\le i\le N} w_t^{(i)} \xrightarrow{\mathrm{P}} 0. \tag{17}$$

Definition 3(Proper Set) We say that the set $\mathcal{C}$ is proper if the following conditions hold:

$$\begin{aligned} &(i) Let\ a, b \in \mathbb{R} and f, g \in \mathcal{C}\ then\ af + bg \in \mathcal{C}.\\ &(ii) Let\ f \in \mathcal{C}\ and\ g\ is\ measurable\ with\ |g| < |f|\ then\ g \in \mathcal{C}.\\ &(iii) Any\ c \in \mathbb{R}\ the\ constant\ function\ f \equiv c \in \mathcal{C}. \end{aligned} \tag{18}$$

We assume that the trace of our parameters satisfies regularity conditions:

Assumption A we assume that the trace of our gradient ascend algorithm lies in a set Θ and it is proper in the sense that, given any $x_{0:t}, \{\theta_i\}_{i=0}^{t+1}$, we have that $\forall 0 \le i, j \le t+1$:

$$\begin{aligned} &\frac{p_{\theta_i}(x_{0:t}|y_{0:t})}{p_{\theta_j}(x_{0:t}|y_{0:t})} \in [l_1(x_{0:t}, y_{0:t}), u_1(x_{0:t}, y_{0:t})],\\ &\frac{p_{\theta_i}(x_{0:t}, y_{0:t})}{p_{\theta_j}(x_{0:t}, y_{0:t})} \in [l_2(x_{0:t}, y_{0:t}), u_2(x_{0:t}, y_{0:t})], \end{aligned} \tag{19}$$

where the bounds $l_1(x_{0:t}, y_{0:t}), u_1(x_{0:t}, y_{0:t}), l_2(x_{0:t}, y_{0:t}), u_2(x_{0:t}, y_{0:t})$ are positive. This is not a strong assumption as when samples are given, it may only fail when $\{\theta_i\}_{i=0}^{t+1}$ contain value(s)θ_i that makes $p_{\theta_i}(x_{0:t}|y_{0:t})$ or $p_{\theta_i}(x_{0:t}, y_{0:t})$ arbitrarily close to 0 or ∞, which is unlikely to occur and will particularly leads to failure of the algorithm by producing *nan* values, as mentioned in section 5.1.1. Henceforth, conditioned on a set of weighted samples and observations, we may drop the $x_{0:t}, y_{0:t}$ from l and u.

Assumption B Under assumption A, denote $\pi_\theta^k \triangleq p_\theta(x_{0:k}|y_{0:k})$. Suppose that A_0 and W_0 are proper sets and we recursively define:

$$\begin{aligned} A_k &\triangleq \{f \in L^2(\mathcal{X}, \pi_\theta^k), E_{q_k}[\frac{\pi_\theta^k}{q_k \pi_\theta^{k-1}} f\ |x_{0:k-1}] \in A_{k-1}, E_{q_k}[(\frac{\pi_\theta^k}{q_k \pi_\theta^{k-1}} f)^2\ |x_{0:k-1}] \in W_{k-1},\ \forall \theta \in \Theta\},\\ W_k &\triangleq \{f \in L^1(\mathcal{X}, \pi_\theta^k), E_{q_k}[(\frac{\pi_\theta^k}{q_k \pi_\theta^{k-1}} f)^2\ |x_{0:k-1}] \in W_{k-1},\ \forall \theta \in \Theta\}; \qquad k \ge 1. \end{aligned} \tag{20}$$

We also suppose that the initialisation $\{x_0^{(i)}\}_{i=0}^N$ are consistent for $\{p_{\theta_0}(x_0|y_0), L^1(\mathcal{X}, p_{\theta_0}(x_0|y_0)\}$ and asymptotically normal for $\{p_{\theta_0}(x_0|y_0), p_{\theta_0}(x_0|y_0), A_0, W_0, \sqrt{Var_{q_0}(f)}\}$.

Lemma 1. *Assume that the weighted samples $\{x_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N$ is consistent for $\{p_{\theta_t}(x_{0:t}|y_{0:t}), \mathcal{C}\}$, and denote $a_\theta(\theta_{t+1}, x_{0:t}, y_{0:t})$ by a_t^{t+1}, then after the retargeting step $\tilde{w}_t^{(i)} \triangleq a_t^{t+1} w_t^{(i)}$, the new weighted samples $\{x_{0:t}^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^N$ is consistent for $\{p_{\theta_{t+1}}(x_{0:t}|y_{0:t}), \mathcal{C}\}$.*

Lemma 2. *Assume that the weighted samples $\{x_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N$ is asymptotically normal for $\{p_{\theta_t}(x_{0:t}|y_{0:t}), \gamma, A_t, W_t, \sigma\}$, then after the retargeting step $\tilde{w}_t^{(i)} \triangleq a_t^{t+1} w_t^{(i)}$, the new weighted samples $\{x_{0:t}^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^N$ is asymptotically normal for $\{p_{\theta_{t+1}}(x_{0:t}|y_{0:t}), \tilde{\gamma}, A_t, W_t, \tilde{\sigma}\}$. And $\tilde{\gamma}(f) = (\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})})^2 \gamma((a_t^{t+1})^2 f), \tilde{\sigma}^2(f) = (\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})})^2 \sigma^2(a_t^{t+1} f)$.*

Combining lemma 1,2 with the results from Chopin (2004), we propose the following theorem for the SemiGA-PIS algorithm with varying parameters.

Theorem 2. *Assume that we are in the context of Algorithm 3 in which the multinomial resampling is employed at every step, while renewing does not occur until time t. Denote $p_{\theta_k}(y_{0:t})$ by $p_k(y_{0:t})$, $p_{\theta_k}(\cdot|y_{0:t})$ by π_k^t, the proposing distribution $q(x_t|\cdots)$ by q_t. For test function f and the initialisation of particles satisfying assumption B, we let $\tilde{V}_0(f) \triangleq Var_{q_0} f$ and recursively define:*

$$\begin{aligned} \tilde{V}_t(f) &= \hat{V}_{t-1}(\mathbb{E}_{q_t}(f)) + \mathbb{E}_{\pi_t^{t-1}}(Var_{q_t}(f)),\ \ t > 0,\\ V_t(f) &= \tilde{V}_t(v_t \cdot (f - \mathbb{E}_{\pi_t^t}(f))),\ \ t \ge 0,\\ \bar{V}_t(f) &= (\frac{p^t(y_{0:t})}{p_{t+1}(y_{0:t})})^2 V_t(a_t^{t+1} f),\ \ t \ge 0,\\ \hat{V}_t(f) &= \bar{V}_t(f) + Var_{\pi_{t+1}^t}(f),\ \ t \ge 0. \end{aligned} \tag{21}$$

Then we have that:

$$\begin{aligned}
&\sqrt{N}(N^{-1}\Sigma_{i=1}^{N} f(x_{0:t}^{(i)}) - \mathbb{E}(\pi_t^{t-1} q_t(f))) \xrightarrow{\mathrm{D}} N(0, \tilde{V}_t(f)),\\
&\sqrt{N}(\frac{\Sigma_{i=1}^{N} w_t(x_{0:t}^{(i)}) f(x_{0:t}^{(i)})}{\Sigma_{i=1}^{N} w_t(x_{0:t}^{(i)})} - \mathbb{E}(\pi_t^{t}(f))) \xrightarrow{\mathrm{D}} N(0, V_t(f)),\\
&\sqrt{N}(\frac{\Sigma_{i=1}^{N} \bar{w}_t(x_{0:t}^{(i)}) f(x_{0:t}^{(i)})}{\Sigma_{i=1}^{N} \bar{w}_t(x_{0:t}^{(i)})} - \mathbb{E}(\pi_{t+1}^{t}(f))) \xrightarrow{\mathrm{D}} N(0, \bar{V}_t(f)),\\
&\sqrt{N}(N^{-1}\Sigma_{i=1}^{N} f(\hat{x}_{0:t}^{(i)}) - \mathbb{E}(\pi_{t+1}^{t}(f))) \xrightarrow{\mathrm{D}} N(0, \hat{V}_t(f)).
\end{aligned} \tag{22}$$

Particularly, we specify:

$$\begin{aligned}
V_t(f) = &\int \frac{(\pi_t^t(x_0))^2}{q(x_0)} (\int f(x_{0:t}) \pi_t^t(x_{1:t}|x_0) dx_{1:t} - \mathbb{E}_{\pi_t^t}(f(x_{0:t})))^2 dx_0\\
&+ \Sigma_{k=1}^{t} \int \frac{(\pi_t^t(x_{0:k}))^2}{q(x_k|x_{0:k-1}) \pi_k^{k-1}(x_{0:k-1})} (\int f(x_{0:t}) \pi_t^t(x_{k+1:t}|x_{0:k}) dx_{k+1:t} - \mathbb{E}_{\pi_t^t}(f(x_{0:t})))^2 dx_{0:k},
\end{aligned}$$

where we let $\int f(x_{0:t}) \pi_t^t(x_{t+1:t}|x_{0:t}) dx_{t+1:t} \triangleq f(x_{0:t})$. The expression is also valid for $t = 0$ if we just keep the first term and ignore the second term $\Sigma_{k=1}^{0} \cdots$. The expressions for $\hat{V}_t(f), \bar{V}_t(f), \tilde{V}_t(f)$ could thus be easily calculated from $V_t(f)$.

The intuition behind this is that combining lemma 1,2 with the result from Chopin (2004) gives the four corresponding variances of the theorem. Then one may simply notice that, although the updated new θ_{t+1} and thus the retargeting step is based on particles at time t, it does not influence the particles between reweighting and retargeting. Thus one may simply combine $\tilde{w}_{t+1} = a_t^{t+1} w_{t+1} = a_t^{t+1} u_{t+1} \tilde{w}_t$ and realise $a_t^{t+1} u_{t+1} = \frac{\pi_{t+1}^t}{q_t \pi_t^{t-1}}$, the result then follows as a strict generalisation of Doucet and Johansen (2009). An explicit proof using induction was presented after the proofs of Lemma 1 & 2.

Proof of Lemma 1. We will drop notations $x_{0:t}, y_{0:t}$ for simplicity. By assumption (19) we have that $|a_\theta(\theta_{t+1}) f| \leq |u_2 f|$ and $|u_2 f| \in \mathcal{C}$ by linearity(taking c≡0 and then use (i)). Thus we have $af \in \mathcal{C}$. Then by applying consistency condition to function af we get:

$$\begin{aligned}
&\Omega_t^{-1} \Sigma_{i=1}^{N} \tilde{w}_t^{(i)} f(x_{0:t}^{(i)})\\
=&\Omega_t^{-1} \Sigma_{i=1}^{N} w_t^{(i)} a_\theta(\theta_{t+1}, x_{0:t}^{(i)}, y_{0:t}) f(x_{0:t}^{(i)})\\
\xrightarrow{\mathrm{P}} &\int p_{\theta_t}(x_{0:t}|y_{0:t}) \frac{p_{\theta_{t+1}}(x_{0:t}, y_{0:t})}{p_{\theta_t}(x_{0:t}, y_{0:t})} f \, dx_{0:t}\\
=&\frac{p_{\theta_{t+1}}(y_{0:t})}{p_{\theta_t}(y_{0:t})} \int p_{\theta_{t+1}}(x_{0:t}|y_{0:t}) f \, dx_{0:t}.
\end{aligned} \tag{23}$$

Particularly, taking $f \equiv 1$ we get $\Omega_t^{-1} \Sigma_{i=1}^{N} \tilde{w}_t^{(i)} = \frac{\tilde{\Omega}_t}{\Omega_t} \xrightarrow{\mathrm{P}} \frac{p_{\theta_{t+1}}(y_{0:t})}{p_{\theta_t}(y_{0:t})}$. Combining with (23) we get $\tilde{\Omega}_t^{-1} \Sigma_{i=1}^{N} \tilde{w}_t^{(i)} f(x_{0:t}^{(i)}) \xrightarrow{\mathrm{P}} \int p_{\theta_{t+1}}(x_{0:t}|y_{0:t}) f \, dx_{0:t}$, as required.

Now we simply notice that $\tilde{\Omega}^{-1} \max_{1 \leq i \leq N} \tilde{w}_t^{(i)} \xrightarrow{\mathrm{P}} \frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})} \Omega^{-1} \max_{1 \leq i \leq N} \tilde{w}_t^{(i)}$ and the fact that $\Omega^{-1} \max_{1 \leq i \leq N} \tilde{w}_t^{(i)} \leq u_2 \Omega^{-1} \max_{1 \leq i \leq N} w_t^{(i)} \xrightarrow{\mathrm{P}} 0$, so we have $\tilde{\Omega}^{-1} \max_{1 \leq i \leq N} \tilde{w}_t^{(i)} \xrightarrow{\mathrm{P}} 0$.

□

Proof of Lemma 2. We denote $p_\theta(f) \triangleq \int p_\theta(x_{0:t}|y_{0:t}) f(x_{0:t}) dx_{0:t}$. We have that:

$$\begin{aligned}
&\sqrt{N} \Omega_t^{-1} \Sigma_{i=1}^{N} w_t^{(i)} \{f(x_{0:t}^{(i)}) - p_{\theta_t}(f)\} \xrightarrow{\mathrm{D}} N(0, \sigma^2(f)),\\
&N \Omega_t^{-2} \Sigma_{i=1}^{N} (w_t^{(i)})^2 f(x_{0:t}^{(i)}) \xrightarrow{\mathrm{P}} \gamma(f),\\
&\sqrt{N} \Omega^{-1} \max_{1 \leq i \leq N} w_t^{(i)} \xrightarrow{\mathrm{P}} 0.
\end{aligned}$$

And we first prove that $\sqrt{N}\tilde{\Omega}_t^{-1}\Sigma_{i=1}^N \tilde{w}_t^{(i)}\{f(x_{0:t}^{(i)}) - p_{\theta_{t+1}}(f)\} \xrightarrow{\mathrm{D}} N(0, \tilde{\sigma}^2(f))$. Notice that $p_{\theta_t}(a_{\theta_t}(\theta_{t+1})f) = \int p_{\theta_t} \frac{p_{\theta_{t+1}}(x_{0:t}, y_{0:t})}{p_{\theta_t}(x_{0:t}, y_{0:t})} f \, dx_{0:t} = \frac{p_{\theta_{t+1}}(y_{0:t})}{p_{\theta_t}(y_{0:t})} p_{\theta_{t+1}}(f)$. Combinining this with the previous fact that $\Omega_t^{-1}\Sigma_{i=1}^N \tilde{w}_t^{(i)} = \frac{\tilde{\Omega}_t}{\Omega_t} \xrightarrow{\mathrm{P}} \frac{p_{\theta_{t+1}}(y_{0:t})}{p_{\theta_t}(y_{0:t})}$ we then get:

$$
\begin{aligned}
&\sqrt{N}\tilde{\Omega}_t^{-1}\Sigma_{i=1}^N \tilde{w}_t^{(i)}\{f(x_{0:t}^{(i)}) - p_{\theta_{t+1}}(f)\} \\
=&\sqrt{N}\{\tilde{\Omega}_t^{-1}\Sigma_{i=1}^N w_t^{(i)} a_{\theta_t}(\theta_{t+1}) f(x_{0:t}^{(i)})\} - \sqrt{N} p_{\theta_{t+1}}(f) \\
=&\sqrt{N}\{\tilde{\Omega}_t^{-1}\Sigma_{i=1}^N w_t^{(i)} a_{\theta_t}(\theta_{t+1}) f(x_{0:t}^{(i)})\} - \sqrt{N}\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})} p_{\theta_t}(a_{\theta_t}(\theta_{t+1})f). \quad (*)
\end{aligned}
$$

By noticing that $\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})} p_{\theta_t}(a_{\theta_t}(\theta_{t+1})f) \xrightarrow{\mathrm{P}} \tilde{\Omega}_t^{-1}\Sigma_{i=1}^N w_t^{(i)} p_{\theta_t}(a_{\theta_t}(\theta_{t+1})f)$, we have:

$$
\begin{aligned}
(*) &\xrightarrow{\mathrm{P}} \sqrt{N}\tilde{\Omega}_t^{-1}\Sigma_{i=1}^N w_t^{(i)}\{a_{\theta_t}(\theta_{t+1}) f(x_{0:t}^{(i)}) - p_{\theta_t}(a_{\theta_t}(\theta_{t+1})f)\} \\
&\xrightarrow{\mathrm{P}} \sqrt{N}\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})}\Omega_t^{-1}\Sigma_{i=1}^N w_t^{(i)}\{a_{\theta_t}(\theta_{t+1}) f(x_{0:t}^{(i)}) - p_{\theta_t}(a_{\theta_t}(\theta_{t+1})f)\} \\
&\xrightarrow{\mathrm{D}} N(0, (\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})})^2 \sigma^2(af)),
\end{aligned}
$$

which proves the first statement. Here we used the fact that $f \in A_t$ implies $af \in A_t$ (similarly for W_t), which is straightforward by combining Assumption A with the construction of $\{A_k, W_k\}_{k=0}^t$.

Then simply notice that $|a^2 f| \leq (u_2)^2 |f|$ and thus

$$
\begin{aligned}
&N\tilde{\Omega}_t^{-2}\Sigma_{i=1}^N (\tilde{w}_t^{(i)})^2 f(x_{0:t}^{(i)}) \\
\xrightarrow{\mathrm{P}}&(\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})})^2 N\Omega_t^{-2}\Sigma_{i=1}^N (w_t^{(i)})^2 a_{\theta_t}(\theta_{t+1})^2 f(x_{0:t}^{(i)}) \\
\xrightarrow{\mathrm{P}}&(\frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})})^2 \gamma(a^2 f).
\end{aligned}
$$

Finally, notice that:

$$
\sqrt{N}\tilde{\Omega}_t^{\ -1} \max_{1\leq i\leq N} \tilde{w}_t^{(i)} \leq u_2 \sqrt{N}\tilde{\Omega}_t^{\ -1} \max_{1\leq i\leq N} w_t^{(i)} \xrightarrow{\mathrm{P}} u_2 \frac{p_{\theta_t}(y_{0:t})}{p_{\theta_{t+1}}(y_{0:t})} \sqrt{N}\Omega_t^{-1} \max_{1\leq i\leq N} w_t^{(i)} \xrightarrow{\mathrm{P}} 0,
$$

which completes the proof. □

Proof of Theorem 1. Recall that the semiGA-PIS algorithm proceeds as propagation, reweighting, retargeting and resampling. The result from Chopin (2004) gives the influence of propagation, reweighting and resampling, while lemma 2 specifies the expression of variance $\bar{V}_t(f)(\tilde{\sigma}^2(f)$in the setting of theorem 2) after retargeting, combining these results gives the four corresponding variances of the theorem. We now prove the expression for $V_t(f)$ by induction.

For $t = 0$ the statement obviously holds, suppose it holds for t, our calculation proceeds by following the inductive definitions. Notice that f(and also its domain) changes when the subscript time index increases, but we retain the notation f to avoid overloading notations.

$$
\begin{aligned}
&\bar{V}_t(f) = (\frac{(p^t(y_{0:t}))^2}{p_{t+1}(y_{0:t})})^2 V_t(a_t^{t+1} f) \\
&= \int \frac{(\pi_t^t(x_0))^2}{q(x_0)} (\int \frac{\pi_{t+1}^t(x_{0:t})}{\pi_t^t(x_{0:t})} f(x_{0:t}) \pi_t^t(x_{1:t}|x_0) dx_{1:t} - \mathbb{E}_{\pi_{t+1}^t}(f(x_{0:t})))^2 dx_0 \\
&+ \Sigma_{k=1}^t \int \frac{(\pi_t^t(x_{0:k}))^2}{\pi_k^{k-1}(x_{0:k}) q(x_k|x_{0:k-1})} [\int \frac{\pi_{t+1}^t(x_{0:t})}{\pi_t^t(x_{0:k})} f(x_{0:t}) dx_{k+1:t} - \int \pi_{t+1}^t f(x_{0:t}) dx_{0:t}]^2 dx_{0:k}.
\end{aligned}
$$

Then

$$\tilde{V}_{t+1}(f) = \bar{V}_t(\int q(x_{t+1}|x_{0:t})f(x_{0:t+1})dx_{t+1}) + \mathrm{Var}_{\pi^t_{t+1}}(f) + \mathbb{E}_{\pi^t_{t+1}}(\mathrm{Var}_{q_{t+1}}(f))$$
$$= \bar{V}_t(\int q(x_{t+1}|x_{0:t})f(x_{0:t+1})dx_{t+1}) + \mathrm{Var}_{\pi^t_{t+1}q_{t+1}}(f)\ (law\ of\ total\ variation)$$
$$= \int \frac{(\pi^t_t(x_0))^2}{q(x_0)}(\int \frac{\pi^t_{t+1}(x_{0:t})}{\pi^t_t(x_{0:t})}q(x_{t+1}|x_{0:t})f(x_{0:t+1})\pi^t_t(x_{1:t}|x_0)dx_{1:t+1} - \int \pi^t_{t+1}q(x_{t+1}|x_{0:t})f(x_{0:t+1})dx_{0:t+1})^2dx_0$$
$$+ \Sigma^t_{k=1}\int \frac{(\pi^t_t(x_{0:k}))^2}{\pi^{k-1}_k(x_{0:k-1})q(x_k|x_{0:k-1})}[\int \frac{\pi^t_{t+1}(x_{0:t})}{\pi^t_t(x_{0:k})}q(x_{t+1}|x_{0:t})f(x_{0:t+1})dx_{k+1:t+1}$$
$$- \int \pi^t_{t+1}(x_{0:t})q(x_{t+1}|x_{0:t})f(x_{0:t+1})dx_{0:t+1}]^2dx_{0:k}$$
$$+ \int \pi^t_{t+1}(x_{0:t})q(x_{t+1}|x_{0:t})[f(x_{0:t+1}) - \int \pi^t_{t+1}(x_{0:t})q(x_{t+1}|x_{0:t})f(x_{0:t+1})dx_{0:t+1}]^2.$$

Therefore

$$V_{t+1}(f) = \tilde{V}_{t+1}(\frac{\pi^{t+1}_{t+1}(x_{0:t+1})}{\pi^t_{t+1}(x_{0:t})q(x_{t+1}|x_{0:t})}[f(x_{0:t+1}) - \int \pi^{t+1}_{t+1}(x_{0:t+1})f(x_{0:t+1})dx_{0:t+1}])$$
$$\overset{(**)}{=} \int \frac{(\pi^{t+1}_{t+1}(x_0))^2}{q(x_0)}(\int \frac{\pi^{t+1}_{t+1}(x_{0:t+1})}{\pi^{t+1}_{t+1}(x_0)}f(x_{0:t+1})dx_{1:t+1} - \int \pi^{t+1}_{t+1}f(x_{0:t+1})dx_{0:t+1})^2dx_0$$
$$+ \Sigma^t_{k=1}\int \frac{\pi^{t+1}_{t+1}(x_{0:k})}{\pi^{k-1}_k(x_{0:k-1})q(x_k|x_{0:k-1})}[\int \pi_{t+1}(x_{k+1:t+1}|x_{0:k})f(x_{0:t+1})dx_{k+1:t+1}$$
$$- \int \pi^{t+1}_{t+1}(x_{0:t+1})f(x_{0:t+1})dx_{0:t+1}]^2dx_{0:k}$$
$$+ \int \frac{(\pi^{t+1}_{t+1}(x_{0:t+1}))^2}{\pi^t_{t+1}(x_{0:t})q(x_{t+1}|x_{0:t})}(f(x_{0:t+1}) - \int \pi^{t+1}_{t+1}(x_{0:t+1})f(x_{0:t+1})dx_{0:t+1})^2dx_{0:t+1}$$
$$= \int \frac{(\pi^{t+1}_{t+1}(x_0))^2}{q(x_0)}(\int f(x_{0:t+1})\pi^{t+1}_{t+1}(x_{1:t+1}|x_0)dx_{1:t+1} - \mathbb{E}_{\pi^{t+1}_{t+1}}(f(x_{0:t+1})))^2dx_0$$
$$+ \Sigma^{t+1}_{k=1}\int \frac{\pi^{t+1}_{t+1}(x_{0:k})}{\pi^{k-1}_k(x_{0:k-1})q(x_k|x_{0:k-1})}[\int \pi_{t+1}(x_{k+1:t+1}|x_{0:k})f(x_{0:t+1})dx_{k+1:t+1}$$
$$- \int \pi^{t+1}_{t+1}(x_{0:t+1})f(x_{0:t+1})dx_{0:t+1}]^2dx_{0:k},$$

where$(**)$ includes simplification$\int \pi^{t+1}_{t+1}[f(x_{0:t+1}) - \int \pi^{t+1}_{t+1}f(x_{0:t+1}dx_{0:t+1})]dx_{0:t+1} = 0$. □

D MORE DETAILS AND RESULTS ON THE NUMERICAL EXPERIMENTS

All numerical experiments were implemented in Python 3.0. For the AR(1) with noise model, we use the optimal proposal for SMC. Standard bootstrap methods are applied for all algorithms in the rest of the models, and for the semiGA-PIS algorithm, we choose $K = 1$ for ESS_k. The compute worker is CPU: Processor 12th Gen Intel(R) Core(TM) i7-12700 2.10 GHz, with installed RAM 32GB, under the 64-bit operating system, x64-based processor. We use the bootstrap SMC, i.e. vanilla SMC in Algorithm 1 with the proposal $q_\theta(x_{t+1}|x_{0:t}) = f_\theta(x_{t+1}|x_t)$. The performance of the algorithms is measured by the Root Mean Squared Error(RMSE):

$$RMSE(\theta; \theta^{1:S}_t) \triangleq \sqrt{S^{-1}\Sigma^S_{s=1}(\theta^s_t - \theta)^2}, \tag{24}$$

where superscript s indicates the s^{th} among the total of S experiments and subscript t is the index of the iteration, θ refers to the value that RMSE is calculated for, usually set to be θ_{true} or $\hat{\theta}_{MLE}$.

Python sometimes returns *nan* value due to the failure of the algorithm, which may come from division by very small values close to zero, or improper command asking the density of $N(a, b)$ with $b < 0$, etc. We point this out because when the *nan* value is returned we could not use the formula for RMSE anymore. Specifically, for the

illustration of robustness in section 5.1(where the parameter of interestθ corresponds to ϕ), we address this by only calculating RMSE when $1.3 > \theta_t^s > 0.6$, and otherwise adding a mild penalty of 0.35^2 on the term.

The following are discussions for the choices of learning rates in all models based on the guidelines from Spall (1998). Specifically γ_t is in the form of $\frac{c_1}{(A+t)^\alpha}$. The finite difference method for Naive-SGA uses SPSA(Spall, 1998) where the perturbation vector is $\tau_t\Delta_t$ where$\tau_t \triangleq \frac{c_2}{(A+t)^\beta}$ and Δ_t is a Rademacher random vector(i.e. each component taking 1 or -1 with probability 0.5) having the same dimensions as the parameter of interest θ. We choose the pair $(\alpha, \beta) = (1, 1/6)$, which is an optimal choice based on a large number of experiments(Spall, 1998). For model(9) we choose $(c_1, c_2, A) = (0.0001, 0.05, 100)$. For model(11) we choose $(c_1, c_2, A) = (0.00005, 0.05, 100)$. For the PAR(1) model we choose $(c_1, c_2, A) = (0.2, 0.02, 2000)$, we choose large A here because the data size $T = 168$ is small compared to the previous experiments with $T = 10000$ and we want to reduce the effect of wrong updates for the online algorithm based on early iterations with possibly outlier observations. The joint choice of c_1, A in all models are based on the magnitude of the gradient and the scale of the parameter. Particularly, we would like to address the fact that the learning rates are reasonable and fair in the sense that their scales are sufficient for multiple algorithms to converge, given that the gradient estimations of the algorithms are correct. More details are contained in the code.

D.1 Tuning the ESS thresholds r and r_1

Essentially this is done by finding reasonable values of ESS in test runs of the algorithms which use smaller data size. Specifically, for the offline estimation, multiple parameter values are selected randomly in a region of reasonable parameter values. By moving each of them with one Newton-Raphson step size which can easily be estimated with the Fisher and Louise identities, a set of ESS values are calculated. The average of these values represents the ESS when the step size is optimal and is chosen as r.

For online estimation, in the test run θ_t is moved by a fixed value of the distance d, instead of the scaled conditional gradient in Algorithm 3, at every t and the value of ESS is recorded. Also in the test run particles are renewed at a relatively low frequency. By doing this we observe the range of ESS which includes the values when the move ranges from too small to too far, and the medium of the recorded ESS is used as r_1. Several values of d are tested by taking the physical meaning of the parameter into consideration, where values of d only giving small or large values of ESS are excluded.

We have tried different values of r and r_1 for the SV model in Section 4.2 where r_1 is obtained by using different d in the above tuning procedure, and the results are similar which is given below.

Table 4: Experimenting different r, r_1 in section 4.2

Algorithms	ϕ	σ_x	σ_y	Algorithms	ϕ	σ_x	σ_y
P-offline $O(N^2)$	2.403	2.031	0.220	AdaptGA-PIS($r = 0.334$)	0.343	0.892	0.059
P-online $O(N^2)$	1.000	1.000	1.000	AdaptGA-PIS($r = 0.449$)	0.139	0.727	0.045
Naive SGA	4.552	2.512	2.023	SemiGA-PIS($r_1 = 0.6$)	0.235	0.265	0.059
Fisher SGA	3.925	1.673	0.896	SemiGA-PIS($r_1 = 0.487$)	0.235	0.237	0.045
AdaptGA-PIS($r = 0.4$)	0.151	0.802	0.048	SemiGA-PIS($r_1 = 0.692$)	0.241	0.285	0.074

The result shows that the performance of our algorithms is satisfying with r and r_1 tuned by the above procedure, and is reasonably insensitive to the choice of r and r_1.

D.2 The Naive and Fisher SGA Algorithms

We present the structure of the two SGA algorithms.

Algorithm 4 Steepest gradient ascent(SGA) using SMC

Assume that $y_{0:T}$, the number of total iterations I, step number K and properly chose $\{\gamma_n\}_{n=0}^{I}$ are given.
Initialisation Specify the initial parameter θ_0.
For $n = 0$ to I:

1. Run the SMC algorithm to obtain $\{x_{0:T}^{(i)}, w_T^{(i)}\}_{i=1}^{N}$ follow the density $p_{\theta_n}(x_{0:T}|y_{0:T})$.
2. Use finite difference or Fisher's identity to estimate $\nabla \log p_\theta\left(y_{0:t}\right)|_{\theta=\theta_n}$, and proceed gradient ascend $\theta_{n+1} = \theta_n + \gamma_n \nabla_\theta \log p_\theta\left(y_{0:T}\right)|_{\theta=\theta_n}$

End For

The algorithm has a similar structure as our proposed offline algorithm, but simulating new particles at every iteration.

D.3 Observed Datasets in Section 4.1 and 4.2

Here in Figure 4 we present the plots for the observed data used in Section 5. The real dataset in Section 5.3 is not presented here.

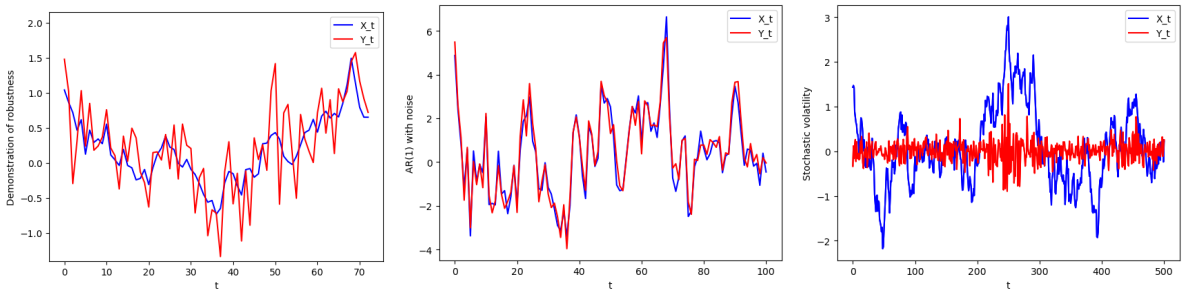

Figure 4: The observed dataset simulated from model (10) (left), model (9)(middle) and model (11)(right)

D.4 More Results for Section 4.1

For model (10), the RMSE plot comparing Algorithm 1 and Algorithm 3 and the one comparing other algorithms are reported in Figure 5 and 6, respectively. These results show that the semi-online algorithm significantly outperforms the ad-hoc online algorithm and the offline algorithms when part of the observed sequence is highly informative about the unknown parameter.

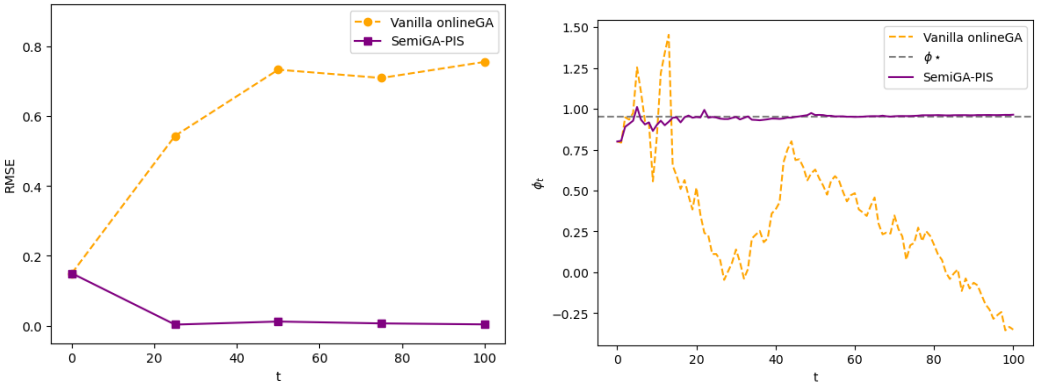

Figure 5: Comparison of Algorithm 1(vanilla onlineGA) and Algorithm 3(semiGA-PIS) model (10) with $\phi_\star = 0.95$ and the initial parameter $\phi_0 = 0.8$. $\sigma_x = 0.5$ and $\sigma_y = 0.5$ are known. The RMSE plot(left) is obtained over 20 replications, and the trajectory plot(right) is from one of the replications.

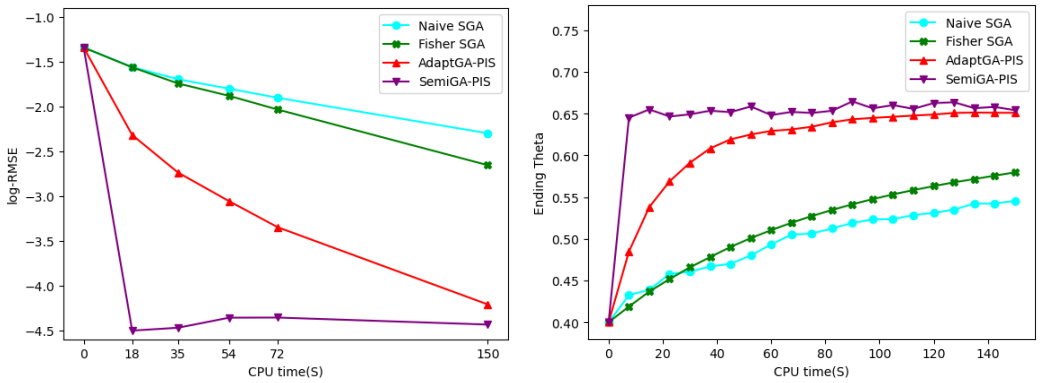

Figure 6: Log-RMSE plot obtained with 20 replications(left) and the trajectory plot from one replication (right) under model (10). The initial value $\phi_0 = 0.4$. The true parameter $\phi^\star = 0.7$, and $\sigma_x = 0.5$ and $\sigma_y = 0.5$ are known.

D.5 More Results for Section 5.2

Consider the following stochastic volatility model where a degenerating trend term is added to the mean volatility:

$$X_{t+1} \sim \phi X_t + \sigma_x \eta_t, \ t = 0, \ldots, T-1,$$
$$Y_t \sim \sigma_y e^{(8\phi^t + X_t)/2} \xi_t, \ t = 0, \ldots, T,$$

where X_0 has mean 0 and η_t and ξ_t follow $N(0,1)$. The parameter to estimate is $\theta = (\phi, \sigma_x, \sigma_y)$. The MLE of θ is obtained by running all algorithm with large values of N and iteration numbers until sure convergence and the converged value is agreed by the majority of algorithms. The RMSE plots for all three parameters are shown in Figure 7. Again the semi-online algorithm significantly outperforms all other algorithms for a similar reason as before. The naive SGA performs the worst, with the Fisher SGA only slightly improving over it.

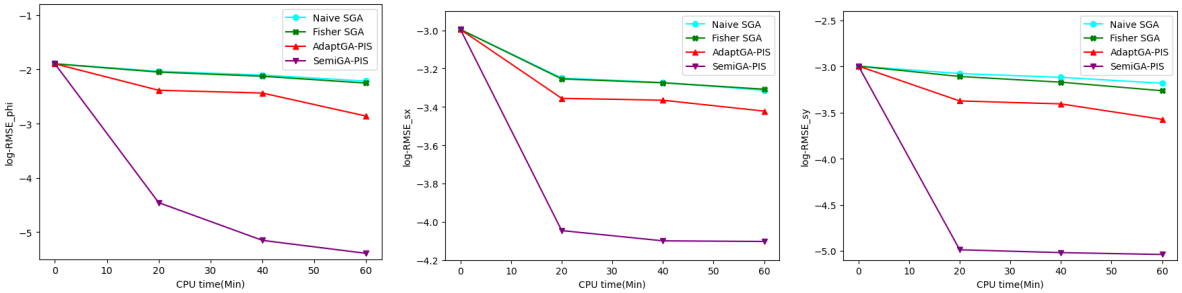

Figure 7: Log-RMSE plots from 20 replications for ϕ(left), σ_x(middle) and σ_y(right) for all algorithms under the SV model. The true parameter $\theta^\star = (0.95, 0.25, 0.2)$ and $T = 500$. For all algorithms $N = 2000$, the initial parameters $\theta_0 = (0.8, 0.2, 0.15)$, $r = 0.6$ for Algorithm 2 and $r_1 = 0.8$ for Algorithm 3.