Kernel Sequential Monte Carlo

Ingmar Schuster[1], Heiko Strathmann[2], Brooks Paige[3], and Dino Sejdinovic[4](✉)

[1] FU Berlin, Berlin, Germany
ingmar.schuster@fu-berlin.de
[2] Gatsby Unit, University College London, London, UK
heiko.strathmann@gmail.com
[3] Alan Turing Institute and University of Cambridge, Cambridge, UK
bpaige@turing.ac.uk
[4] University of Oxford and Alan Turing Institute, Oxford, UK
dino.sejdinovic@stats.ox.ac.uk

Abstract. We propose kernel sequential Monte Carlo (KSMC), a framework for sampling from static target densities. KSMC is a family of sequential Monte Carlo algorithms that are based on building emulator models of the current particle system in a reproducing kernel Hilbert space. We here focus on modelling nonlinear covariance structure and gradients of the target. The emulator's geometry is adaptively updated and subsequently used to inform local proposals. Unlike in adaptive Markov chain Monte Carlo, continuous adaptation does not compromise convergence of the sampler. KSMC combines the strengths of sequental Monte Carlo and kernel methods: superior performance for multimodal targets and the ability to estimate model evidence as compared to Markov chain Monte Carlo, and the emulator's ability to represent targets that exhibit high degrees of nonlinearity. As KSMC does not require access to target gradients, it is particularly applicable on targets whose gradients are unknown or prohibitively expensive. We describe necessary tuning details and demonstrate the benefits of the the proposed methodology on a series of challenging synthetic and real-world examples.

1 Introduction

Monte Carlo methods for estimating integrals have become one of the main inference tools of statistics and machine learning over the last thirty years. They are used to numerically approximate intractable integrals with respect to Bayesian posterior distributions. Importantly, they also provide means to quantify uncertainty in the form of variance estimates, credible intervals and regions of high posterior density. The most widely adopted Monte Carlo method is Markov Chain Monte Carlo (MCMC), which constructs a Markov chain that admits the desired target as its stationary distribution; MCMC generates approximate samples from the target when the chain is run sufficiently long. Poorly tuned

I. Schuster and H. Strathmann—Equal contribution.

M. Ceci et al. (Eds.): ECML PKDD 2017, Part I, LNAI 10534, pp. 390–409, 2017.
https://doi.org/10.1007/978-3-319-71249-9_24

MCMC samplers may need to run 'burn in' for a very long time before reaching its equilibrium distribution, and successive samples may be highly correlated.

In contrast, sequential Monte Carlo (SMC) methods are based on iterative importance sampling, and have traditionally been applied to inference in filtering problems with a sequence of time-varying target distributions [9], e.g. in state-space models, where each intermediate distribution is typically defined on a successively larger latent space. In this paper, we focus on static SMC methods, which recently have generated increasing interest as an alternative to MCMC for Bayesian inference on a single target distribution [3,4,6,11]. Static SMC frames inference over a fixed target distribution as a sequential problem by defining an artificial series of incremental targets. This can be done by tempering the target density [6], by including data points sequentially [4], or by targeting the full density at every iteration. The latter is a special case known as population Monte Carlo (PMC, [2]).

Kernel methods have recently been employed to construct efficient adaptive MCMC algorithms: via modelling a Markov chain trajectory in a reproducing kernel Hilbert space (RKHS) and using geometry therein, it is possible to significantly improve mixing on target distributions with nonlinear interactions between components. Covariance in the RKHS can be used to construct an adaptive random walk scheme, kernel adaptive Metropolis Hastings (KAMH), with proposals that are locally aligned with the target density [20]. Gradients of exponential families in the RKHS can be used to construct kernel Hamiltonian Monte Carlo (KHMC), an algorithm that behaves similar to Hamiltonian Monte Carlo (HMC) but without requiring access to gradient information [22]. Both KAMH and KHMC fall back to a random walk in yet unexplored regions, inheriting convergence properties such as geometric ergodicity on log-concave targets (c.f. Proposition 3 in [22]).

In this paper, we develop a framework for *kernel sequential Monte Carlo (KSMC)* for sampling from static models. Similarly to the previous work in adaptive MCMC [20,22], KSMC represents the (weighted) particle system of SMC algorithms in a RKHS. The learned geometry of the corresponding 'emulator' model is used to construct proposal distributions for both MCMC rejuvenation and importance sampling steps inside SMC.

We apply this framework to two existing SMC algorithms, combining the strengths of SMC with those of kernel adaptive MCMC. Firstly, we introduce *kernel adaptive sequential Monte Carlo (KASMC)*, where the global covariance estimate in the adaptive SMC sampler [ASMC, 11] is replaced by a kernel-informed local covariance [20]. Similar to ASMC, KASMC's proposals start as a standard random walk and then smoothly transition to taking locally aligned steps. As a result, sampling efficiency can be significantly improved over ASMC. Secondly, we use an infinite dimensional exponential family model [21] to estimate target gradients as in Strathmann et al. [22]. This results in *kernel gradient importance sampling (KGRIS)*, a gradient-free version of gradient importance sampling (GRIS) [19]. KGRIS is a novel adaptation of kernel gradient estimation ideas for constructing Langevin diffusions, and inherits their sampling efficiency

Algorithm 1. Sequential Monte Carlo for Static Models

Input: Sequence of target densities $\pi_0, \ldots, \pi_T$ (where $\pi_T = \pi$), size of particle system N
Output: sets $\mathbf{X}_1, \ldots, \mathbf{X}_T$ and $\mathbf{W}_1, \ldots, \mathbf{W}_T$ of samples and accompanying weights
Initialise $\mathbf{X}_0$ to N samples from π_0, and $\mathbf{W}_0$ to equal weights $1/N$
for $t = 1$ through $t = T$ **do**
  $\widetilde{\mathbf{W}}_t = \{W_{t-1}^i \pi_t(X_{t-1}^i)/\pi_{t-1}(X_{t-1}^i)\}_{i=1}^N$
  construct $\widetilde{\mathbf{X}}_t$ by re-sampling $(\mathbf{X}_{t-1}, \widetilde{\mathbf{W}}_t)$, resulting in N copies of samples in $\mathbf{X}_{t-1}$
  construct or update proposal q_t
  if using an MH transition kernel **then**
    Set $\mathbf{X}_t$ to
      $\{X_t^i \sim \text{MH kernel with proposal } q_t(\cdot|\widetilde{X}_t^i)\}_{i=1}^N$
    $\mathbf{W}_t = \{1/N\}_{i=1}^N$
  else
    Set $\mathbf{X}_t$ to N samples from
      $q_t^{\text{Mixt}}(\cdot) = \frac{1}{N}\sum_{i=1}^N q_t(\cdot|\widetilde{X}_t^i)$
    $\mathbf{W}_t = \{\pi_t(X_{t,i})/q_t^{\text{Mixt}}(X_{t,i})\}_{i=1}^N$
  end if
end for
return $\mathbf{X}_1, \ldots, \mathbf{X}_T$ and $\mathbf{W}_1, \ldots, \mathbf{W}_T$

compared to random walks. Our contribution includes crucial implementation details, such as Rao-Blackwelisation, stratification, and tuning of the presented algorithms.

Unlike for Langevin diffusions or Hamiltonian dynamics, our framework does not require gradients or higher-order information of the target. Consequently, the KSMC framework is particularly useful in combination with importance sampling frameworks such as SMC^2 [5] and IS^2 [23] for sampling from doubly intractable targets, where gradient information is unavailable.

We finally argue that (adaptive) SMC is a more natural framework for employing RKHS-based representations. Adaptive MCMC samplers require a vanishing adaptation schedule in order to ensure convergence to the correct target [18], creating a difficult to tune exploration-exploitation trade-off with limited principled guidance on selecting such adaptation schedules. In contrast, SMC proposals can continuously be adapted and the choice of an adaptation schedule is thus entirely circumvented. An easy to use Python package implementing the proposed methods is available under an open source licence.[1]

2 Background

Sequential Monte Carlo algorithms [6,8] approximate a target density π by iteratively targeting a sequence of incremental densities $\pi_0, \ldots, \pi_T$, with $\pi_T = \pi$. These incremental densities are typically defined such that the initial density π_0

[1] Source code available at https://github.com/ingmarschuster/kameleon_rks.

is easy to sample from (e.g. the prior in a Bayesian model). Consecutive distributions π_t, π_{t+1} are 'close', in the sense that drawing samples from π_{t+1} given samples from π_t is easier than drawing samples from π_{t+1} directly. At each stage t, we approximate the target density π_t with a set of N samples $\mathbf{X}_t = \{X_t^i\}_{i=1}^N$ with associated importance weights $\mathbf{W}_t = \{W_t^i\}_{i=1}^N$, with

$$\hat{\pi}_t(X) = \sum_{i=1}^{N} W_t^i \delta_{X_t^i}(X) \tag{1}$$

where $\delta_{X_t^i}$ is a Dirac point mass on X_t^i. In contrast to SMC as applied to state space models, in a static SMC setting each target density π_t is defined on the same space $\mathcal{X}$.

We initialise the algorithm by sampling an initial set of N samples $\mathbf{X}_0$ from the initial density q_0, with equal importance weights $1/N$. For each subsequent $t = 1, \ldots, T$, given a particle set $(\mathbf{X}_{t-1}, \mathbf{W}_{t-1})$ approximating π_{t-1}, we construct a new particle set which approximates π_t. This is a three-step process, summarised in Algorithm 1. First, we re-weight each particle relative to the new target density, setting

$$\widetilde{W}_t^i = W_{t-1}^i \frac{\pi_t(X_{t-1}^i)}{\pi_{t-1}(X_{t-1}^i)}.$$

Weighting the points in $\mathbf{X}_{t-1}$ by $\{\widetilde{W}_t^i\}_{i=1}^N$ yields an approximation to π_t in the same manner as in (1) — the new importance weights correct for the change from π_{t-1} to π_t.

Static SMC then applies re-sampling, constructing an equally-weighted set of particles $\widetilde{\mathbf{X}}_t = \{\widetilde{X}_t^i\}_{i=1}^N$ by sampling with replacement from $\mathbf{X}_{t-1}$ with weights proportional to $\widetilde{W}_t^i$, [7]. Together, these samples form an approximation to π_t, where values from $\mathbf{X}_{t-1}$ with high weight under π_t have been duplicated and those with low weight under π_{t-1} have been discarded. This duplication of values, however, can lead to a sample impoverishment problem: many of the re-sampled values $\widetilde{X}_t^i$ may have identical values. This can be avoided by applying a so-called *rejuvenation* step after re-sampling [4], constructing an overall approximation $(\mathbf{X}_t, \mathbf{W}_t)$ to π_t with a diverse set of values of X_t^i.

The rejuvenation step consists of a proposal $q_t(\mathbf{X}_t|\widetilde{\mathbf{X}}_t)$. We here consider two ways of incorporating such a proposal. One traditional option is to use a Markov density q_t as a proposal in a Metropolis-Hastings (MH) kernel which leaves π_t invariant: For each $\widetilde{X}_t^i$ in $\widetilde{\mathbf{X}}_t$, we propose a new value X_t^i from $q_t(X_t^i|\widetilde{X}_t^i)$ and accept it according to a standard MH acceptance ratio targeting π_t. In this case, each importance weight in $\mathbf{W}_t$ will be identically $1/N$.

An alternative is to consider the mixture proposal of all such Markov densities q_t as an importance sampling proposal over π_t, a common approach in PMC. We can define

$$q_t^{\text{Mixt}}(X_t) = \frac{1}{N} \sum_{i=1}^{N} q_t(X_t|\widetilde{X}_t^i),$$

and draw N samples X_t from q_t^{Mixt} to generate $\mathbf{X}_t$. Now we set importance weights in $\mathbf{W}_t$ to $W_t^i = \pi_t(X_t^i)/q_t^{\text{Mixt}}(X_t^i)$ for $i = 1, \ldots, N$.

2.1 Existing SMC Algorithms

In SMC algorithms, we are free in choosing a proposal q_t. In contrast to MCMC, it may be directly informed by the previous samples $\mathbf{X}_{t-1}$ and their weights $\mathbf{W}_{t-1}$. The following two existing SMC algorithms are examples that we will extend to kernel-based alternatives.

Adaptive SMC. The adaptive SMC sampler (ASMC) studied by Fearnhead and Taylor [11] is based on continuously estimating the global covariance Σ_t of π_t, and updating a scaling parameter ν^2. This is done from the re-weighted particle system, which is subsequently moved through a Markov kernel. The proposal distribution used within the MH kernel at point X in Algorithm 1 is $q_t(\cdot|X) = \mathcal{N}(\cdot|X, \nu^2\Sigma_t + \gamma^2 I)$.

Gradient importance sampling. In addition to using the estimated covariance Σ_t of π as in ASMC, gradient importance sampling [GRIS, 19] incorporates a drift term based on the log target gradient. For target gradient $\nabla \log \pi$ and previous sample X, the proposal distribution in Algorithm 1 is $q_t(\cdot) = \mathcal{N}(\cdot|X + D(\nabla \log \pi(X)), \nu^2\Sigma_t)$, for each individual particle X in the current (unweighted) particle set. A typical choice for the drift function is $D(y) = \delta y$ with $0 < \delta < 1$. Rather than incorporating a MH step, the updated values are importance weighted — GRIS is a population Monte Carlo (PMC) algorithm. In numerical experiments, GRIS compares favourably to its closest MCMC relatives like the adaptive MALTA algorithm and adaptive Metropolis [19].

2.2 Kernel Adaptive MCMC Proposals

The previously described SMC algorithms are based on target covariance and gradients. We now review how these quantities were previously modelled using kernel methods in the context of MCMC. Note that any form of adaptation in MCMC requires care in order to preserve ergodicity of the resulting Markov chain, and some form of vanishing adaptation is needed [1,18]. This can be achieved e.g. by updating the proposal family with vanishing probability [20,22].

Covariance emulator. Sejdinovic et al. [20] introduced a kernel covariance emulator as a method for adapting the proposal distribution in a Metropolis-Hastings MCMC algorithm, based on the history of the Markov chain $\mathbf{X} = \{X_1, X_2, \ldots\}$. The idea is to represent covariance of the target as an empirical Gaussian measure with mean $\mu_{\mathbf{X}} := \frac{1}{|\mathbf{X}|}\sum_{X\in\mathbf{X}} \mathbf{k}(X, \cdot)$ and covariance $\frac{1}{|\mathbf{X}|}\sum_{X\in\mathbf{X}} \mathbf{k}(X, \cdot) \otimes \mathbf{k}(X, \cdot) - \mu_{\mathbf{X}} \otimes \mu_{\mathbf{X}}$ in a RKHS with kernel $\mathbf{k}$. This measure can be sampled from exactly, and it is possible to (approximately) map samples back to the original space.

Sejdinovic et al. [20] showed that it is possible to integrate out the RKHS proposal analytically, which elegantly results in a *closed form* Gaussian proposal density in the input space. For a Gaussian kernel, the proposal at particle X_j locally aligns to the structure of the posterior at X_j, and is given by

$$q_{\mathrm{KAMH}}(\cdot|X_j) = \mathcal{N}(\cdot|X_j, \gamma^2 I + \nu^2 M_{\mathbf{X},X_j} C M_{\mathbf{X},X_j}^\top),$$

where $C = I - \frac{1}{n}11^\top$ is a centering matrix and $M_{\mathbf{X},X_j}$ collects kernel gradients with respect to all particles,

$$M_{\mathbf{X},X_j} = 2[\nabla_x \mathbf{k}(x, X_1)|_{x=X_j}, ..., \nabla_x \mathbf{k}(x, X_N)|_{x=X_j}].$$

Additional exploration noise with variance γ^2 avoids that the proposal collapses in unexplored regions of the input space.

Gradient emulator. To overcome random walk behaviour of KAMH, Strathmann et al. [22] constructed an algorithm that adaptively learns the gradient structure of the Markov chain history, and mimics Hamiltonian dynamics using the learned gradients. This is done by fitting an un-normalised infinite dimensional exponential family model with density function $\exp(\langle f, k(x,\cdot)\rangle_{\mathcal{H}} - A(f))$. Here, $\langle f, k(x,\cdot)\rangle_{\mathcal{H}} = f(x)$ is the inner product between natural parameters f and sufficient statistics $k(x,\cdot)$ in a RKHS $\mathcal{H}$, and $A(f)$ is the (intractable) log-partition function. Remarkably, it is possible to efficiently estimate f via minimising the expected L^2 error of $\nabla_x f(x)$ without dealing with $A(f)$. Combining this model with a further approximation, based on random basis functions [KMC finite; 22], allows for efficient on-line updates of the emulator. Similar to Hamiltonian Monte Carlo, the resulting KHMC algorithm offers substantial improvements over random walks. Tt does so, however, *without* requiring gradient information of the target. This allows application to intractable likelihood models, where we cannot evaluate the target densities π_t even up to a normalizing constant, and gradients are similarly unavailable.

3 Kernel Sequential Monte Carlo

We now develop a kernel sequential Monte Carlo framework. KSMC is based on combining classical adaptive SMC with the emulator based proposals of kernel adaptive MCMC. In general, once a kernel emulator is fitted to past particle systems, we can use it in either of two ways: as proposals for MH rejuvenation steps inside SMC or as importance densities in PMC.

Key contributions. Our main contribution is to combine several yet unconnected pieces of literature into a novel framework that performs favourably compared to its individual parts: adaptive SMC proposals, SMC for intractable likelihoods, and kernel emulators for efficient proposals. This combination is simple yet very natural: As compared to (kernel) adaptive MCMC, the KSMC framework (i) circumvents the need for vanishing adaptation, (ii) can represent multimodality,

(iii) allows to estimate model evidence in a straight-forward manner. On the other hand, as compared to plain adaptive SMC and PMC, the use of kernel emulators (iv) leads to faster convergence for nonlinear targets.

We present two novel algorithms, KASMC and KGRIS, both of which are weighted and kernelised generalisations of existing kernel MCMC and SMC respectively. These modifications can lead to significant mixing improvements in practice. Our contribution furthermore includes variance reduction techniques that are critical in practice. In particular, Naïve implementations can suffer from high variance induced by simplifications. As this results in lower quality emulators, too high variance would be self-reinforcing and is to be strictly avoided.

3.1 Kernel Adaptive Rejuvenation: KASMC

We can use both kernel emulators for the rejuvenation step of SMC. More specifically, at time-step $t+1$, we target distribution π_{t+1}, based on a particle system approximating π_t. After re-weighting, the new system $\{(W_{t+1,i}, X_{t+1,i})\}_{i=1}^N$ is a weighted approximation to π_{t+1}. We here focus on the nonlinear covariance emulator which can be either fitted using the equally-weighted re-sampled values $\widetilde{\mathbf{X}}_t$, or the original particle set with weights $\widetilde{\mathbf{W}}_t$. The proposal distribution for Algorithm 1 at X then is exactly q_{KAMH}. As in KAMH, this results in covariance matrices for Gaussian proposals which locally align with the target [20], now taking the SMC particle weights into account. The resulting kernel adaptive SMC sampler (KASMC) inherits KAMH's ability to explore non-linear targets more efficiently than proposals based on estimating global covariance structure such as in Fearnhead and Taylor [11] and Haario et al. [13]. Figure 1 (left) shows a simple illustration of a global (ASMC) and local proposal distribution (KASMC). Compared to previous work on kernel induced local covariance matrices for MCMC [20], we implement a random features approximation in order to enable computationally efficient updates with information gained from new samples [17].

3.2 Kernel Induced Importance Densities: KGRIS

Another way to use kernel-based emulators is for generating proposals which are corrected by importance sampling, i.e. in PMC. In our second approach, a kernel emulator is fitted to weighted particles, which were previously corrected via importance weights. As an example, we here use the kernel gradient emulator by Strathmann et al. [22], in its finite dimensional approximation (KMC finite), c.f. [22, Proposition 2].

The log density of the approximate estimator takes the simple form $f(x) = \theta^\top \phi_x$, where $\phi_x \in \mathbb{R}^m$ is an embedding of x into an m-dimensional feature space, and $\theta \in \mathbb{R}^m$ is estimated by $\hat{\theta} = C^{-1}b$ from samples x. Given a weighted particle system $\{(W_{t,i}, X_{t,i})\}_{i=1}^N$, then b, C are weighted averages of the form

$$b := -\frac{1}{\sum_{i=1}^{N} W_{t,i}} \sum_{i=1}^{N} W_{t,i} \sum_{\ell=1}^{d} \ddot{\phi}_x^\ell,$$
$$C := \frac{1}{\sum_{i=1}^{N} W_{t,i}} \sum_{i=1}^{N} W_{t,i} \sum_{\ell=1}^{d} \dot{\phi}_x^\ell \left(\dot{\phi}_x^\ell\right)^\top,$$

with element-wise derivatives $\dot{\phi}_x^\ell := \frac{\partial}{\partial x_\ell}\phi_x$ and $\ddot{\phi}_x^\ell := \frac{\partial^2}{\partial x_\ell^2}\phi_x$. Note that the estimator can be updated in an online fashion once the particle system changes. Rather than simulating Hamiltonian dynamics to generate a proposal, we here take single gradient steps, i.e. the Markov density at in Algorithm 1 at X is $q_t(\cdot|X) = \mathcal{N}(\cdot|X + \delta\nabla f(X), \nu^2\Sigma_t)$ for some parameters $\delta > 0, \nu^2 > 0$. This keeps the risk of divergence due to wrongly estimated gradients low. We arrive at kernel GRIS, a gradient-free variant of GRIS [19].

3.3 Controlling Emulator Variance in PMC

PMC is somewhat sensitive to badly scaled proposals, as these are not rejected as in a Metropolis-Hastings step. In particular for gradient emulators used within PMC, variance reduction is important to avoid numerical divergence. The original PMC paper introduces re-sampling in order to deal with un-weighted instead of weighted samples [2], though at the cost of an increased variance. While some approaches avoid re-sampling altogether [3], we consider re-sampling here as a way to obtain a set of locations $\{\widetilde{X}_t^i\}_{i=1}^N$ for our Markov proposal components of the mixture $q_t^{\text{Mixt}}(\cdot) = \frac{1}{N}\sum_{i=1}^N q_t(\cdot|\widetilde{X}_t^i)$, due to better behaving variance in high dimension. With re-sampling, Monte Carlo variance only grows as $O(D)$ rather than $O(\exp(D))$ without re-sampling, where D is the dimensionality [8].

Given a re-sampled number of N particles and the updated emulator q_t, we simulate from the mixture distribution q_t^{Mixt} with stratification, i.e. we draw exactly one sample from each of the equally weighted mixture components. Another view of this scheme is to draw a single realisation from $q_t(\cdot|\widetilde{X}_t^i)$ for all $i = 1, \ldots, N$ and Rao-Blackwellise. Finally, we can view the scheme as an instance of the deterministic mixture idea [10]. Without this technique, i.e. using weights $\pi(\cdot)/q_t(\cdot|\widetilde{X}_t^i)$, variance might grow catastrophically large, as too high variance can be self-reinforcing by resulting in emulators of low quality.

4 Evaluation

We empirically evaluate performance of KASMC on a simple non-linear target, on a multi-modal sensor network localisation problem, and in estimating Bayesian model evidence in a model with an intractable likelihood on a real-world dataset. The final experiment uses a challenging stochastic volatility model with S&P 500 data from Chopin et al. [5] to evaluate KGRIS.

For the KASMC experiments on static target distributions, a sequence of incremental target densities can be defined using a geometric bridge with $\pi_t \propto \pi_0^{1-\rho_t}\pi^{\rho_t}$

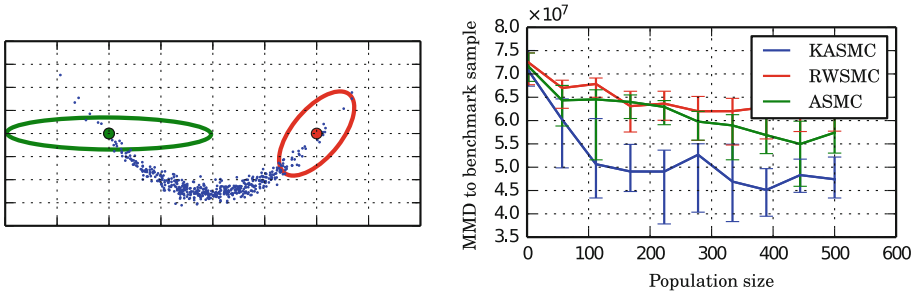

Fig. 1. Left: Proposal distributions around one of many particles (blue) for each KASMC (red) and ASMC (green). KASMC proposals locally align to the target density while ASMC's global covariance estimate might result in poor MH rejuvenation moves. **Right:** Improved convergence of all mixed moments up to order 3 of KASMC compared to using SMC with static or adaptive Metropolis-Hastings steps. (Color figure online)

for some initial distribution π_0, where $(\rho_t)_{t=1}^T$ is an increasing sequence satisfying $\rho_T = 1$. The bandwidth parameter of the kernel emulator models is set to the median distance between particles [12].

We also note these algorithms have a free scaling parameter ν^2, which we would like to adapt online. To accomplish parameter tuning, we use the standard framework of stochastic approximation for tuning MCMC kernels [1], i.e. tuning acceptance rate α_t towards an asymptotically optimal acceptance rate $\alpha_{\text{opt}} = 0.234$ for random walk proposals [18]. After the MCMC rejuvenation step, a Rao-Blackwellised estimate $\hat{\alpha}_t$ of expected acceptance probability is available by simply averaging the acceptance probabilities for all MH proposals. Then, set $\nu_{t+1}^2 = \nu_t^2 + \lambda_t(\hat{\alpha}_t - \alpha_{\text{opt}})$ for some non-increasing sequence $\lambda_1, \ldots, \lambda_T$. This strategy of approximating optimal scaling assumes that consecutive targets are close enough so that the acceptance rate when using ν_t^2 to target π_t provides information about the expected acceptance rate when using ν_t^2 with target π_{t+1}. This is discussed further in the supplemental material.

4.1 KASMC: Improved Convergence on Synthetic Nonlinear Target

We begin by studying convergence of KASMC compared to existing algorithms on a simple benchmark example: the strongly twisted banana-shaped distribution in $D = 8$ dimensions used in Sejdinovic et al. [20]. This distribution is a multivariate Gaussian with a non-linearly transformed second component, defined as

$$\mathcal{B}(y; b, v) = \mathcal{N}(y_1; 0, v)\mathcal{N}(y_2; b(y_1^2 - v), 1) \prod_{j=3}^{D} \mathcal{N}(y_j; 0, 1).$$

We compare SMC algorithms using different rejuvenation MH steps: a static random walk Metropolis move (RWSMC) with fixed scaling $\nu = 2.38/\sqrt{D}$,

ASMC, and KASMC using a Gaussian RBF kernel. For the latter two algorithms, all particles are used to compute the proposal, and a fixed learning rate of $\lambda = 0.1$ is chosen to adapt scale parameters. Starting with particles from a multivariate Gaussian $\mathcal{N}(0, 50^2)$, we use a geometric bridge that reaches the target $\mathcal{B}(y; b = 0.1, v = 100)$ in 20 steps. We repeat the experiment over 30 runs. Figure 1 (right) shows that KASMC achieves faster convergence of the first 3 moments, i.e. in MMD[2] distance to a large benchmark sample.

4.2 A Multi-modal Application: Sensor Network Localisation

We next study performance of KASMC on a multi-modal target arising in a real-world application: inferring the locations of S sensors within a network, as discussed in [14,15]. We here focus on the static case: assume a number of stationary sensors that measure distance to each other in a 2-dimensional space; a distance measurement is successful with a probability that decays exponentially in the squared distance, and the observation is missing otherwise. If distance is measured, it is corrupted by Gaussian noise. The posterior over the unknown sensor locations forms an extremely constrained non-linear and multi-modal distribution induced by the spatial set-up.

Assume S sensors with unknown locations $\{x_i\}_{i=1}^S \subseteq \mathbb{R}^2$. Define an indicator variable $Z_{i,j} \in \{0, 1\}$ for the distance $Y_{ij} \in \mathbb{R}^+$ between a pair of sensors (x_i, x_j) being either observed ($Z_{i,j} = 1$) or not ($Z_{i,j} = 0$), according to

$$Z_{i,j} \sim \texttt{Binom}\left(1, \exp\left(-\frac{\|x_i - x_j\|_2^2}{2R^2}\right)\right).$$

If the distance is observed, then Y_{ij} is corrupted by Gaussian noise, i.e.

$$Y_{i,j} | Z_{i,j} = 1 \sim \mathcal{N}\left(\|x_i - x_j\|, \sigma^2\right),$$

and $Y_{i,j} = 0$ otherwise.

Previously, [14] focussed on MAP estimation of the sensor locations, and [15] focussed on a well-conditioned case ($S = 8$ sensors and $B = 3$ base sensors with known locations) that results in almost no ambiguity in the posterior. We argue that Bayesian quantification of uncertainty is more important for cases where noise and missing measurements *does not* allow to reconstruct the sensor locations exactly. We therefore reuse the dataset from [15] ($R = 0.3$, $\sigma^2 = 0.02$)[3], but only use the first $S = 3$ locations/observations. In order to encourage ambiguities in the localisation task, we only use the first 2 base sensors of [15] with known locations that each do observe distances to the S unknown sensors but not of each other. Unlike [15], we use a Gaussian prior $\mathcal{N}(\mathbf{0.5}, I)$ to avoid the posterior being situated in a bounded domain.

[2] The maximum mean discrepancy, here using a polynomial kernel of order 3, quantifies differences of all mixed moments up to order 3 of two independent sets of samples.

[3] Downloaded from http://www.ics.uci.edu/~slan/lanzi/CODES_files/WHMC-code.zip on 8/Oct/2015.

Figure 2 shows the marginalised posterior for one run each of KASMC (SMC) and KAMH (MCMC), where we matched the number of likelihood evaluations (500,000). We run KASMC using $10,000$ particles and a bridge length of 50, and MCMC-KAMH for $50 \times 10,000$ iterations of which we discard half as burn-in; both were initialized with samples from the prior. Tuning parameters ν^2 are set using a diminishing adaptation schedule $\lambda_t = 1/\sqrt{t}$ for KAMH and a fixed learning rate $\lambda_t = 1$ for KASMC. MCMC is not able to traverse between the multiple modes and interpretations of the data, in contrast to SMC.

In order to compare ASMC to KASMC, we created a benchmark sample via running 100 standard MCMC chains (randomly initialised to cover all modes) each for 50000 iterations, discarding half the samples as burn-in, and randomly down-sampling to a size of 100. We then compute the empirical MMD distance to the output of the individual algorithms, averaged over 10 runs. For the chosen number of sensors, ASMC and KASMC perform similarly. With less sensors, i.e. more ambiguity, KASMC produces samples with both less MMD distance from a benchmark sample and less variance. For example, for a set-up with $S = 2$ and 1000 particles, we get a MMD distance to a benchmark sample of 0.76 ± 0.4 for KASMC and 0.94 ± 0.7 for ASMC.

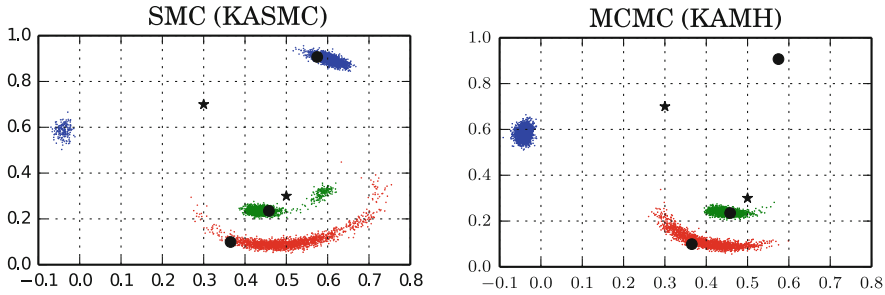

Fig. 2. Posterior samples of unknown sensor locations (in color) by kernel-based SMC and MCMC on the sensors dataset. The set-up of the true sensor locations (black dots) and base sensors (black stars) causes uncertainty in the posterior. SMC recovers all modes while MCMC does not. The posterior has a clear non-linear structure. (Color figure online)

4.3 KASMC: Evidence Estimation in Gaussian Process Classification

Following Sejdinovic et al. [20], we consider Bayesian classification on the UCI Glass dataset, discriminating window glass from non-window glass, using a Gaussian process (GP). It was found that the induced posterior is indeed non-linear [20,22]. In Sejdinovic et al. [20], samples from the marginal posterior over GP hyper-parameters were simulated (the GP latent variables integrated out). We emphasise a different point here: KSMC's ability to estimate the model evidence as compared to KAMH, and its faster convergence compared to ASMC.

Consider the joint distribution of latent variables $\mathbf{f}$, labels $\mathbf{y}$ (with covariate matrix X), and hyper-parameters θ, given by

$$p(\mathbf{f}, \mathbf{y}, \theta) = p(\theta)p(\mathbf{f}|\theta)p(\mathbf{y}|\mathbf{f}),$$

where $\mathbf{f}|\theta \sim \mathcal{N}(0, \mathcal{K}_\theta)$, with $\mathcal{K}_\theta$ modelling the covariance between latent variables evaluated at the input covariates. Consider the binary logistic classifier, i.e. $p(y_i|f_i) = \frac{1}{1-\exp(-y_i f_i)}$ where $y_i \in \{-1, 1\}$. In order to perform Bayesian model selection (i.e. comparing different covariance functions), we need to estimate the model evidence of the marginal posterior given the hyper-parameters. Here, the marginal likelihood $p(\mathbf{y}|\theta)$ is intractable for non-Gaussian likelihoods $p(\mathbf{y}|\mathbf{f})$.

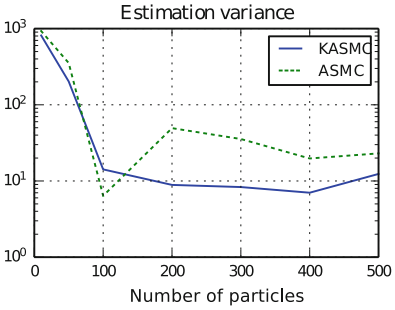

Fig. 3. Estimating model evidence of a GP using the IS^2 framework. The plot shows the MC variance over 50 runs as a function of the size of the particle system.

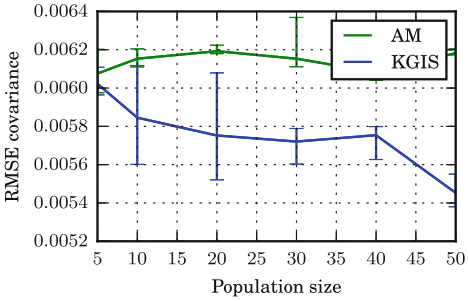

Fig. 4. Convergence of RMSE for estimating all elements of the posterior covariance matrix of the stochastic volatility model.

We estimate model evidence for the GP classifier equipped with a standard Gaussian Automatic Relevance Determination (ARD) covariance kernel; an unbiased estimate can be obtained using importance sampling

$$\hat{p}(\mathbf{y}|\theta) := \frac{1}{n_{\text{imp}}} \sum_{i=1}^{n_{\text{imp}}} p(\mathbf{y}|\mathbf{f}^{(i)}) \frac{p(\mathbf{f}^{(i)}|\theta)}{q(\mathbf{f}^{(i)}|\theta)}, \tag{2}$$

where $\{\mathbf{f}^{(i)}\}_{i=1}^{n_{\text{imp}}} \sim q(\mathbf{f}|\theta)$ are n_{imp} importance samples, e.g. from a Laplace approximation of $p(\mathbf{f}|\mathbf{y}, \theta)$. We here do not tune the number of 'inner' importance samples, but follow [20] and use $n_{\text{imp}} = 100$.

Figure 3 shows that evidence estimates of KASMC exhibit less variance than those of ASMC. The ground truth model evidence was established via running 20 SMC instances using $N = 1000$ particles and a bridge length of 30, and averaging their evidence estimates. The experiment is performed 50 times, using $N = 100$ particles and a bridge length of 20, starting from he prior on the log hyper-parameters $\pi_0 = p(\theta_d) \equiv \mathcal{N}(0, 5^2)$. The learning rate is constant $\lambda_t = 1$, and adaptation is towards an acceptance rate of 0.23.

4.4 KGRIS: Stochasitic Volatility Model with Intractable Likelihood

A particularly challenging class of Bayesian inverse problems are stochastic volatility models. As time series models, they often involve high-dimensional nuisance variables, which usually cannot be integrated out analytically. Furthermore, risk management necessitates to account for parameter and model uncertainty, and models have to capture the non-linearities in the data [5]. We here concentrate on the prediction of daily volatility of asset prices, reusing the model and dataset studied by Chopin et al. [5] to evaluate KGRIS. Due to the lack of analytically available gradients for this model, we compare two gradient free PMC versions: KGRIS and a random walk PMC with global covariance adaptation in the style of Haario et al. [13].

Let s_t be the value of some financial asset on day t, then $y_t = 10^{(5/2)} \log(s_t/s_{t-1})$ is called the log-returns (upscaling for numerical reasons). We model the observed log-return y_t as dependent on a latent v_t by the observation equation

$$y_t = \mu + \beta v_t + \sqrt{v_t}\epsilon_t$$

for $t \leq 1$. Here ϵ_t is a sequence of i.i.d. standard Gaussian errors and v_t is assumed to be a stationary stochastic process known as the *actual volatility*. Chopin et al. [5] develop a hierarchical model for v_t based on the idea of analytically integrating a continuous time volatility model over daily intervals [for details see 5]. Using this construction, the (discrete time) v_t is parameterised by stationary mean ξ and variance ω^2 of the so called *spot volatility* and the exponential decay λ of its auto-correlation. This results in the following model for the actual volatility v_t:

$$k \sim \text{Pois}(\lambda\xi^2/\omega^2), c_{1:k} \sim \text{U}(t, t+1), e_{1:k} \sim \text{Exp}(\xi/\omega^2)$$

$$z_{t+1} = z_t \exp(-\lambda) + \sum_{j=1}^{k} e_j \exp(-\lambda(t+1-c_j))$$

$$v_{t+1} = \lambda^{-1}\left(z_t - z_{t+1} + \sum_{j=1}^{k} e_j\right), x_{t+1} = (v_{t+1}, z_{t+1})^\top$$

where z_t is the discretely sampled spot volatility process and $(v_{t+1}, z_{t+1})^\top$ is the Markovian representation of the state process. The variables $k, c_{1:k}$ and $e_{1:k}$ are generated independently for each time period. For $k = 0$, the set $1 : k$ is defined to be empty. The dynamics imply $\Gamma(\xi^2/\omega^2, \xi^2/\omega^2)$ to be the stationary distribution for z_t, which is also used as the initial distribution on z_0. The parameters of the model are $\theta = (\mu, \beta, \xi, \omega^2, \lambda)$ and the likelihood is intractable. Chopin et al. [5] developed a sampler for θ based on iterated batch importance sampling using nested SMC with pseudo-marginal MCMC moves for integrating out the x_t and dubbed their approach SMC^2.

In our experiment, we use KGRIS proposals in a population Monte Carlo setting, i.e. without resorting to MCMC moves at all. We re-use the code developed for the original SMC^2 paper in order to integrate out the x_t and thus

get likelihood estimates, with the same settings for algorithm parameters. The observed s_t are the 753 observations from consecutive days of the S&P 500 index also used by Chopin et al. [5]. KGRIS uses a particle system of increasing sizes with each particle going through 100 iterations. See Fig. 5 in the Appendix for a plot of the pair-wise marginals of this posterior.

We use the same vague priors as Chopin et al. [5],

$$\mu \sim \mathcal{N}(0, \sigma^2 = 2), \beta \sim \mathcal{N}(0, \sigma^2 = 2), \xi \sim \text{Exp}(0.2)$$
$$\omega^2 \sim \text{Exp}(0.2), \lambda \sim \text{Exp}(1).$$

Figure 4 shows that the incorporated gradients lead to better performance of KGRIS in estimating the target covariance matrix. This is in-line with the finding that GRIS improves over pure random walk methods [19].

5 Discussion

In this paper, we developed a framework for kernel sequential Monte Carlo. KSMC adaptively learns the target geometry via kernel emulators and subsequently uses this information for local proposals. KSMC is especially attractive in the case where likelihoods and gradients are intractable. We instantiated two algorithms within KSMC: estimating nonlinear covariance in combination with MCMC rejuvenation and estimating gradients in combination with importance sampling proposals. Both significantly outperform state-of-the-art gradient-free SMC algorithms in practice. We conclude with some discussion on computational complexity, more general usage of the learned emulators, and on the relative benefits of PMC in the kernel setting.

Computational costs & increasing dimensions. While adaptive schemes for SMC (and MCMC) can increase statistical efficiency of the sampling scheme, they impose additional computational costs. Somewhat surprisingly, however, these relatively large costs do not severely impact the efficiency per runtime ratio in practice. The reason is that in the context of intractable likelihoods, the computational cost of fitting a kernel emulator is typically dominated by the larger cost of evaluating model likelihood. In our real-world experiments on GP classification and a stochastic volatility model in Sects. 4.3 and 4.4, a profiler reveals that less than 5% of the overall wall-clock time is spent in computing kernel informed proposals. This effect increases with dataset size and model complexity, as evaluating likelihood gets more costly. Clearly however, in the case where we need not resort to pseudo-marginal or SMC^2 type samplers, the application of kernel based estimators might result in slower sampling without much gain in Monte Carlo error.

In growing dimensions, the number of data required to sufficiently estimate nonlinear covariance and gradients quickly becomes infeasible. High dimensional sampling problems typically arise in non-parametric models, e.g. Gaussian processes, where each data point comes with additional parameters.

In the intractable likelihood framework that we consider here, however, the marginal posterior over hyper-parameters typically is independent of such latent variables — and therefore usually of moderate dimension. Random walk methods, which are the default choice for intractable likelihoods, scale badly in high dimensions themselves [16]. Our method is an improvement in the intermediate case: closed form gradients are not available, but the dimensionality of the problem allows to estimate the target geometry just accurately enough to improve mixing. Strathmann et al. [22] reported their gradient estimator to scale up from dozens to a hundred dimensions on laptop computers, depending on smoothness properties of the target. It is an active area of research to further scale up these techniques by exploiting structure in the target density.

Emulators as a posterior approximation. The kernel approximation of the target density could be considered itself as an output of our algorithms, representing the posterior directly instead of using the kernel approximation within a sampler. There are a number of problems with this approach though: firstly, we note that our emulator models do not need to be perfect to generate useful proposals, therefore allowing us to exploit posterior structure much earlier (even with non-perfect model fit) during sampling, still resulting in a correct SMC sampler. Also, approximating integrals of test functions with respect to the posterior using the kernel approximation is not possible in closed form, while it is straight forward using a Monte Carlo sum. For example, assume a log density model $f(x) = \sum_i \alpha_i \mathbf{k}(x_i, x)$. For the Gaussian kernel $\mathbf{k}(x, y) = \exp(-||x - y||^2)$, the density is the exponential of a sum of Gaussian centred at the points x_i. Computing an integral as simple as the posterior mean, $\mu = Z^{-1} \int x \exp(\sum_i \alpha_i \exp(-||x_i - x||^2))\mathrm{d}x$, already is intractable, even if the evidence Z were known. Thirdly, it is not possible to sample from the kernel emulator directly using ordinary Monte Carlo. One could imagine running a second MCMC/SMC targeting the emulator model. Not only would this defeat the purpose of the algorithm (this is the problem we are trying to solve in the first place), it also leads to samples that are not guaranteed to consistently estimate posterior expectations unlike kernel SMC or kernel MCMC.

SMC versus PMC for kernel based proposals. The consensus in the wider SMC community is that using an artificial sequence of proposal distributions for sampling from a static target is preferable to the PMC approach. This is based on the fact that the coverage of the final target is better in these tempering-style algorithms. It however results in a considerable computational investment for those iterations where an intermediate target is considered.

We also note that on-line updates of the kernel emulator are not possible: the target changes in every iteration. The contrary is true in PMC, where the the actual distribution of interest is targeted in every iteration. Here, a popular approximation technique of kernels is a good fit: By expressing the emulator model in terms of finite dimensional random Fourier features, we can perform cheap on-line updates [22]. The emulator therefore can accumulate information

from all PMC iterations without the computational efforts of re-computing its solution, providing a relative advantage to SMC in this context.

Acknowledgments. I.S. was supported by a PSL postdoc grant and DFG through grant CRC 1114 "Scaling Cascades in Complex Systems", Project B03 "Multilevel coarse graining of multiscale problems". H.S. was supported by the Gatsby Chaitable foundation. B.P. was supported by The Alan Turing Institute under the EPSRC grant EP/N510129/1.

A Implementation details

In this section, we cover a number of implementation details for using KASMC in practice, such as optimal scaling, adaptive re-sampling and re-weighting between iterations.

A.1 Scaling Parameters

Similar to other MH proposals, KAMH has a free scaling parameter denoted ν^2 which we would like to adapt after one SMC iteration. To accomplish parameter tuning, we use the standard framework of stochastic approximation for tuning MCMC kernels [1], i.e. tuning acceptance rate α_t towards an asymptotically optimal acceptance rate $\alpha_{\text{opt}} = 0.234$ for random walk proposals [18]. More precisely, after the MCMC rejuvenation step, a Rao-Blackwellised estimate $\hat{\alpha}_t$ of expected acceptance probability is available by simply averaging the acceptance probabilities for all MH proposals. Then, set

$$\nu_{t+1}^2 = \nu_t^2 + \lambda_t(\hat{\alpha}_t - \alpha_{\text{opt}}) \tag{3}$$

for some non-increasing sequence $\lambda_1, \ldots, \lambda_T$. This strategy of approximating optimal scaling assumes that consecutive targets are close enough so that the acceptance rate when using ν_t^2 to target π_t provides information about the expected acceptance rate when using ν_t^2 with target π_{t+1}. As an alternative to this, one could treat ν_t^2 as an auxiliary random variable and define a distribution over it designed to maximise expected utility, an approach taken in the adaptive SMC sampler [11].

A.2 Construction of a Target Sequence

One possibility for constructing a sequence of distributions is the geometric bridge defined by

$$\pi_t \propto \pi_0^{1-\rho_t} \pi^{\rho_t}$$

for some initial distribution π_0, where $(\rho_t)_{t=1}^T$ is an increasing sequence satisfying $\rho_T = 1$. This is the construction used in the experimental section. Another construction is to use a mixture $\pi_t \propto (1-\rho_t)\pi_0 + \rho_t \pi$. When π is a Bayesian posterior, one can also add more data with increasing t, e.g. by defining the intermediate distributions as $\pi_t(X) = \pi(X|d_1, \ldots, d_{\lfloor \rho_t D \rfloor})$ where d_j is a datapoint

and D is the number of data points. This results in an online inference algorithm called Iterated Batch Important Sampling (IBIS) [4]. In IBIS especially, we can apply non-diminishing adaptation, unlike in adaptive MCMC.

When using a distribution sequence that computes the posterior density π using the full dataset (such as the geometric bridge or the mixture sequence), one can reuse the intermediate samples when targeting π_t for posterior estimation. As the value of π is computed for the geometric bridge and the mixture sequence, we re-use the weight $\pi(X)/\pi_{t-1}(X)$ for posterior estimation while employing $\pi_t(X)/\pi_{t-1}(X)$ to inform proposal distributions at iteration t. This way, the evaluation of π (which is typically costly) is put to good use for improving the posterior estimate.

As a simple alternative, leading to the algorithm known as Population Monte Carlo (PMC) [2], we can simply target the final distribution π at each iteration, i.e. with all $\pi_t = \pi$. The original work on PMC exhibited striking resemblance of commonly used MCMC methods such as Random Walk metropolis, often finding that the same proposal kernel with PMC produces better estimates than with MCMC [2].

A.3 Re-weighting and Adaptive Re-sampling

The fact that the weighted approximation to the final target is returned in our algorithm stems from the fact that this approximation has lower variance than the re-sampled particle system [8]. This is why in practice re-sampling might not be performed at every iteration. Rather, re-sampling only when Effective Sample Size (ESS) for the current target falls below a certain threshold will decrease Monte Carlo variance. For details we refer to reviews on SMC [8,9]. Furthermore, care should be taken with respect to implementation of re-weighting: caching values between iterations saves much computation time.

A.4 Intractable Likelihoods and Evidence Estimation

In the the case where likelihoods are intractable, SMC is still a valid algorithm when likelihood values can be estimated unbiasedly. This can be done using e.g. importance sampling or SMC [5,23]. A simple way to think about such nested estimation schemes is in terms of an extended sampling space that spans the actual parameters of interest as well as any nuisance variables. Intractable likelihoods usually result in unavailability of gradients. Consequently, efficient gradient-based sampling schemes based such as GRIS or HMC are unavailable. Current practice there is based on moving particles using random walk schemes solely.

An important issue in Bayesian model selection and averaging is that of estimating the normalizing constant, or *evidence*. The evidence is the marginal probability of the data under a model and can easily be estimated in SMC instantiations [8,11] – as compared to MCMC. This enables routine computation of Bayes factors and posterior model probabilities while also sampling from a posterior over parameters of each model. Under the assumption that the normalizing

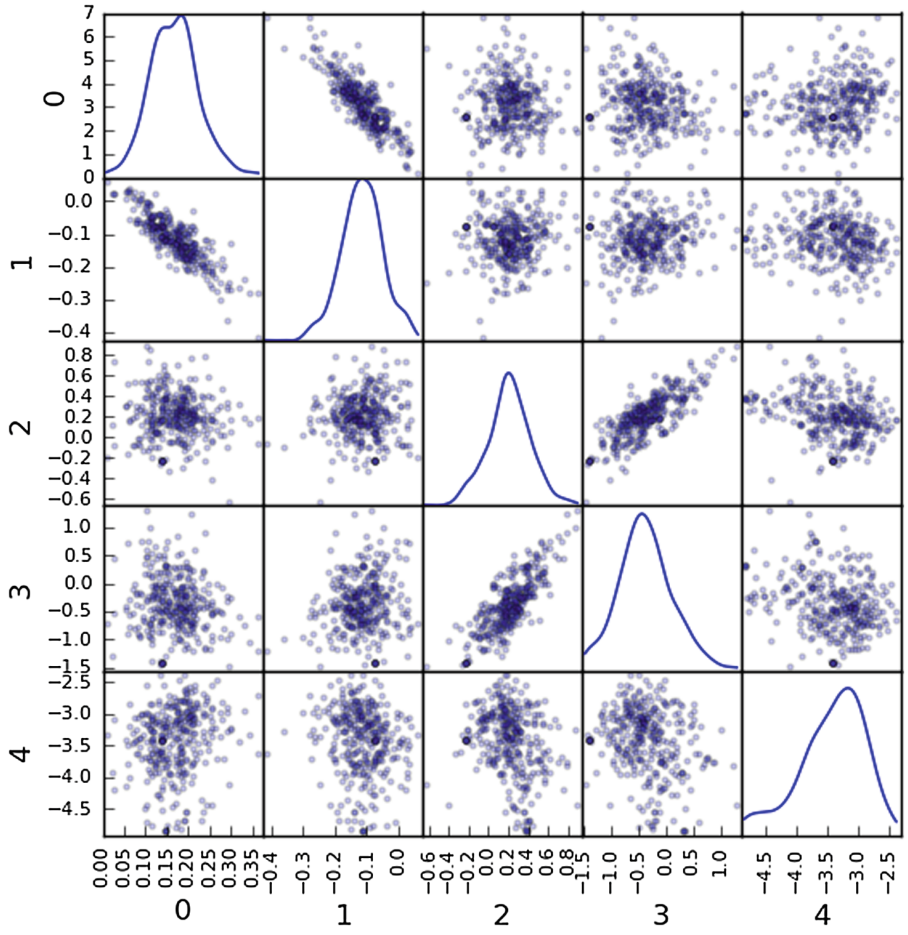

Fig. 5. Samples from the doubly stochastic volatility model used in Sect. 4.4.

constant Z_0 of π_0 (the distribution that is used for initially setting up the particle system) is known, one can estimate the ratio of normalizing constants of any two consecutive targets by

$$\frac{Z_t}{Z_{t-1}} \approx \frac{1}{N} \sum_{j=1}^{N} W_{t,j} \tag{4}$$

for $W_{t,j} = \pi_t(X_{t-1,j})/\pi_{t-1}(X_{t-1,j})$ and thus an estimate for $Z = Z_T$ can be found recursively by

$$Z = Z_T \approx Z_0 \prod_{t=1}^{T} \frac{1}{N} \sum_{j} W_{t,j} \tag{5}$$

starting with known value Z_0. When the likelihood is intractable and importance weights are noisy, evidence estimation is still valid [23, Lemma 3].

References

1. Andrieu, C., Thoms, J.: A tutorial on adaptive MCMC. Stat. Comput. **18**, 343–373 (2008). ISSN 09603174
2. Cappé, O., Guillin, A., Marin, J.M., Robert, C.P.: Population Monte Carlo. J. Comput. Graph. Stat. **13**(4), 907–929 (2004). ISSN 1061–8600
3. Cappé, O., Douc, R., Guillin, A., Marin, J.M., Robert, C.P.: Adaptive importance sampling in general mixture classes. Stat. Comput. **18**(4), 447–459 (2008). ISSN 0960–3174
4. Chopin, N.: A sequential particle filter method for static models. Biometrika **89**(3), 539–552 (2002). ISSN 0006–3444
5. Chopin, N., Jacob, P.E., Papaspiliopoulos, O.: SMC2: an efficient algorithm for sequential analysis of state space models. J. Royal Stat. Soc. Ser. B (Stat. Methodol.) **75**(3), 397–426 (2013)
6. Del Moral, P., Doucet, A., Jasra, A.: Sequential Monte Carlo samplers. J. Royal Stat. Soc. Ser. B (Stat. Methodol.) **68**(3), 411–436 (2006)
7. Douc, R., Cappé, O.: Comparison of resampling schemes for particle filtering. In: Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, pp. 64–69 (2005). ISBN 953-184-089-X
8. Doucet, A., Johansen, A.: A tutorial on particle filtering and smoothing: fifteen years later. Handb. Nonlinear Filtering, 4–6 (2009)
9. Doucet, A., de Freitas, N., Gordon, N.: An introduction to sequential Monte Carlo methods. In: Doucet, A., de Freitas, N., Gordon, N. (eds.) Sequential Monte Carlo Methods in Practice. Statistics for Engineering and Information Science, pp. 3–14. Springer, Heidelberg (2011). https://doi.org/10.1007/978-1-4757-3437-9_1
10. Elvira, V., Martino, L., Luengo, D., Bugallo, M.F.: Generalized multiple importance sampling. Technical report (2015)
11. Fearnhead, P., Taylor, B.M.: An adaptive sequential Monte Carlo sampler. Bayesian Anal. **2**, 411–438 (2013)
12. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.: A kernel two-sample test. J. Mach. Learn. Res. **13**(1), 723–773 (2012)
13. Haario, H., Saksman, E., Tamminen, J.: An adaptive metropolis algorithm. Bernoulli **7**(2), 223–242 (2001). ISSN 13507265
14. Ihler, A.T., Fisher, J.W., Moses, R.L., Willsky, A.S.: Nonparametric belief propagation for self-localization of sensor networks. IEEE J. Sel. Areas Commun. **23**(4), 809–819 (2005)
15. Lan, S., Streets, J., Shahbaba, B.: Wormhole Hamiltonian Monte Carlo. In: Twenty-Eighth AAAI Conference on Artificial Intelligence (2014)
16. Neal, R.M.: MCMC using Hamiltonian dynamics. Handb. Markov Chain Monte Carlo. Chapman and Hall/CRC (2011)
17. Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: Advances in Neural Information Processing Systems, pp. 1177–1184 (2007)
18. Rosenthal, J.S.: Optimal proposal distributions and adaptive MCMC. Handb. Markov Chain Monte Carlo **4**, 93–112 (2011). Chapman and Hall
19. Schuster, I.: Gradient importance sampling. arXiv preprint arXiv:1507.05781 (2015)

20. Sejdinovic, D., Strathmann, H., Garcia, M.L., Andrieu, C., Gretton, A.: Kernel adaptive metropolis-hastings. In: International Conference on Machine Learning (ICML), pp. 1665–1673 (2014)
21. Sriperumbudur, B., Fukumizu, K., Kumar, R., Gretton, A., Hyvärinen, A.: Density estimation in infinite dimensional exponential families. arXiv preprint arXiv:1312.3516 (2014)
22. Strathmann, H., Sejdinovic, D., Livingstone, S., Szabo, Z., Gretton, A.: Gradient-free Hamiltonian Monte Carlo with efficient kernel exponential families. In: NIPS (2015)
23. Tran, M.-N., Scharth, M., Pitt, M.K., Kohn, R.: Importance sampling squared for Bayesian inference in latent variable models. arXiv preprint arXiv:1309.3339 (2013)