Applied Probability Trust (5 February 2018)

MONTE CARLO FUSION

HONGSHENG DAI,* *University of Essex*

MURRAY POLLOCK,** *University of Warwick*

GARETH ROBERTS,*** *University of Warwick*

Abstract

This paper proposes a new theory and methodology to tackle the problem of unifying distributed analyses and inferences on shared parameters from multiple sources, into a single coherent inference. This surprisingly challenging problem arises in many settings (for instance, expert elicitation, multi-view learning, distributed 'big data' problems etc.), but to-date the framework and methodology proposed in this paper (Monte Carlo Fusion) is the first general approach which avoids any form of approximation error in obtaining the unified inference. In this paper we focus on the key theoretical underpinnings of this new methodology, and simple (direct) Monte Carlo interpretations of the theory. There is considerable scope to tailor the theory introduced in this paper to particular application settings (such as the big data setting), construct efficient parallelised schemes, understand the approximation and computational efficiencies of other such unification paradigms, and explore new theoretical and methodological directions.

Keywords: Fork-and-join; Fusion; Langevin diffusion; Monte Carlo

2010 Mathematics Subject Classification: Primary 65C05;65C60

Secondary 62C10;65C30

* Postal address: Department of Mathematical Sciences, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ

* Email address: hdaia@essex.ac.uk

** Postal address: Department of Statistics, University of Warwick, Gibbet Hill Road, Coventry, CV4 7ES

** Email address: m.pollock@warwick.ac.uk; gareth.o.roberts@warwick.ac.uk

1. Introduction

A common problem arising in statistical inference is the need to unify distributed analyses and inferences on shared parameters from multiple sources, into a single coherent inference. This unification (or what we term 'fusion') problem can arise either explicitly due to the nature of a particular application, or artificially as a consequence of the approach a practitioner takes to tackling an application.

Typically there will exist no closed form analytical approach to unifying distributed inferences, and so we focus on a Monte Carlo approach. Stated generally, in this paper we are interested in sampling (without error) the following d-dimensional (fusion) target density,

$$f(\boldsymbol{x}) \propto f_1(\boldsymbol{x}) \cdots f_C(\boldsymbol{x}), \tag{1}$$

where each $f_c(\boldsymbol{x})$ ($c \in \{1, \ldots, C\}$) is a density (up to a multiplicative constant) representing one of the C distributed inferences we wish to unify. Each $f_c(\boldsymbol{x})$ (which we term a sub-posterior) may in practice itself be represented by a Monte Carlo sample ($\tilde{f}_c(\boldsymbol{x})$), and in this paper we assume we are able to sample (directly and exactly) from each $f_c(\boldsymbol{x})$.

Specific examples of this problem arising naturally in an application include: expert elicitation (Berger (1980), Genest and Zidek (1986)), in which the (distributional) views of multiple experts on a topic (or set of parameters) have to be pooled into a single view before a decision-maker can make an informed decision; and, multi-view learning (Zhao et al. (2017), Li et al. (2015)) and meta-analysis (Fleiss (1993), Smith et al. (1995)), in which an interpretation could be that we are synthesising multiple inferences on a particular parameter set (computed on datasets which may or may not be of the same type), but the underlying raw data is not directly available for the unified inference. Obtaining the raw data may itself be an insoluble problem due to reasons including the nature of the original publication, data confidentiality, or simply time and storage constraints.

This fusion problem also arises artificially in a number of settings, particularly within modern statistical methodologies tackling 'big data'. The computational cost of algorithms such as Metropolis-Hastings, which is an iterative algorithm requiring full access at every iteration to the full dataset, scale poorly with increasing volumes of data unless a modification is found.

One common modification in light of the challenge of big data is to deploy a 'divide-and-conquer' approach (or more accurately termed 'fork-and-join' approach (Stamatakis and Aberer (2013))). In

this setting the full dataset is artificially split into a large number of smaller data sets, inference is then conducted on each smaller data set in isolation, and the resulting inferences are unified (see for instance, Scott et al. (2016), Agarwal and Duchi (2012), Neiswanger et al. (2014), Wang and Dunson (2013), Minsker et al. (2014), Srivastava et al. (2016) and Li et al. (2017)). The rationale for such an approach is that inference on each small data set can be conducted independently, and in parallel, and so one could exploit large clusters of computing cores to greatly reduce the elapsed time to conduct the full inference. The weakness of these approaches is that the fusion of the separately conducted inferences is inexact. It should be noted that divide-and-conquer methodologies will typically have additional constraints due to hardware concerns – such as minimising or removing any communication between computing cores to reduce the effect of *latency*. We focus in this paper on the general fusion problem, and so do not fully address the problem in the context of big data (to which we return in subsequent work).

The framework and methodology we outline in this paper (Monte Carlo Fusion) for sampling exactly from (1) can be viewed as a simple rejection sampling scheme on an extended space – we develop and sample from efficient proposal densities for (1), the samples from which we retain according to an appropriate acceptance probability. The mathematical complication in this paper is in computing the intractable acceptance probability – which requires the auxiliary simulation of collections of Brownian (or Ornstein-Uhlenbeck) bridges. Our fusion approach provides a principled way to understand the error in existing unification schemes, using a simple linear combination and correction of Monte Carlo samples (analogous to a traditional meta-analysis approach).

The presentation of this paper broadly follows this pedagogy: In Section 2 we present a density on an extended space which admits (1) as a marginal. The remainder of the paper then develops a rejection sampler for the extended density in Section 2. In Section 3 we develop general theory and methodology for sampling (1) based on a collection of independent Brownian bridges. In Section 4 we present a modification of the theory developed in Section 3 using Ornstein-Uhlenbeck bridges, resulting in sampling efficiencies for the particular (common) setting in which the fusion density is believed to be approximately Gaussian. In Section 5 we consider examples of our methodology applied to both light-tailed and heavy-tailed fusion target densities. Finally, in Section 6 we conclude by discussing the exciting new research directions possible using Monte Carlo Fusion. Much of the technical detail in the paper is suppressed for ease of reading, but can be found in the appendices.

2. An extended fusion density

Consider $f(\boldsymbol{x})$ and $f_c(\boldsymbol{x})$ as described in (1), where $f(\boldsymbol{x})$ is integrable and we can sample from the density proportional to $f_c(\boldsymbol{x})$.

The following simple observation will form the foundation of our approach: Suppose that $p_c(\boldsymbol{y} \,|\, \boldsymbol{x})$ is the transition density (with respect to Lebesgue measure) of a Markov chain on $\mathbf{R}^d$ with invariant density proportional to f_c^2.

Proposition 1. *The $(C+1)d$-dimensional (fusion) density proportional to the integrable function*

$$g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}) = \prod_{c=1}^{C} \left[f_c^2\Big(\boldsymbol{x}^{(c)}\Big) \cdot p_c\Big(\boldsymbol{y} \,\Big|\, \boldsymbol{x}^{(c)}\Big) \cdot \frac{1}{f_c(\boldsymbol{y})} \right], \tag{2}$$

admits the marginal density f for $\boldsymbol{y}$.

The proof of Proposition 1 is elementary and is omitted. The statistical interpretation of Proposition 1 is simply if it were possible to sample from the $(C+1)d$-dimensional (fusion) density g in (2), then as a by-product we would obtain a draw from our fusion target density f in (1). How to directly sample from (2) is not clear, even if it were possible to simulate from $p_c(\cdot|\boldsymbol{x})$. Our strategy instead will be to use rejection sampling. Two rejection sampling methods (which have differing efficiencies) are provided in Sections 3 and 4.

3. A fusion rejection sampler using Brownian bridges

3.1. The methodology

Consider the proposal density for the extended fusion target (2) proportional to the function

$$h^{bm}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}) = \prod_{c=1}^{C} \Big[f_c\Big(\boldsymbol{x}^{(c)}\Big) \Big] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T} \right), \tag{3}$$

where $\bar{\boldsymbol{x}} = C^{-1} \sum_{c=1}^{C} \boldsymbol{x}^{(c)}$, and T is an arbitrary positive constant.

Simulation from the proposal h^{bm} can be achieved directly. In particular, $\boldsymbol{x}^{(1)}, \ldots \boldsymbol{x}^{(C)}$ are first drawn independently from $f_1, \ldots f_C$ respectively, and then $\boldsymbol{y}$ is simply a Gaussian random variable centred on $\bar{\boldsymbol{x}}$.

In Proposition 1 we presented a general form of the extended fusion target, in which $p_c(\boldsymbol{y} \,|\, \boldsymbol{x})$ is the transition density (with respect to Lebesgue measure) of a Markov chain on $\mathbf{R}^d$ with invariant density proportional to f_c^2. In this paper we set $p_c(\boldsymbol{y} \,|\, \boldsymbol{x}) := p_{T,c}^{\mathrm{dl}}(\boldsymbol{y} \,|\, \boldsymbol{x})$, the transition density of a double Langevin diffusion for f_c (i.e. the transition density of a Langevin diffusion for f_c^2) from $\boldsymbol{x}$ to $\boldsymbol{y}$ over a pre-defined (user-specified) time $T > 0$. To distinguish the resulting extended fusion target from the general case, we further denote the extended fusion target by g^{dl}. In particular, for all $1 \leq c \leq C$, we consider the d-dimensional (double) Langevin (DL) diffusion processes $\mathfrak{X} = \{\boldsymbol{X}_t^{(c)}, t \in [0,T], c = 1, \cdots, C\}$, given by

$$\mathrm{d}\boldsymbol{X}_t^{(c)} = \nabla A_c^{\mathrm{dl}}\left(\boldsymbol{X}_t^{(c)}\right) \mathrm{d}t + \mathrm{d}\boldsymbol{W}_t^{(c)}, \tag{4}$$

where $\boldsymbol{W}_t^{(c)}$ is a d-dimensional Brownian motion, ∇ is the gradient operator over $\boldsymbol{x}$ and

$$A_c^{\mathrm{dl}}(\boldsymbol{x}) := \log f_c(\boldsymbol{x}). \tag{5}$$

$\boldsymbol{X}_t^{(c)}$ has invariant distribution $f_c^2(\boldsymbol{x})$, *for any* $t \in [0,T]$ (Hansen, 2003). We also impose the following standard regularity property (where **div** denotes the divergence operator)

Condition 1. *Define*

$$\phi_c^{dl}(\boldsymbol{x}) := \frac{1}{2}(\|\nabla A_c^{dl}(\boldsymbol{x})\|^2 + \mathbf{div}\ \nabla A_c^{dl}(\boldsymbol{x})). \tag{6}$$

There exists constant $\Phi_c^{bm} > -\infty$ *such that for all* $\boldsymbol{x}$ *and each* $c \in \{1, \cdots, C\}$, $\phi_c^{dl}(\boldsymbol{x}) \geq \Phi_c^{bm}$.

Then we have the following proposition which gives a rejection sampling method for $g^{dl}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})$.

Proposition 2. *Under Condition 1 we can write*

$$\frac{g^{dl}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h^{bm}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} = \left[\frac{\sqrt{C}}{\sqrt{2\pi T}}\right]^C \times \rho^{bm} \times Q^{bm} \times \prod_{c=1}^{C} e^{-T\Phi_c^{bm}}, \tag{7}$$

where

$$\rho^{bm} := \rho^{bm}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}) = e^{-\frac{C\sigma^2}{2T}}, \qquad \sigma^2 = C^{-1}\sum_{c=1}^{C} \|\boldsymbol{x}^{(c)} - \bar{\boldsymbol{x}}\|^2, \tag{8}$$

and

$$Q^{bm} = \mathbf{E}_{\overline{\mathbb{W}}}\left(E^{bm}\right), \tag{9}$$

with $\overline{\overline{\mathbb{W}}}$ *denoting the law of* C *Brownian bridges* $\boldsymbol{x}_t^{(1)}, \cdots, \boldsymbol{x}_t^{(C)}$ *with* $\boldsymbol{x}_0^{(c)} = \boldsymbol{x}^{(c)}$ *and* $\boldsymbol{x}_T^{(c)} = \boldsymbol{y}$ *in the time interval* $[0, T]$ *(noting that these Brownian bridges are independent conditional on the starting and ending points) and*

$$E^{bm} := \prod_{c=1}^{C} \left[\exp\left\{ -\int_0^T \left(\phi_c^{dl}\left(\boldsymbol{x}_t^{(c)}\right) - \Phi_c^{bm} \right) \mathrm{d}t \right\} \right]. \tag{10}$$

Proof. From the Dacunha-Castelle representation (Dacunha-Castelle and Florens-Zmirou, 1986) we have that

$$p_{T,c}^{\mathrm{dl}}\left(\boldsymbol{y} \,\middle|\, \boldsymbol{x}^{(c)}\right) = \frac{f_c(\boldsymbol{y})}{f_c(\boldsymbol{x}^{(c)})} \times \frac{\sqrt{C}}{\sqrt{2\pi T}} \exp\left(\frac{-\|\boldsymbol{y} - \boldsymbol{x}_c\|^2}{2T} \right) \times \mathbf{E}_{\overline{\mathbb{W}}} \left[\exp\left\{ -\int_0^T \phi_c^{\mathrm{dl}}\left(\boldsymbol{x}_t^{(c)}\right) \mathrm{d}t \right\} \right]. \tag{11}$$

The result then follows from (2) and (3) by rearrangement and recalling that $\sum_{c=1}^{C} \|\boldsymbol{y} - \boldsymbol{x}^{(c)}\|^2 = \sum_{c=1}^{C} \|\boldsymbol{x}^{(c)} - \bar{\boldsymbol{x}}\|^2 + C||\boldsymbol{y} - \bar{\boldsymbol{x}}||^2$. □

Here ρ^{bm} and Q^{bm} are both necessarily bounded by 1, and when interpreted methodologically (see next section) correspond to separate acceptance steps within our rejection sampling framework. An event of probability ρ^{bm} can be simulated by direct computation, and an event of probability Q^{bm} can be simulated using the extensive efficient methodology on Poisson samplers (using an auxiliary diffusion bridge path-space rejection sampler as developed in (for instance) Beskos et al. (2006a,b, 2008), Chen and Huang (2013), Dai (2014), Pollock (2013), Pollock et al. (2016a) and Pollock et al. (2016b)). Note that there is a trade-off involved in the (user-specified) choice of T. For small T, ρ^{bm} will likely be small while Q^{bm} is large, whereas for large T the opposite will be true. A small value of T is usually preferred since the computational cost for the diffusion bridge rejection sampling for Q^{bm} is comparatively expensive.

The algorithm for simulating from f (by means of g) therefore proceeds as per Algorithm 1 (which we term *Monte Carlo Fusion*).

3.2. Practical interpretation of the algorithm

Remark 1. Adjustment for the simple average of the sample from sub-densities.

In the above algorithm, for all c, $\boldsymbol{x}^{(c)}$ is simulated from $f_c(\boldsymbol{x})$ as in other Monte Carlo Fusion algorithms. The proposed combined value $\boldsymbol{y}$, however, is actually generated from a Gaussian distribution with mean $\bar{\boldsymbol{x}}$ and covariance matrix $C^{-1}T\mathbf{I}_d$. Therefore, $\boldsymbol{y}$ can be viewed as a simple

1 Initialize a value $T > 0$;

2 For $c = 1, \cdots, C$, simulate $\boldsymbol{x}_c$ from the density $f_c(\boldsymbol{x})$ and calculate $\bar{\boldsymbol{x}}$;

3 Simulate $\boldsymbol{y}$ from the Gaussian distribution, with density $\exp\left(-\frac{C \cdot \|\boldsymbol{y}-\bar{\boldsymbol{x}}\|^2}{2T}\right)$;

4 Generate standard uniform random variable U_1;

5 **if** $\log U_1 \leq -\frac{C\sigma^2}{2T}$ **then**

6 Generate standard uniform variable U_2 and the independent Brownian bridges $\boldsymbol{x}_t^{(c)}, c = 1, \cdots, C$ in $[0, T]$, conditional on the starting point $\boldsymbol{x}_c$ and ending point $\boldsymbol{y}$;

7 **if**

$$U_2 \leq E^{bm} \tag{12}$$

then

8 Accept and output $\boldsymbol{y}$ as a sample from $f(\boldsymbol{x})$;

`// The event (12) can be dealt with via the path-space rejection sampling methods in Beskos et al. (2006a, 2008); Beskos and Roberts (2005); Pollock et al. (2016a)`

9 **else**

10 Go back to step 2;

11 **end**

12 **else**

13 Go back to step 2;

14 **end**

Algorithm 1: Monte Carlo Fusion (Brownian Bridge Approach)

average of the values $\{\boldsymbol{x}^{(c)},\ c \in C\}$ added to a Gaussian random error term. This algorithm indicates exactly how the simple average of these independent sub-posterior samples can be adjusted as a draw from the target distribution. This adjustment is in the form of the accept/reject step with acceptance probability $\rho^{bm} \times Q^{bm}$. □

Remark 2. Implication on Bayesian group decision theory.

In Step 5 of Algorithm 1, we can also write $\log U_1 \leq -\frac{C\sigma^2}{2T}$ as

$$\sigma^2 \leq -\frac{2T \log U_1}{C}. \tag{13}$$

Note that σ^2 is actually the *sample variance* of the simulated starting points $\boldsymbol{x}^{(c)}$s (strictly speaking, it is the sum of variances for each component of $\boldsymbol{x}^{(c)}$). Thus in this step, the acceptance condition (13) implies that $\boldsymbol{y}$ will have a reasonable probability of being accepted as a sample from $f(\boldsymbol{x})$

only when the variance of $\boldsymbol{x}^{(c)}$s is small enough. This coincides with our intuition in group decision making. For example, the small variance of $\{\boldsymbol{x}^{(c)},\ c \in C\}$ (satisfying condition (13)) means that the decisions/results from each group are similar, so that we can combine these decisions/results in step 7, using (12), in Algorithm 1. On the other hand, if the variance of $\{\boldsymbol{x}^{(c)},\ c \in C\}$ (not satisfying condition (13)) is large, then $\{\boldsymbol{x}^{(c)},\ c \in C\}$ provide contradictory evidence and should usually be rejected. Note that algorithmically, this initial rejection sampling step is efficient since early rejection then avoids the need to carry out the more complicated (and computationally expensive) step 7, via the condition of (12). □

Remark 3. An extreme case.

A very interesting extreme case is to choose $T = 0$. Then the event (12) will certainly happen, but the event $U_1 \leq \exp\left(-\frac{C\sigma^2}{2T}\right)$ will occur only if $\boldsymbol{x}^{(1)} = \cdots = \boldsymbol{x}^{(C)} = \boldsymbol{y}$. Therefore in such an extreme case, Algorithm 1 (theoretically) draws a sample $\boldsymbol{y}$ from

$$h^{\mathrm{bm}}\left(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \ \propto \ \prod_{c=1}^{C} f_c(\boldsymbol{y})$$

which is the target distribution. In practice, however, we have to choose $T > 0$, since the independent sub-posterior samples $\boldsymbol{x}^{(c)}$s have zero probability to be the same. □

4. A fusion rejection sampler using Ornstein-Uhlenbeck bridges

4.1. The methodology

In the previous section, the proposal density h^{bm} uses a simple average of the sub-posterior samples $\boldsymbol{x}^{(c)}$ as the mean of the proposal $\boldsymbol{y}$. Another approach is to consider using a weighted average of the sub-posterior samples $\boldsymbol{x}^{(c)}$ as the mean of the proposal $\boldsymbol{y}$. For example, in a typical meta-analysis, a weighted average from different research outputs is typically used as the unification mean, and an individual output with more certainty (or, smaller variance) should consequently have larger weights (Fleiss, 1993).

Denoting $\hat{\boldsymbol{\mu}}_c$ and $\hat{\boldsymbol{\Lambda}}_c$ as the mean and the inverse of covariance matrix estimates for distribution $f_c(\boldsymbol{x})$, respectively. We consider the following more general proposal density

$$h^{\mathrm{ou}}\left(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \left[\prod_{c=1}^{C} f_c\left(\boldsymbol{x}^{(c)}\right)\right] \mathbf{etr}\left[-\frac{1}{2}\left[\boldsymbol{y} - \widetilde{\boldsymbol{x}}\right]^{\otimes 2} \mathbf{D}\right] \tag{14}$$

where

$$\begin{aligned}\mathbf{D} &= \sum_{c=1}^{C}\mathbf{D}_c,\ \ \mathbf{D}_c = \mathbf{V}_c^{-1} - \hat{\mathbf{\Lambda}}_c \\ \mathbf{V}_c &:= \mathbf{V}_c(T) = \mathbf{Var}\left(\int_0^T e^{\hat{\mathbf{\Lambda}}_c(t-T)}d\boldsymbol{W}_t^{(c)}\right) = \frac{\hat{\mathbf{\Lambda}}_c^{-1}}{2}\left(\mathbf{I}_d - e^{-2\hat{\mathbf{\Lambda}}_c T}\right)\end{aligned} \tag{15}$$

and

$$\begin{aligned}\widetilde{\boldsymbol{x}} &= \mathbf{D}^{-1}\left\{\sum_{c=1}^{C}\left(\mathbf{V}_c^{-1}\boldsymbol{m}_c - \hat{\mathbf{\Lambda}}_c\hat{\boldsymbol{\mu}}_c\right)\right\} \\ \boldsymbol{m}_c &:= \boldsymbol{m}_c(\boldsymbol{x}^{(c)}, T) = \hat{\boldsymbol{\mu}}_c + e^{-\hat{\mathbf{\Lambda}}_c T}(\boldsymbol{x}^{(c)} - \hat{\boldsymbol{\mu}}_c).\end{aligned} \tag{16}$$

It is also straightforward to simulate from $h^{\mathrm{ou}}(\cdot)$, since $\boldsymbol{x}^{(c)}$s are independently drawn from $f_c(\boldsymbol{x})$ and then $\boldsymbol{y}$ is generated from a Gaussian distribution with mean $\tilde{\boldsymbol{x}}$ and covariance matrix $\mathbf{D}^{-1}$.

Here the vector $\boldsymbol{m}_c$ and the matrix $\mathbf{V}_c$ are actually the mean and covariance matrix, respectively, for the following OU process at time T conditional on the starting point $\boldsymbol{x}_0^{(c)} = \boldsymbol{x}^{(c)}$ (Masuda, 2004),

$$\mathrm{d}\boldsymbol{X}_t^{(c)} = \nabla A_c^{\mathrm{ou}}\left(\boldsymbol{X}_t^{(c)}\right)\mathrm{d}t + \mathrm{d}\boldsymbol{W}_t^{(c)},\ \ \boldsymbol{X}_0^{(c)} \sim f_c^2(\boldsymbol{x}) \tag{17}$$

with

$$A_c^{\mathrm{ou}}(\boldsymbol{x}) = -\frac{(\hat{\boldsymbol{\mu}}_c - \boldsymbol{x})^{tr}\hat{\mathbf{\Lambda}}_c(\hat{\boldsymbol{\mu}}_c - \boldsymbol{x})}{2}. \tag{18}$$

We shall also require the following regularity property.

Condition 2. *Define*

$$\phi_c^{ou}(\boldsymbol{x}) = \frac{1}{2}\left(\|\hat{\mathbf{\Lambda}}_c(\hat{\boldsymbol{\mu}}_c - \boldsymbol{x})\|^2 - \ \mathbf{trace}\ (\hat{\mathbf{\Lambda}}_c)\right). \tag{19}$$

For any $c \in \{1, \cdots, C\}$, there exists $\Phi_c^{ou} > -\infty$ such that, for all $\boldsymbol{x}$,

$$\phi_c^{dl}(\boldsymbol{x}) - \phi_c^{ou}(\boldsymbol{x}) \geq \Phi_c^{ou}. \tag{20}$$

We now have the following result.

Proposition 3. *Define function*

$$\begin{aligned}\rho^{ou} &:= \rho^{ou}(\boldsymbol{x}^{(1)},\cdots,\boldsymbol{x}^{(C)}) \\ &= \mathbf{etr}\left\{-\frac{1}{2}\left[\mathbf{H}\mathbf{D}^{-1}+\sum_{c=1}^{C}\mathbf{M}_{1,c}\left(\boldsymbol{m}_c+\mathbf{M}_{1,c}^{-1}\mathbf{M}_{2,c}\mathbf{V}_c\hat{\mathbf{\Lambda}}_c\hat{\boldsymbol{\mu}}_c\right)^{\otimes 2}\right]\right\}\end{aligned} \tag{21}$$

with

$$\begin{aligned}\mathbf{M}_{1,c} &= e^{2\hat{\mathbf{\Lambda}}_c T}\hat{\mathbf{\Lambda}}_c-\mathbf{V}_c^{-1}\left(\sum_{c=1}^{C}\hat{\mathbf{\Lambda}}_c\right)\mathbf{D}^{-1} \\ \mathbf{M}_{2,c} &= \mathbf{V}_c^{-1}\left(\sum_{c=1}^{C}\hat{\mathbf{\Lambda}}_c\right)\mathbf{D}^{-1}-2\hat{\mathbf{\Lambda}}_c e^{2\hat{\mathbf{\Lambda}}_c T} \\ \mathbf{H} &= \left(\sum_{c=1}^{C}\left(\boldsymbol{m}_c-\mathbf{V}_c\hat{\mathbf{\Lambda}}_c\hat{\boldsymbol{\mu}}_c\right)^{\otimes 2}\mathbf{V}_c^{-1}\right)\left(\sum_{c=1}^{C}\mathbf{V}_c^{-1}\right)-\left\{\sum_{c=1}^{C}\mathbf{V}_c^{-1}\left(\boldsymbol{m}_c-\mathbf{V}_c\hat{\mathbf{\Lambda}}_c\hat{\boldsymbol{\mu}}_c\right)\right\}^{\otimes 2}.\end{aligned}$$

Under Condition 2 we can write

$$\frac{g^{dl}(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(C)},\boldsymbol{y})}{h^{ou}\big(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(C)},\boldsymbol{y}\big)} \propto \rho^{ou}(\boldsymbol{x}^{(1)},\cdots,\boldsymbol{x}^{(C)})\times Q^{ou}\times\prod_{c=1}^{C}e^{-T\Phi_c^{ou}} \tag{22}$$

where

$$Q^{ou}=\mathbf{E}_{\overline{\mathbb{O}}}(E^{ou}) \tag{23}$$

with $\overline{\mathbb{O}}$ *denoting the law of* C *OU bridges* $\boldsymbol{x}_t^{(c)}, c=1,\cdots,C$ *in time interval* $[0,T]$. *These* $\boldsymbol{x}_t^{(c)}$*s are independent conditional on the starting point* $\boldsymbol{x}^{(c)}$ *and common ending point* $\boldsymbol{y}$ *and*

$$E^{ou}:=\prod_{c=1}^{C}\left[\exp\left\{-\int_0^T\left(\phi_c^{dl}\Big(\boldsymbol{x}_t^{(c)}\Big)-\phi_c^{ou}\Big(\boldsymbol{x}_t^{(c)}\Big)-\Phi_c^{ou}\right)\mathrm{d}t\right\}\right]. \tag{24}$$

Proof. See Appendix. □

Since ρ^{ou} and Q^{ou} in (22) are always no more than 1, we have the following rejection sampling algorithm, Algorithm 2.

In Algorithm 2, T is a tuning parameter as well. If we choose a small value of T, the acceptance probability $\rho^{ou}(\boldsymbol{x}^{(1)},\cdots,\boldsymbol{x}^{(C)})$ will be very low. This is noticed by the fact that $\mathbf{trace}\,(\mathbf{H}\mathbf{D}^{-1})=\infty$ and $\mathbf{M}_{1,c}$ and $\mathbf{M}_{2,c}$ are finite matrices, if $T=0$. Therefore in practice, we should choose a reasonably large value T. However, if we choose a large T, the acceptance probability (25) will be very small. In practice it is usually preferable to choose a small T since the computational cost for the acceptance / rejection step of (25) is much higher.

1 Initialise a value $T > 0$ and $\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\Lambda}}_c$;

2 For $c = 1, \cdots, C$, simulate $\boldsymbol{x}^{(c)}$ from the density $f_c(\boldsymbol{x})$ and calculate $\widetilde{\boldsymbol{x}}, \mathbf{D}$;

3 Simulate $\boldsymbol{y}$ from the Gaussian distribution, with mean $\widetilde{\boldsymbol{x}}$ and covariance matrix $\mathbf{D}^{-1}$;

4 Generate standard uniform random variable U_1;

5 **if** $U_1 \leq \rho^{ou}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)})$ *(given in the formula (21))* **then**

6 Generate standard uniform random variable U_2 and independent OU bridges $\boldsymbol{x}_t^{(c)}, c = 1, \cdots, C$ in $[0, T]$, conditional on the starting point $\boldsymbol{x}_0^{(c)} = \boldsymbol{x}^{(c)}$ and ending point $\boldsymbol{y}$;

7 **if**

$$U_2 \leq \exp\left[-\sum_{c=1}^{C} \int_0^T \left(\phi_c^{dl}\left(\boldsymbol{x}_t^{(c)}\right) - \phi_c^{ou}\left(\boldsymbol{x}_t^{(c)}\right) - \Phi_c^{ou}\right) \mathrm{d}t\right] \tag{25}$$

`// The event (25) can be dealt with via the path-space rejection sampling methods in Beskos et al. (2006a, 2008); Beskos and Roberts (2005); Pollock et al. (2016a)`

8 **then**

9 Accept and output $\boldsymbol{y}$ as a sample from $f(\boldsymbol{x})$;

10 **else**

11 go back to step 2

12 **end**

13 **else**

14 go back to step 2

15 **end**

Algorithm 2: Monte Carlo Fusion (Ornstein-Uhlenbeck Approach)

4.2. Connection with Consensus Monte Carlo

In practice, people may employ an approximate version of Algorithm 2, since we can ignore the condition (25) in the rejection step 7, if we choose a small value T. In other words, from (22) of Proposition 3, we have that for small T the target $g^{dl}(\cdot)$ is approximately equal to a function proportional to

$$\tilde{h}^{ou} = h^{ou} \cdot \rho^{ou} = \left[\prod_{c=1}^{C} f_c\left(\boldsymbol{x}^{(c)}\right)\right] \mathbf{etr}\left[-\frac{1}{2}\left[\boldsymbol{y} - \widetilde{\boldsymbol{x}}\right]^{\otimes 2} \mathbf{D}\right] \cdot \rho^{ou}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}) \tag{26}$$

Such an approximation will be very good since Q^{ou}, the acceptance probability (25), will be very close to 1 with a small T. In other words, we only simulate $\boldsymbol{y}$ from $\tilde{h}^{ou}(\cdot)$ and accept $\boldsymbol{y}$ as a sample approximately from the target distribution $f(\cdot)$.

If we draw $\boldsymbol{x}_c, c = 1, \cdots, C$ independently from each $f_c(\boldsymbol{x})$, respectively. Another naive approach to combine these draws is to use the following linear combination

$$\boldsymbol{y} = \left(\sum_{c=1}^{C} \hat{\boldsymbol{\Lambda}}_c \right)^{-1} \left[\sum_{c=1}^{C} \hat{\boldsymbol{\Lambda}}_c \boldsymbol{x}_c \right]. \tag{27}$$

In practice, $\hat{\boldsymbol{\Lambda}}_c^{-1}$ can be obtained via preliminary analysis. Such a $\boldsymbol{y}$ can be viewed as a sample approximately from $f(\boldsymbol{x})$ as well. This is named as Consensus Monte Carlo (CMC) in (Scott et al., 2016).

The following lemma tells us how the simulation of $\boldsymbol{y}$ from $\tilde{h}^{ou}(\cdot)$ is related to the consensus Monte Carlo sample (27).

Lemma 1. *If $f_c(\boldsymbol{x})$ is a Gaussian distribution and if we choose $T = \infty$, then simulating $\boldsymbol{y}$ from $\tilde{h}^{ou}(\cdot)$ will be the same as the Consensus Monte Carlo method. Both draw samples exactly from the target distribution.*

Proof. See Appendix B. □

This lemma tells us why Consensus Monte Carlo does not provide good results. It is because CMC simulates $\boldsymbol{y}$ from $\tilde{h}^{ou}(\cdot)$ with $T = \infty$, however, T should be chosen as a small value to achieve better approximation or even use the exact Algorithm 2 with the diffusion path rejection sampler.

5. Simulation studies

5.1. Distribution with light tails

We consider the target distribution $f(x) \propto e^{-x^4/2}$. We choose $C = 4$ and $f_c(x) = e^{-x^4/2C}, c = 1, \cdots, C$. It is easy to check that $\phi^{dl}(x) = \frac{1}{2}\left(\frac{4x^6}{C^2} - \frac{6x^2}{C}\right)$ satisfies Condition 1. On the other hand, for any chosen values $\hat{\boldsymbol{\mu}}_c$, $\hat{\boldsymbol{\Lambda}}_c$, $\phi_c^{\mathrm{ou}}(\boldsymbol{x})$ in (19) satisfies Condition 2 as well. Therefore both Algorithm 1 and Algorithm 2 can be applied to simulate from $f(x)$.

We compare the following Monte Carlo methods for the estimation of the density function of $f(x)$:

1. Simulating MC samples directly from $f(x)$ via a simple rejection sampling with standard Gaussian distribution as the proposal;

2. simulating MC samples based on the exact simulation method, Algorithm 1 with $T = 1$;
3. simulating MC samples based on the exact simulation method, Algorithm 2 with $T = 1$;
4. simulating MC samples based on the Consensus method in Scott et al. (2016);

The density curve estimation results are summarised in Figure 1. Note that all results are based on $10,000$ realisations. The black solid cure (Simulation [1.], the true fitted density curve), the blue solid curve (Simulation [2.]) and the pink solid curve (Simulation [3.]) are all exact algorithms and they are almost identical. The Consensus methods (Simulation [4.] – the red dashed curve, has very large biases. Note that both CMC algorithm and Algorithm 2 use the same values of $\hat{\boldsymbol{\mu}}_c$ and $\hat{\boldsymbol{\Lambda}}_c$ based on preliminary analysis.

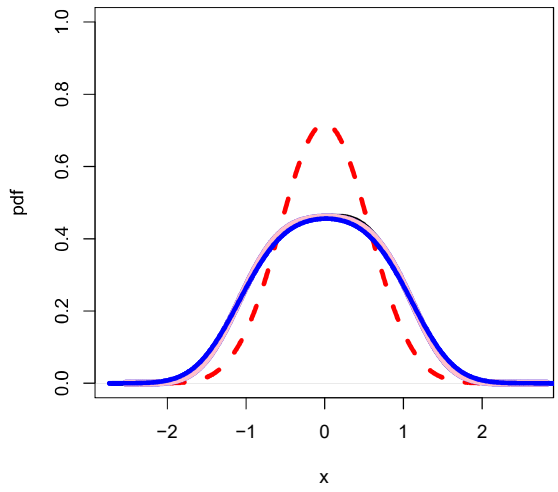

FIGURE 1: Kernel density fitting with bandwidth 0.25 for density proportional to $e^{-x^4/2}$, based on different Monte Carlo methods. [1.]– black solid curve, standard exact MC; [2.]– pink solid curve, Algorithm 1; [3.]– blue solid curve, Algorithm 2; [4.]– red dashed curve, CMC algorithm.

The running time of the algorithms are presented in Table 1. It seems that Algorithm 2 uses the most system running time. This is because Algorithm 2 has a smaller acceptance probability (about 0.011 for the path-space rejection sampling (25)) than that in Algorithm 1 (about 0.139 for the path-space rejection sampling (12)).

Note that Condition 1 is usually satisfied in most applications, however, Condition 2 only holds

Algorithms	CMC	Algorithm 1	Algorithm 2
Running time	0.05	0.36	4.05

TABLE 1: System running times in seconds for simulating 10,000 realisations.

when the target $f(x)$ has lighter tails than Gaussian distributions. We present an example in the following section, where only Condition 1 is true.

5.2. Beta distribution

Consider the target distribution as the Beta distribution with density $\pi(u) \propto u^4(1-u)$, $u \in [0,1]$, i.e. Beta$(5,2)$. To use the proposed algorithms, the support of the target distribution should be in the whole real axis. Therefore we need to use the variable transformation $x = \log(u/(1-u))$ and consider the target distribution as

$$f(x) \propto \left[\frac{\exp(x)}{1+\exp(x)}\right]^5 \left[\frac{1}{1+\exp(x)}\right]^2.$$

We decompose $\pi(x)$ into $C = 5$ components,

$$\begin{aligned} f(x) &\propto f_1(x)\cdots f_C(x) \\ f_c(x) &= \left[\frac{\exp(x)}{1+\exp(x)}\right]\left[\frac{1}{1+\exp(x)}\right]^{0.4}. \end{aligned} \tag{28}$$

Note that for this simple example Condition 1 is satisfied but Condition 2 is not satisfied. Therefore we compare the following Monte Carlo methods for the estimation of the density function of Beta$(5,2)$:

1. Simulating MC samples directly from Beta$(5,2)$ via the simple R command, *rbeta*;
2. simulating MC samples based on the exact simulation method, Algorithm 1 with $T = 3$;
3. simulating MC samples based on the Consensus method in Scott et al. (2016), with variable transformation and with the decomposition in (28).

For simulations [3.], $\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\Lambda}}_c$ (also known as the weight of each consensus sample in CMC algorithm) are chosen, respectively, as the estimated mean and inverse of variance of $f_c(\cdot)$, as suggested by Scott et al. (2016).

The density curve estimation results are summarized in Figure 2. Note that all results are based on $10,000$ realisations. The black solid cure (Simulation [1.]) and the blue solid curve (Simulation [2.])

are almost identical, since both of them are based on exact simulation methods. Again, Concensus Monte Carlo gives very biased results.

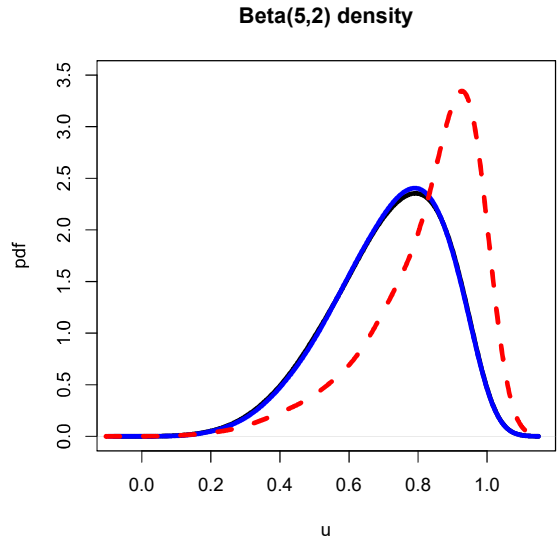

FIGURE 2: Kernel density fitting with bandwidth 0.04 for Beta$(5, 5)$, based on different Monte Carlo methods. [1.]– black solid curve, standard exact MC; [2.]– blue solid curve, Algorithm 1; [3.]– red dashed curve, CMC algorithm.

Remark 4. Tuning parameters and density decomposition

We may choose any value T in the proposed algorithms. However, as we mentioned before, value T is a tunning parameter for the efficiency of Algorithm 1, which indeed shown by our simulation results. The CPU running time for simulating 10,000 realisations based on Algorithm 1 for the Beta-example is summarized in Figure 3. The optimum value is about $T = 2$. Note that such optimum value can be found in practice via a small amount of preliminary simulations.

In addition, when we split the target f into $f \propto f_1 \cdots f_C$, we actually chose $f_1 = \cdots = f_C = f^{1/C}$, since such decomposition will always give the smallest variation for $\boldsymbol{x}^{(c)}, c = 1, \cdots, C$, as suggested in Dai (2017).

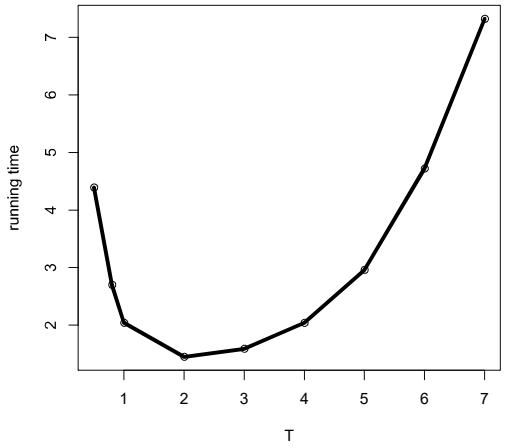

FIGURE 3: CPU time (in seconds) for Algorithm 1, based on different T.

6. Conclusion

In this paper we have introduced a novel theoretical framework, and direct methodological implementation (Monte Carlo Fusion), to address the common (but challenging) 'fusion' problem – unifying distributed analyses and inferences on shared parameters from multiple sources, into a single coherent inference – by viewing it as a simple rejection sampler on an extended space (Section 2), and developing an appropriate sampling mechanism (Section 3).

Our fusion approach is not only the first paper to answer in a principled manner how to combine samples from multiple sources, but (as shown in Section 4), also provides a principled approach to understand the errors that arise in other existing unification schemes (such as those in big data divide-and-conquer approaches). The errors in existing unification schemes can be considerable, even for simple one dimensional unification targets (such as those we consider in Section 5).

Characterising the error in existing unification schemes is possible by setting the (proposal) sampling mechanisms of those schemes within our framework, and finding a representation for the remaining error (which we could remove by further acceptance or rejection). This opens interesting avenues of research in which existing unification schemes are adapted within our framework into efficient proposal mechanisms for our extended fusion target density (2).

A number of avenues to apply our work directly to interesting applications are possible. In addition

to those discussed in Section 1 (namely expert elicitation, multi-view learning and meta-analysis), other application areas include Bayesian group decision theory (see Remark 2) and Bayesian sensitivity analysis. In the case of Bayesian sensitivity analysis we need to assess a large number of prior distributions, but existing methods address this by using approximations (Tan et al., 2015).

A key avenue for (on-going) future research is to fully explore how to use the Monte Carlo Fusion framework we introduce to modify, and remove approximation, from existing Monte Carlo methods (particularly within the big data setting) that use the divide-and-conquer (fork-and-join) strategies described in Section 1. In the particular setting of big data the unification of distributed inferences is only part of the problem – additional constraints are imposed due to practical computational and hardware concerns (such as avoiding as far as possible communication between computing cores). As such, key future research will focus on how to implement Monte Carlo Fusion with this particular set of constraints (as direct implementation is not possible).

Another interesting big data direction would be to blend the multi-core approach of divide-and-conquer strategies, with state-of-the-art single-core approaches for big data (for instance, Pollock et al. (2016a)).

Acknowledgements

The authors would like to thank the Isaac Newton Institute for Mathematical Sciences for support and hospitality during the programme "Scalable inference; statistical, algorithmic, computational aspects (SIN)" when work on this paper was undertaken. MP and GOR were supported by the EPSRC [grant number EP/K014463/1]. GOR was additionally supported by the EPSRC [grant numbers EP/K034154/1, EP/D002060/1].

Appendix A. Proof of Proposition 3

To prove the proposition, we first introduce a lemma about a density proportional to the following function

$$\begin{aligned} & \tilde{g}^{ou}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}) \\ = & \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right) p_{T,c}^{\mathrm{ou}}\left(\boldsymbol{y} \,\middle|\, \boldsymbol{x}^{(c)}\right) \exp\left(A_c^{\mathrm{ou}}\left(\boldsymbol{x}^{(c)}\right) - A_c^{\mathrm{ou}}\left(\boldsymbol{y}\right)\right)\right] \end{aligned} \tag{29}$$

where $p^{\text{ou}}_{T,c}\left(\boldsymbol{y} \,\middle|\, \boldsymbol{x}^{(c)}\right)$ is the transition density from $\boldsymbol{x}^{(c)}$ at time 0 to $\boldsymbol{y}$ at time T, for the OU process given by (17).

Lemma 2. The expression of $\tilde{g}^{ou}$. *The formula in (29) can be rewritten as*

$$\begin{aligned} & \tilde{g}^{ou}\left(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \\ \propto \quad & \left[\prod_{c=1}^{C} f_c\left(\boldsymbol{x}^{(c)}\right)\right] \mathbf{etr}\left[-\frac{1}{2}\left[\boldsymbol{y}_T - \widetilde{\boldsymbol{x}}\right]^{\otimes 2} \mathbf{D}\right] \rho^{ou}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}) \end{aligned} \tag{30}$$

where $\widetilde{\boldsymbol{x}}$ and ρ^{ou} are given by (16) and (21), respectively. □

Proof. See the supplementary file. □

Now we prove Proposition 3.

Proof. If we denote $\overline{\mathbb{DL}}$ as the law of C d-dimensional Langevin diffusion bridges given in (4), with starting points $\boldsymbol{x}^{(c)}$ and common ending point $\boldsymbol{y}$, the result of Proposition 2 can be written as

$$\frac{g^{dl}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h^{bm}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \times \mathrm{d}\overline{\mathbb{DL}}(\vec{\boldsymbol{x}}) \propto \rho^{bm} \times E^{bm} \times \prod_{c=1}^{C} e^{-T\Phi_c} \times \mathrm{d}\overline{\mathbb{W}}(\vec{\boldsymbol{x}}) \tag{31}$$

where $\vec{\boldsymbol{x}} = \{\boldsymbol{x}_t^{(c)}, c = 1, \cdots, C, t \in [0, T]\}$ are typical diffusion bridge paths, with starting points $\boldsymbol{x}^{(c)}$ and common ending point $\boldsymbol{y}$.

If we consider a very special case where each $f_c(\boldsymbol{x})$ is a Gaussian density $\exp(A_c^{\text{ou}}(\boldsymbol{x}))$, we immediately have that the target g^{dl} becomes

$$g^{ou} = \prod_{c=1}^{C}\left[e^{2A_c^{\text{ou}}\left(\boldsymbol{x}^{(c)}\right)} p^{\text{ou}}_{T,c}\left(\boldsymbol{y} \,\middle|\, \boldsymbol{x}^{(c)}\right) \frac{1}{e^{A_c^{\text{ou}}(\boldsymbol{y})}}\right]$$

In addition, the proposal density h^{bm} in Propostion 2 becomes

$$\hbar^{bm}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}) = \prod_{c=1}^{C}\left[e^{A_c^{\text{ou}}\left(\boldsymbol{x}^{(c)}\right)}\right] e^{-\frac{\|\boldsymbol{y}-\bar{\boldsymbol{x}}\|^2}{2T}},$$

and further the rejection sampling ratio in Proposition 2 becomes

$$\frac{g^{ou}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{\hbar^{bm}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \times \mathrm{d}\overline{\mathbb{O}}(\vec{\boldsymbol{x}}) \propto \rho^{bm} \times E_*^{ou} \times \mathrm{d}\overline{\mathbb{W}}(\vec{\boldsymbol{x}}) \tag{32}$$

$$E_*^{ou} := \prod_{c=1}^{C}\left[\exp\left\{-\int_0^T \phi_c^{\text{ou}}\left(\boldsymbol{x}_t^{(c)}\right) \mathrm{d}t\right\}\right] . \tag{33}$$

Now we have,

$$\begin{aligned}
\frac{g^{dl}}{h^{ou}} \cdot \frac{\mathrm{d}\overline{\mathbb{DL}}}{\mathrm{d}\overline{\mathbb{O}}} &= \frac{g^{dl}}{h^{bm}} \frac{\mathrm{d}\overline{\mathbb{DL}}}{\mathrm{d}\overline{\mathbb{W}}} \cdot \frac{\hbar^{bm}}{g^{ou}} \frac{\mathrm{d}\overline{\mathbb{W}}}{\mathrm{d}\overline{\mathbb{O}}} \cdot \frac{g^{ou}}{h^{ou}} \cdot \frac{h^{bm}}{\hbar^{bm}} \\
&\propto \frac{E^{bm}}{E_*^{ou}} \cdot \frac{\prod_{c=1}^{C} \left[e^{2A_c^{\mathrm{ou}}(\boldsymbol{x}^{(c)})} p_{T,c}^{\mathrm{ou}}(\boldsymbol{y} \,|\, \boldsymbol{x}) \frac{1}{e^{A_c^{\mathrm{ou}}(\boldsymbol{y})}} \right]}{\left[\prod_{c=1}^{C} f_c(\boldsymbol{x}^{(c)}) \right] \mathbf{etr}\left[-\frac{1}{2} [\boldsymbol{y} - \widetilde{\boldsymbol{x}}]^{\otimes 2} \mathbf{D} \right]} \cdot \frac{\left[\prod_{c=1}^{C} f_c(\boldsymbol{x}^{(c)}) \right]}{\left[\prod_{c=1}^{C} e^{A_c^{\mathrm{ou}}(\boldsymbol{x}^{(c)})} \right]} \\
&= \frac{E^{bm}}{E_*^{ou}} \cdot \frac{\tilde{g}^{ou}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{\left[\prod_{c=1}^{C} f_c(\boldsymbol{x}^{(c)}) \right] \mathbf{etr}\left[-\frac{1}{2} [\boldsymbol{y} - \widetilde{\boldsymbol{x}}]^{\otimes 2} \mathbf{D} \right]}.
\end{aligned}$$

Lemma 2 and expressions of E^{bm} and E_*^{ou} immediately gives

$$\frac{g^{dl}}{h^{ou}} = \frac{E^{bm}}{E_*^{ou}} \cdot \rho^{ou} \propto E^{ou} \cdot \rho^{ou}$$

where E^{ou} given in (24). □

Appendix B. Proof of Lemma 1

We consider the formula of ρ^{ou} in (21). If $T = \infty$, we have

$$\boldsymbol{m}_c = \hat{\boldsymbol{\mu}}_c, \qquad \mathbf{V}_c = \frac{\hat{\boldsymbol{\Lambda}}_c^{-1}}{2}$$

and

$$\begin{aligned}
\mathbf{D} &= \sum_{c=1}^{C} \hat{\boldsymbol{\Lambda}}_c, \\
\mathbf{M}_{1,c} &= e^{2\hat{\boldsymbol{\Lambda}}_c T} \hat{\boldsymbol{\Lambda}}_c - 2\hat{\boldsymbol{\Lambda}}_c \\
\mathbf{M}_{2,c} &= 4\hat{\boldsymbol{\Lambda}}_c - 2e^{2\hat{\boldsymbol{\Lambda}}_c T} \hat{\boldsymbol{\Lambda}}_c = -2\mathbf{M}_{1,c}
\end{aligned}$$

Therefore

$$\begin{aligned}
& \lim_{T\to\infty} \mathbf{M}_{1,c} \left(\boldsymbol{m}_c + \mathbf{M}_{1,c}^{-1} \mathbf{M}_{2,c} \mathbf{V}_c \hat{\boldsymbol{\Lambda}}_c \hat{\boldsymbol{\mu}}_c \right)^{\otimes 2} \\
= & \lim_{T\to\infty} \left(\mathbf{M}_{1,c}^{1/2} \hat{\boldsymbol{\mu}}_c + \mathbf{M}_{1,c}^{-1/2} \mathbf{M}_{2,c} \frac{\hat{\boldsymbol{\mu}}_c}{2} \right)^{\otimes 2} \\
= & \mathbf{0}
\end{aligned}$$

and further $\rho^{ou}(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)})$ becomes a value not depending on $\boldsymbol{X}_0^{(1:C)}$ at all, as $T \to \infty$. Therefore with $T = \infty$ the density function $\tilde{h}^{ou}(\cdot)$ becomes

$$\tilde{h}^{ou}\cdot(\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(C)}, \boldsymbol{y}) \quad \propto \quad \left[\prod_{c=1}^{C} f_c\Big(\boldsymbol{x}^{(c)}\Big)\right] \mathbf{etr}\left[-\frac{1}{2}\left[\boldsymbol{y} - \widetilde{\boldsymbol{x}}\right]^{\otimes 2}\left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)\right]$$

with

$$\widetilde{\boldsymbol{x}} = \left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)^{-1}\left\{\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\hat{\boldsymbol{\mu}}_c\right\}$$

Then we can generate $\boldsymbol{y}$ as, with some standard Gaussian random errors $\boldsymbol{\epsilon}$,

$$\begin{aligned}
\boldsymbol{y} &= \widetilde{\boldsymbol{x}} + \left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)^{-1/2}\cdot\boldsymbol{\epsilon} \\
&= \left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)^{-1}\left\{\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\hat{\boldsymbol{\mu}}_c + \left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)^{1/2}\boldsymbol{\epsilon}\right\} \\
&\overset{distr.}{=} \left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)^{-1}\left\{\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\hat{\boldsymbol{\mu}}_c + \sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c^{1/2}\boldsymbol{\epsilon}_c\right\} \\
&= \left(\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\right)^{-1}\left\{\sum_{c=1}^{C}\hat{\boldsymbol{\Lambda}}_c\left(\hat{\boldsymbol{\mu}}_c + \hat{\boldsymbol{\Lambda}}_c^{-1/2}\boldsymbol{\epsilon}_c\right)\right\}
\end{aligned}$$

where $\overset{distr.}{=}$ means 'equal in distribution' and $\boldsymbol{\epsilon}_c, c = 1, \cdots, C$ means C independent standard normal vectors.

By noticing that $\hat{\boldsymbol{\mu}}_c + \hat{\boldsymbol{\Lambda}}_c^{-1/2}\boldsymbol{\epsilon}_c$ has the same distribution as $\boldsymbol{x}^{(c)}$ if $f_c(\boldsymbol{x})$ is a Gaussian distribution, the lemma is proved.

References

Agarwal A. and Duchi J. C. (2012). Distributed delayed stochastic optimization, *51st IEEE Conference on Decision and Control*, Maui Hawaii, USA.

Berger O. J. (1980). *Statistical Decision Theory and Bayesian Analysis*, Springer-Verlag, New York, Inc.

Beskos A., Papaspiliopoulos O., Roberts G. O. and Fearnhead P. (2006a). Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of Royal Statistical Society*, B, **68**: 333–382.

Beskos, A. and Papaspiliopoulos, O. and Roberts, G.O. (2006b), Retrospective Exact Simulation of Diffusion Sample Paths with Applications, *Bernoulli*, **12**(6):1077–1098.

Beskos A., Papaspiliopoulos O. and Roberts G. O. (2008). A factorisation of diffusion measure and finite sample path constructions. *Methodology and Computing in Applied Probability*, **10**: 85–104.

Beskos A. and Roberts G. O. (2005). An exact simulation of diffusions. *The Annals of Applied Probability*, **15**: 2422–2444.

Chen, N. and Huang, Z. (2013). Localisation and Exact Simulation of Brownian Motion Driven Stochastic Differential Equations. *Mathematics of Operational Research*, **38**: 591–616.

Dacunha-Castelle, D. and Florens-Zmirou, D. (1986). Estimation of the coefficients of a diffusion from discrete observations *Stochastics*, **19**: 263–284.

Dai H. (2014). *Exact simulation for diffusion bridges – an adaptive approach*, *Journal of Applied Probability*, **51**:346-358.

Dai H. (2017). *A new rejection sampling method without using hat function*, *the Bernoulli Journal*, **23**:2434-2465.

Fleiss J. L. (1993). Statistical basis of meta-analysis. *Statistical methods in medical research*, **2**(2):121-145.

Genest, C. and Zidek, J.V. (1986). Combining probability distributions: A critique and an annotated bibliography. *Statistical Science*, **1**: 114-135.

Hansen. N. R. (2003). Geometric ergodicity of discrete-time approximations to multivariate diffusions. *Bernoulli*, **9**(4):725-743.

Li, C. and Srivastava, S. and Dunson, D.B. (2017), Simple, scalable and accurate posterior interval estimation, *Biometrika*, **104**(3):665–680.

Li, Y. and Yang M. and Zhang Z. (2015), Multi-view representation learning: A survey from shallow methods to deep methods, *Journal of Latex Class Files*, **14**(8), August.

Masuda H. (2004). On multidimensional Ornstein-Uhlenbeck processes driven by a general Levy Process. *Bernoulli*, **10**(1):97-120.

Minsker, S. and Srivastava, S. and Lin, L. and Dunson, D.B. (2014), Scalable and robust Bayesian inference via the median posterior, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 1656–1664.

Neiswanger W., Wang C. and Xing E. (2014). Asymptotically Exact, Embarrassingly Parallel MCMC. *Proceedings of the Thirtieth Conference on Uncertainty In Artificial Intelligence (2014)*, 623–632.

Pollock, M. (2013), Some Monte Carlo Methods for Jump Diffusions, PhD Thesis, Department of Statistics, University of Warwick.

Pollock M, Fearnhead P., Johansen A. M. and Roberts G. O. (2016a). The Scalable Langevin Exact Algorithm: Bayesian Inference for Big Data, submitted to *Journal of Royal Statistical Society*, B.

Pollock, M. and Johansen, A.M. and Roberts, G.O. (2016b), On the Exact and ϵ-Strong Simulation of (Jump) Diffusions, *Bernoulli*, **22**:(2), 794–856.

Roberts G. O. and Stramer O. (2001). On inference for partially observed nonlinear diffusion models using the Metropolis-Hastings algorithm. *Biometrika*, **88**:603-621.

Scott, S.L. and Blocker, A.W. and Bonassi, F.V. and Chipman, H.A. and George, E.I. and McCulloch, R.E. (2016). Bayes and Big Data: The Consensus Monte Carlo Algorithm. *International Journal of Management Science and Engineering Management*, **11**(2):78-88.

Smith, T.C. and Spiegelhalter, D.J. and Thomas, A. (1995). Bayesian approaches to random-effects meta-analysis: A comparative study *Statistics in Medicine*, **14**: 2685–2699.

Srivastava, S. and Cevher, V. and Tan-Dinh, Q. and Dunson, D.B. (2016), WASP: Scalable Bayes via barycenters of subset posteriors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2016)*, 912–920.

Stamatakis A. and Aberer A. J. (2013). Novel Parallelization Schemes for Large-Scale Likelihood-based Phylogenetic Inference. 2013 IEEE 27th International Symposium on Parallel and Distributed Processing, DOI: 10.1109/IPDPS.2013.70.

Tan, A., Doss, H. and Hobert, J.P. (2015) Honest importance sampling with multiple Markov chains. *Journal of Computational and Graphical Statistics*, **24:**, 792-826.

Wang X. and Dunson. (2013). Parallelizing MCMC via Weierstrass Sampler. arXiv preprint arXiv:1312.4605.

Zhao J. and Xie X. and Xu X. and Sun S. (2017). Multi-view learning overview: Recent progress and new challenges *Information Fusion*, **38**: 43–54.