Massively Parallel Reweighted Wake-Sleep

Thomas Heap[1] **Gavin Leech**[1] **Laurence Aitchison**[1]

[1]Department of Computer Science, University of Bristol, Bristol

Abstract

Reweighted wake-sleep (RWS) is a machine learning method for performing Bayesian inference in a very general class of models. RWS draws K samples from an underlying approximate posterior, then uses importance weighting to provide a better estimate of the true posterior. RWS then updates its approximate posterior towards the importance-weighted estimate of the true posterior. However, recent work [Chatterjee and Diaconis, 2018] indicates that the number of samples required for effective importance weighting is exponential in the number of latent variables. Attaining such a large number of importance samples is intractable in all but the smallest models. Here, we develop massively parallel RWS, which circumvents this issue by drawing K samples of all n latent variables, and individually reasoning about all K^n possible combinations of samples. While reasoning about K^n combinations might seem intractable, the required computations can be performed in polynomial time by exploiting conditional independencies in the generative model. We show considerable improvements over standard "global" RWS, which draws K samples from the full joint.

1 INTRODUCTION

Many machine learning tasks involve inferring the latent variables from underlying observations [Jaynes, 2003, MacKay et al., 2003]. One approach to inferring these latent variables from data is to use Bayesian inference. In Bayesian inference, we define a generative model which consists of a prior, $\mathrm{P}\left(\text{latents}\right)$, describing the probability of the latent variable before seeing data, and a likelihood, $\mathrm{P}\left(\text{data}|\text{latents}\right)$, describing the probability of the data given the latents. The goal is then to compute the posterior using Bayes theorem,

$$\mathrm{P}\left(\text{latents}|\text{data}\right) \propto \mathrm{P}\left(\text{data}|\text{latents}\right)\mathrm{P}\left(\text{latents}\right). \quad (1)$$

However, computing this posterior is typically intractable, especially in more complex models where the likelihood or prior is parameterised by a neural network.

As an alternative, modern approaches such as variational inference [VI; Jordan et al., 1999, Wainwright et al., 2008, Kingma and Welling, 2013, Rezende et al., 2014, Blei et al., 2017, Nguyen et al., 2017, Zhang et al., 2018, Kingma et al., 2019, Gayoso et al., 2021] and reweighted wake-sleep [RWS; Bornschein and Bengio, 2014, Le et al., 2020] learn the parameters, ϕ, of an approximate posterior, $\mathrm{Q}_\phi\left(\text{latents}|\text{data}\right)$. In VI, we learn this posterior by optimizing the evidence lower-bound objective (ELBO) using the reparameterisation trick [Kingma and Welling, 2013, Rezende et al., 2014]. This bound often has considerable slack, which can bias inferences. To address this issue importance weighted auto-encoders [IWAEs; Burda et al., 2015, Cremer et al., 2017] draw multiple samples from the approximate posterior and use importance weighting to provide a tighter bound on the model evidence. In RWS, we draw multiple samples from the approximate posterior, reweight those samples to approximate the true posterior, then update the approximate posterior towards the reweighted approximation of the true posterior (specifically, this is the wake-phase Q update; see Bornschein and Bengio, 2014).

However, recent work [Chatterjee and Diaconis, 2018] showed that the number of samples required to get accurate importance weighted estimates is very large. Specifically, they showed that the required number of samples scales as $e^{\mathrm{D_{KL}}(\mathrm{P}(z|x)\|\mathrm{Q}(z|x))}$. This is particularly problematic because we expect the KL divergence to scale linearly in the number of latent variables, n. Indeed, if $\mathrm{P}\left(z|x\right)$ and $\mathrm{Q}\left(z|x\right)$ are IID over the n latent variables, then the KL-divergence is exactly proportional to n. Overall, this implies that we expect the required number of samples to be exponential in the number of latent variables, which is clearly infeasible for larger models.

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

This problem has been addressed in the IWAE context using TMC [Aitchison, 2019], which draws K samples for each of the n latent variables, and individually reasons about each of the K^n combinations of samples. Here, we develop an analogous approach for RWS, which we call massively parallel (MP) RWS. Critically, this is not a simple extension of the derivations in Aitchison [2019]. The derivations in Aitchison [2019] are either restricted to factorised approximate posteriors, or use an augmented state-space viewpoint which cannot be applied to RWS. We therefore give very different and considerably more general derivations in Sec. 4. Indeed, these more general derivations allow us to use a more general class of approximate posteriors, even in the original VI setting.

2 RELATED WORK

Of course, our methods are based on fundamental work on VI [Jordan et al., 1999, Wainwright et al., 2008, Kingma and Welling, 2013, Rezende et al., 2014, Blei et al., 2017, Nguyen et al., 2017, Zhang et al., 2018, Kingma et al., 2019, Gayoso et al., 2021], IWAE [Burda et al., 2015, Cremer et al., 2017] and RWS [Bornschein and Bengio, 2014, Le et al., 2020].

Perhaps the most obvious related work is TMC [Aitchison, 2019], which also draws K samples for each of the n latent variables, and considers all K^n combinations. The key difference to our work is that TMC only applies to VI, while our work applies to RWS. However, our more general derivations improve on TMC itself. Specifically, TMC is restricted to approximate posteriors that are IID across the K particles for one latent variable. In contrast, our derivations allow us to couple the distribution over K particles for a single latent variable (Appendix 2), which gives scope for e.g. applying variance reduction strategies.

Further, there is a body of work improving VI, but not RWS in specific restricted classes of model.

The first model class is a single-level hierarchical model, with a Bayesian parameter, z_0, common to all datapoints, and latent variables, $z_1 \dots z_n$, each associated with a different datapoint. Geffner and Domke [2022] propose a "local" importance weighting (LIW) scheme for this class of model, which contrasts with standard importance weighting schemes that they describe as "global". We adopt their "global" terminology for standard IWAE and RWS, which draw K samples from the full joint approximate posterior. LIW in effect does IWAE separately for each datapoint: it separately draws K IWAE samples for the latent variables, $z_1 \dots z_n$, associated with each datapoint, $x_1, \dots, x_n$. This looks very similar to TMC and massively parallel RWS, which draw K samples for the Bayesian parameter, z_0 and the latent variables, $z_1, \dots, z_n$, and reasons about all K^{n+1} combinations of all samples on $z_0, z_1, \dots, z_n$. However LIW differs from TMC and massively parallel RWS in that LIW draws only a single sample of the Bayesian parameter, z_0. Of course, there are additional differences. In particular, LIW, like TMC, ultimately performs VI, while massively parallel RWS applies RWS. Further, massively parallel RWS (like TMC) can be applied to a very broad class of models, while LIW is restricted to these single-level hierarchical models.

A second class of models is timeseries models. Massively parallel methods in timeseries may bear some resemblance to particle filtering/sequential Monte-Carlo (SMC) [Gordon et al., 1993, Doucet et al., 2009, Andrieu et al., 2010, Maddison et al., 2017, Le et al., 2017, Lindsten et al., 2017, Naesseth et al., 2018, Lai et al., 2022], in that SMC/particle filters also reason over multiple samples for each latent variable. However, work which learns a proposal/approximate posterior in the particle filtering setting focuses on VI rather than RWS [Maddison et al., 2017, Le et al., 2017, Lindsten et al., 2017, Naesseth et al., 2018, Lai et al., 2022]. Moreover, most work in SMC / particle filtering considers only a restrictive class of timeseries model, while massively parallel methods operate in a very general class of models. While there is some work extending SMC to more general generative models [e.g. Lindsten et al., 2017], this work does not, for instance, give a mechanism to learn an approximate posterior using e.g. IWAE or RWS, let alone to have an approximate posterior whose structure differs from that of the underlying generative model.

3 BACKGROUND

Here, we give background on IWAE and RWS, which are methods for performing Bayesian inference in a probabilistic generative model. Both IWAE and RWS work with a collection of K samples of the latent variables. The full collection of K samples is denoted z, while an individual sample (specifically the kth sample) is denoted z^k,

$$z = (z^1, z^2, \dots, z^K) \in \mathcal{Z}^K. \tag{2}$$

For standard global VI and RWS, K samples are drawn by sampling K times from the underlying single-sample approximate posterior, $\mathrm{Q}_\phi\left(z^k|x\right)$, which has parameters, ϕ,

$$\mathrm{Q}_{\text{global}}\left(z|x\right) = \prod_{k\in\mathcal{K}} \mathrm{Q}_\phi\left(z^k|x\right), \tag{3}$$

where $\mathcal{K} = \{1, \dots, K\}$.

IWAE and RWS can be written in terms of an unbiased estimator of the marginal likelihood (Appendix 3.1.2),

$$\mathcal{P}_{\text{global}}(z) = \frac{1}{K}\sum_{k\in\mathcal{K}} r_k(z), \tag{4}$$

$$r_k(z) = \frac{\mathrm{P}_\theta\left(x, z^k\right)}{\mathrm{Q}_\phi\left(z^k|x\right)}, \tag{5}$$

where $\mathrm{P}\left(x, z^k\right)$ is the generative probability, and $r_k(z)$ is the ratio of generative and approximate posterior probabilities, $r(z^k)$.

3.1 IMPORTANCE WEIGHTED AUTOENCODER

In IWAE [Burda et al., 2015], we optimize ϕ and θ using the IWAE objective, $\mathcal{L}_{\text{global}}$, which forms a lower-bound on the marginal likelihood, $\log \mathrm{P}_\theta(x)$,

$$\log \mathrm{P}_\theta(x) \geq \mathcal{L}_{\text{global}}(\theta, \phi) = \mathrm{E}_{\mathrm{Q}_{\text{global}}(z|x)}\left[\log \mathcal{P}_{\text{global}}(z)\right] \tag{6}$$

Differentiating this objective wrt the parameters of the generative model is straightforward, as $\mathrm{Q}_\phi(z|x)$ does not depend on θ so we can interchange the expectation and gradient operators. In contrast, the distribution over which the expectation is taken does depend on ϕ, so the ϕ update is more difficult to implement and requires reparameterisation [Kingma and Welling, 2013, Rezende et al., 2014].

3.2 REWEIGHTED WAKE-SLEEP

In RWS [Bornschein and Bengio, 2014], we do not have a single unified objective. Instead, we update the generative model and approximate posterior by drawing K samples from an approximate posterior, $\mathrm{Q}_\phi\left(z^k|x\right)$. We then use importance reweighting to bring those samples closer to the true posterior, $\mathrm{P}_\theta(z|x)$, and do a maximum likelihood-like update with those reweighted samples. In particular, the P update resembles the M-step in EM, and maximizes $\log \mathrm{P}_\theta\left(z^k, x\right)$ for the reweighted samples. Likewise, the (wake-phase) Q update maximizes $\log \mathrm{Q}_\phi\left(z^k|x\right)$ for the reweighted samples mirroring the true posterior, and therefore brings $\mathrm{Q}_\phi\left(z^k|x\right)$ closer to the true posterior,

$$\Delta\theta_{\text{global}} = \tag{7a}$$
$$\mathrm{E}_{\mathrm{Q}_{\text{global}}(z|x)}\left[\frac{1}{K}\sum_{k\in\mathcal{K}}\frac{r_k(z)}{\mathcal{P}_{\text{global}}(z)}\nabla_\theta \log \mathrm{P}_\theta\left(z^k, x\right)\right],$$

$$\Delta\phi_{\text{global}} = \tag{7b}$$
$$\mathrm{E}_{\mathrm{Q}_{\text{global}}(z|x)}\left[\frac{1}{K}\sum_{k\in\mathcal{K}}\frac{r_k(z)}{\mathcal{P}_{\text{global}}(z)}\nabla_\phi \log \mathrm{Q}_\phi\left(z^k|x\right)\right].$$

See Appendix 3.2.1 for a derivation of these updates. However, it turns out that implementing the updates in (Eq. 7) directly is difficult, as it requires us to separately compute the gradients for each sample, z^k. Instead, we typically use,

$$\Delta\theta_{\text{global}} = \mathrm{E}_{\mathrm{Q}_{\text{global}}(z|x)}\left[\nabla_\theta \log \mathcal{P}_{\text{global}}(z)\right], \tag{8a}$$
$$\Delta\phi_{\text{global}} = \mathrm{E}_{\mathrm{Q}_{\text{global}}(z|x)}\left[\nabla_\phi\left(-\log \mathcal{P}_{\text{global}}(z)\right)\right]. \tag{8b}$$

See Appendix 1 for a proof of equivalence.

4 METHODS

These previous approaches draw K samples from the full joint latent space. However, the required number of samples scales exponentially in the number of latent variables [Chatterjee and Diaconis, 2018]. Thus, we define a massively parallel scheme in which we draw K samples for each latent variable, then effectively obtain K^n samples by considering all combinations of K samples for each of the n latent variables. To that end, we denote each of the separate samples for separate latent variables $z_i^k \in \mathcal{Z}_i$, where k indexes the sample and i indexes the latent variable. We can write the collection of K samples for a single latent variable (the ith) as,

$$z_i = \left(z_i^1, z_i^2, \ldots, z_i^K\right) \in \mathcal{Z}_i^K. \tag{9}$$

To sample all K copies of the full joint latent space, TMC [Aitchison, 2019] uses an IID distribution over the K samples, $z_i^1, \ldots, z_i^K$,

$$\mathrm{Q}_{\text{TMC}}(z|x) = \prod_{i=1}^{n}\prod_{k\in\mathcal{K}} \mathrm{Q}_{\text{TMC}}\left(z_i^k \middle| z_j \text{ for all } j \in \text{qa}(i)\right). \tag{10}$$

Here, $\text{qa}(i)$ are the indices of parents of the ith latent variable under the approximate posterior. In contrast, massively parallel methods allow for dependencies between the K samples for the ith latent variable, $z_i^1, \ldots, z_i^K$ (Appendix 2),

$$\mathrm{Q}_{\text{MP}}(z|x) = \prod_{i=1}^{n} \mathrm{Q}_{\text{MP}}\left(z_i | z_j \text{ for all } j \in \text{qa}(i)\right). \tag{11}$$

There are no formal constraints on these dependencies. However, there are practical constraints, namely that we need to be able to efficiently compute the single-particle marginals, $\mathrm{Q}\left(z_i^k \middle| z_j \text{ for all } j \in \text{qa}(i)\right)$. In Appendix 2, we give more specifics about choices of $\mathrm{Q}_{\text{MP}}\left(z_i | z_j \text{ for all } j \in \text{qa}(i)\right)$ and $\mathrm{Q}_{\text{TMC}}\left(z_i^k \middle| z_j \text{ for all } j \in \text{qa}(i)\right)$. At a high level, these distributions are constructed by mixing the underlying single-sample approximate posterior, Q_ϕ, for different combinations of parent particles.

The generative model is more complicated, because we want to evaluate the generative probability for any of the K^n possible combinations of the K samples of the n latent variables. To facilitate writing down these generative probabilities, we begin by defining a vector of indices,

$$\mathbf{k} = (k_1, k_2, \ldots, k_n) \in \mathcal{K}^n, \tag{12}$$

which has one index, k_i, for each of the n latent variables. The latent variables specified by these indices is known as the "indexed" latent variables and can be written,

$$z^{\mathbf{k}} = \left(z_1^{k_1}, z_2^{k_2}, \ldots, z_n^{k_n}\right) \in \mathcal{Z}. \tag{13}$$

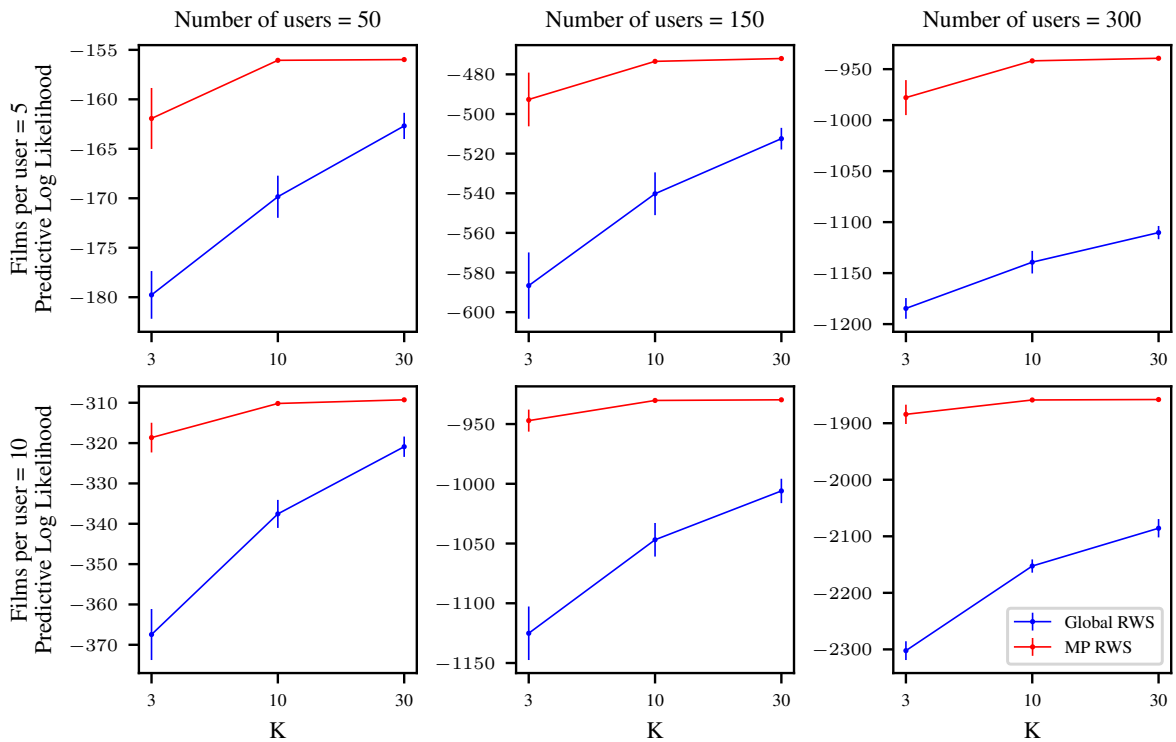

Figure 1: Results of massively parallel RWS and standard or "global" RWS for a hierarchical model on subsets of MovieLens with differing numbers of users and films per user, showing the predictive log likelihood after 25k training iterations.

The generative probability can thus be written,

$$\mathrm{P}_\theta\left(x, z^{\mathbf{k}}\right) = \mathrm{P}_\theta\left(x \middle| z_j^{k_j} \text{ for all } j \in \mathrm{pa}\left(x\right)\right) \prod_{i=1}^{n} \mathrm{P}_\theta\left(z_i^{k_i} \middle| z_j^{k_j} \text{ for all } j \in \mathrm{pa}\left(i\right)\right). \tag{14}$$

Here, $\mathrm{pa}\left(i\right)$ are the indices of parents of the ith latent variable under the generative model, and $\mathrm{pa}\left(x\right)$ are the parents of the data under the generative model. Our use of $\mathrm{pa}\left(i\right)$ mirrors our use of $\mathrm{qa}\left(i\right)$ to denote indices of parents of the ith latent variable under the approximate posterior. Of course, the structure of the generative model and approximate posterior may differ, so $\mathrm{qa}\left(i\right)$ and $\mathrm{pa}\left(i\right)$ can also differ.

Looking at $\mathcal{P}_{\text{global}}(z)$ (Eq. 4) we average only over K terms, corresponding to our K samples from the full joint latent space. Our key contribution is to adapt RWS for the case where we average over all K^n combinations of samples for each latent variable, indexed $\mathbf{k}$.

We can define an alternative unbiased marginal likelihood estimator, $\mathcal{P}_{\text{MP}}(z)$. This estimator is obtained by averaging over all K^n combinations of all samples of all latent variables,

$$\mathcal{P}_{\text{MP}}(z) = \frac{1}{K^n} \sum_{\mathbf{k} \in \mathcal{K}^n} r_{\mathbf{k}}(z), \tag{15}$$

$$r_{\mathbf{k}}(z) = \frac{\mathrm{P}\left(x, z^{\mathbf{k}}\right)}{\prod_i \mathrm{Q}_{\text{MP}}\left(z_i^{k_i} \middle| z_j \text{ for all } j \in \mathrm{qa}\left(i\right)\right)} \tag{16}$$

For the proof that $\mathcal{P}_{\text{MP}}(z)$ is an unbiased marginal likelihood estimator, see Appendix 3.1.3. By analogy with global IWAE, we can define an objective for massively parallel IWAE,

$$\mathcal{L}_{\text{MP}} = \mathrm{E}_{\mathrm{Q}_{\text{MP}}(z|x)}\left[\log \mathcal{P}_{\text{MP}}(z)\right]. \tag{17}$$

We prove that this quantity has the required properties (specifically, that it is a lower-bound on the log marginal likelihood) in Appendix 3.1.3. This quantity is very similar to that given in [Aitchison, 2019], except that it allows for a slightly more general proposal, Q_{MP}, which allows for dependencies between the K samples for a single latent variable, $z_i^1, \ldots, z_i^K$. Our key contribution is to design

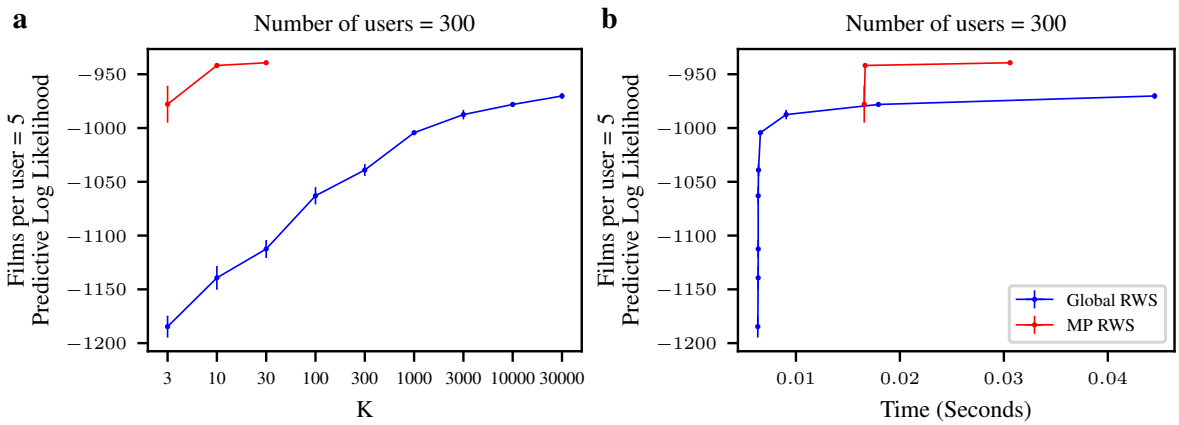

Figure 2: Comparison of performance between massively parallel RWS and global RWS, for the movielens dataset. **a** Same as Fig. 1 (top right), except that we use much higher values of K for global RWS. **b** As massively parallel RWS may take longer than global RWS for a given K, we plotted time for a single training iteration against the predictive log-likelihood.

massively parallel updates for RWS,

$$\Delta\theta_{\text{MP}} = \mathrm{E}_{\mathrm{Q}_{\text{MP}}(z|x)}\left[\frac{1}{K^n}\sum_{\mathbf{k}\in\mathcal{K}^n}\frac{r_{\mathbf{k}}(z)}{\mathcal{P}_{\text{MP}}(z)}\nabla_\theta \log \mathrm{P}_\theta\left(z^{\mathbf{k}}, x\right)\right] \tag{18a}$$

$$\Delta\phi_{\text{MP}} = \mathrm{E}_{\mathrm{Q}_{\text{MP}}(z|x)}\left[\frac{1}{K^n}\sum_{\mathbf{k}\in\mathcal{K}^n}\frac{r_{\mathbf{k}}(z)}{\mathcal{P}_{\text{MP}}(z)}\nabla_\phi \log \mathrm{Q}_\phi\left(z^{\mathbf{k}}, x\right)\right] \tag{18b}$$

These updates are derived in Appendix 3.2.2, and they can be implemented using,

$$\Delta\theta_{\text{MP}} = \mathrm{E}_{\mathrm{Q}_{\text{MP}}(z|x)}\left[\nabla_\theta \log \mathcal{P}_{\text{MP}}(z)\right], \tag{19a}$$

$$\Delta\phi_{\text{MP}} = \mathrm{E}_{\mathrm{Q}_{\text{MP}}(z|x)}\left[\nabla_\phi \left(-\log \mathcal{P}_{\text{MP}}(z)\right)\right], \tag{19b}$$

(see Appendix 3.2.1).

Algorithm 1 Massively Parallel RWS

Require: Data x, Prior P_θ, Proposal Q_{MP}, $K \geq 1$
  for $i \leftarrow 1$ to n **do**
    Sample $z_i \sim \mathrm{Q}_{\text{MP}}\left(z_i | z_j \text{ for all } j \in \text{qa}(i)\right)$
    $z \leftarrow \{z_1, ..., z_{i-1}\} \cup z_i$
    $f^i_{k_i,\mathbf{k}_{\text{pa}(i)}}(z) \leftarrow \frac{\mathrm{P}_\theta\left(z_i^{k_i} \middle| z_j^{k_j} \text{ for all } j\in\text{pa}(i)\right)}{\mathrm{Q}_{\text{MP}}\left(z_i^{k_i}|x, z_j \text{ for all } j\in\text{qa}(i)\right)}$
  end for
  $f^x_{\mathbf{k}_{\text{pa}(x)}}(z) \leftarrow \mathrm{P}_\theta\left(x \middle| z_j^{k_j} \text{ for all } j \in \text{pa}(x)\right)$
  $\mathcal{P}_{\text{MP}}(z) \leftarrow \frac{1}{K^n}\sum_{\mathbf{k}^n} f^x_{\mathbf{k}_{\text{pa}(x)}}(z)\prod_i f^i_{k_i,\mathbf{k}_{\text{pa}(i)}}(z)$
  $\Delta\theta_{\text{MP}} \leftarrow \nabla_\theta \log \mathcal{P}_{\text{MP}}(z)$
  $\Delta\phi_{\text{MP}} \leftarrow \nabla_\phi\left(-\log \mathcal{P}_{\text{MP}}(z)\right)$

4.1 EFFICIENTLY AVERAGING EXPONENTIALLY MANY TERMS

It should be surprising that we can compute $\mathcal{P}_{\text{MP}}(z)$ (Eq. 15) efficiently, as it involves summing over exponentially many (K^n) terms. However, it turns out that efficient computation is possible if we exploit structure in the generative model. To exploit structure, we first need to write down the generative probability for the $\mathbf{k}$th sample of all latent variables, $z^{\mathbf{k}}$. This looks alot like Eq. (14), as it follows the same graphical model structure, with pa (x) and pa (i) giving the indices of parents of the data and the ith latent variable respectively,

$$r_{\mathbf{k}}(z) = \mathrm{P}_\theta\left(x \middle| z_j^{k_j} \text{ for all } j \in \text{pa}(x)\right) \prod_{i=1}^{n}\frac{\mathrm{P}_\theta\left(z_i^{k_i} \middle| z_j^{k_j} \text{ for all } j \in \text{pa}(i)\right)}{\mathrm{Q}_{\text{MP}}\left(z_i^{k_i}|x, z_j \text{ for all } j \in \text{qa}(i)\right)}. \tag{20}$$

If we fix z (i.e. all K^n samples of all n latent variables), then $r_{\mathbf{k}}(z)$ can be regarded as a big tensor with K^n elements, indexed by $\mathbf{k}$. In that case, each term in the product defining $r_{\mathbf{k}}(z)$ (Eq. 20) can also be regarded as a tensor. The key observation is that the individual tensors in the product typically have only a few indices. For instance, the probability of the data, x, depends only on the indices of samples of the parents (i.e. $(k_j \text{ for all } j \in \text{pa}(x))$). These indices of the samples of the parents can be written,

$$\mathbf{k}_{\text{pa}(x)} = (k_j \text{ for all } j \in \text{pa}(x)) \in \mathcal{K}^{|\text{pa}(x)|}, \tag{21a}$$

$$\mathbf{k}_{\text{pa}(i)} = (k_j \text{ for all } j \in \text{pa}(i)) \in \mathcal{K}^{|\text{pa}(i)|}. \tag{21b}$$

and where $|\text{pa}(x)|$ and $|\text{pa}(i)|$ are the number of parents latent variables. To make explicit the idea that the individual terms in Eq. (20) can be understood as tensors, we define $f^x_{\mathbf{k}_{\text{pa}(x)}}(z)$ as the tensor for the data and $f^i_{k_i,\mathbf{k}_{\text{pa}(i)}}(z)$ as the tensor for the ith latent variable,

$$f^x_{\mathbf{k}_{\text{pa}(x)}}(z) = \mathrm{P}_\theta\left(x \middle| z_j^{k_j} \text{ for all } j \in \text{pa}(x)\right), \tag{22a}$$

$$f^i_{k_i,\mathbf{k}_{\text{pa}(i)}}(z) = \frac{\mathrm{P}_\theta\left(z_i^{k_i} \middle| z_j^{k_j} \text{ for all } j \in \text{pa}(i)\right)}{\mathrm{Q}_{\text{MP}}\left(z_i^{k_i}|x, z_j \text{ for all } j \in \text{qa}(i)\right)}. \tag{22b}$$

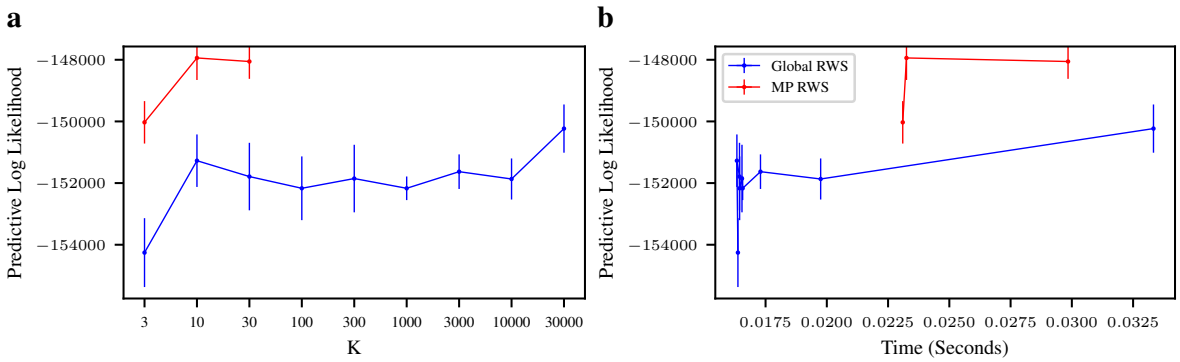

Figure 3: Comparison of performance for massively parallel RWS and global RWS on the NYC bus breakdown dataset. **a** Predictive log-likelihood against K after 75k training iterations. **b** Predictive log-likelihood against the time for a single training iteration.

Thus, we can write $r_{\mathbf{k}}(z)$ as a product of these factors,

$$r_{\mathbf{k}}(z) = f^{x}_{\mathbf{k}_{\text{pa}(x)}}(z) \prod_i f^{i}_{k_i, \mathbf{k}_{\text{pa}(i)}}(z), \tag{23}$$

and $\mathcal{P}_{\text{MP}}(z)$ can be understood as a big tensor product,

$$\mathcal{P}_{\text{MP}}(z) = \frac{1}{K^n} \sum_{\mathbf{k}^n} f^{x}_{\mathbf{k}_{\text{pa}(x)}}(z) \prod_i f^{i}_{k_i, \mathbf{k}_{\text{pa}(i)}}(z). \tag{24}$$

This tensor product can be efficiently computed in polynomial time by ordering the sums and products using Python packages such as opt-einsum [Daniel et al., 2018].

5 EXPERIMENTS

We present an empirical evaluation of massively parallel RWS [1]. Since the RWS wake phase Q requires multiple importance samples we test massively parallel RWS (MP RWS) with $K \in \{3, 10, 30\}$ and global RWS with $K \in \{3, 10, 30, 100, 300, 1000, 3000, 10000, 30000\}$. Unless otherwise stated our variational posterior is of the form $q_\phi(\mathbf{z}) = \prod_{i=1}^{L} q(z_i)$, where $q(z_i)$ is from the same family of distributions as z_i's distribution in the generative model. We compare massively parallel RWS (Eq. 18 and Eq. 19) and against standard "global" RWS (Eq. 7 and Eq. 8).

Optimisation is done using Adam [Kingma and Ba, 2014] with $\beta = (0.9, 0.999)$, no weight decay, and a learning rate of 0.001 which is decreased by a factor of 10 every 10k iterations. In all cases we plot the result of 5 runs with different random seeds and plot the mean and standard error. All times are measured on a single Nvidia A100 GPU.

5.1 MOVIELENS DATASET

We show results on the MovieLens100K dataset [Harper and Konstan, 2015]. This dataset consists of 100K ratings from $\mathrm{M} = 943$ users (indexed m) of $\mathrm{N} = 1682$ films (indexed j). Each film, indexed j, has as a feature vector $\mathbf{x}_j$. We observe user ratings, and following [Geffner and Domke, 2022], binarise ratings of $(0, 1, 2, 3)$ to 0 and ratings of $(4, 5)$ to 1. We use the following hierarchical model:

$$\begin{aligned} \boldsymbol{\mu} &\sim \mathcal{N}(\mathbf{0}_{18}, 1) \\ \psi &\sim \text{Categorical}([0.1, 0.5, 0.4, 0.05, 0.05]) \\ \mathbf{z}_m &\sim \mathcal{N}(\boldsymbol{\mu}, \exp(\psi)\mathbf{I}),\ m = 1, \ldots, M \\ \text{Rating}_{mj} &\sim \text{Bernoulli}(\sigma(\mathbf{z}_m^\intercal \mathbf{x}_j)),\ j = 1, \ldots, \mathrm{N} \end{aligned} \tag{25}$$

This model, first samples a global mean, $\boldsymbol{\mu}$, and a discrete variance, ψ. We then sample a vector, $\mathbf{z}_m$ for each user, which describes the types of films that they will rate highly. The probability of a high rating is then given by taking the dot-product of the latent user-vector, $\mathbf{z}_m$ and the film's feature vector, $\mathbf{x}_j$. A corresponding graphical model can be seen in Appendix 3.3.

Note that this model has a discrete latent variable ψ. As RWS does not reparameterise gradients of the ELBO, inference can proceed straightforwardly, without needing any approaches to discrete latent variables from VI, such as summing out the latent variable, applying REINFORCE gradient estimators or using continuous relaxations [Le et al., 2020]. We compare the two methods by calculating the predictive log likelihood on a test set the same size as the training set.

To evaluate inference methods effectively, it is important to ensure that the posterior distributions are broad, and have not collapsed to very narrow point-like distributions. As such, we evaluate on subsets of the full MovieLens dataset, composed of either 5 or 10 films per user, and 50, 150 or 300 users.

Results are shown in Fig. 1 and Fig. 2. massively parallel RWS gives considerably higher predictive log-likelihoods for all K (Fig. 1, 2a). Importantly, the massively parallel RWS updates are more complex than the global RWS updates, so may take longer. We therefore also considered

[1] Code for reproduction of experiments can be found: `https://github.com/ThomasHeap/MPRW-S`

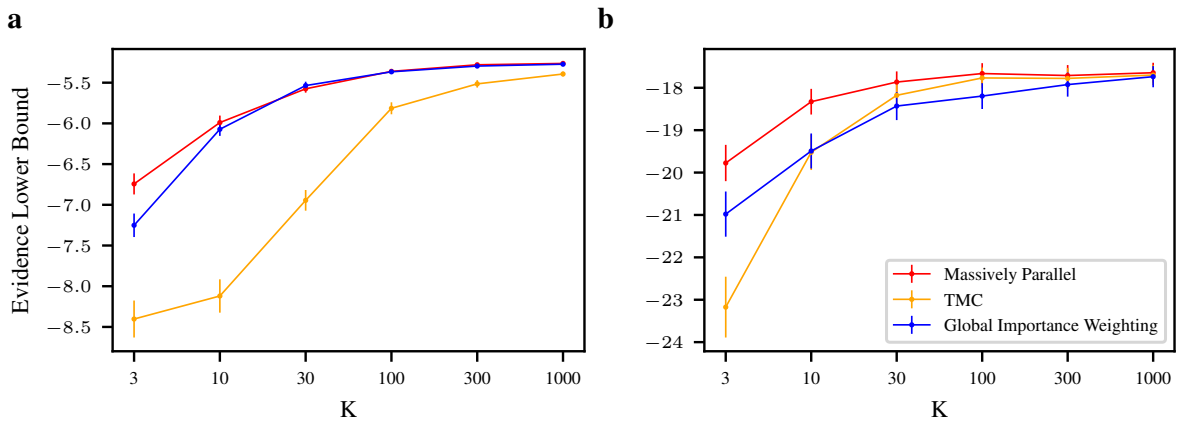

Figure 4: Comparison of performance between massively parallel methods, TMC, particle filter and global importance weighting. **a** The timeseries model with one observation. **b** The timeseries model with multiple observations.

the performance, measured as the predictive log-likelihood, against the time for a single training iteration. We again found considerable, albeit less dramatic, improvements (Fig. 2b).

5.2 NYC BUS BREAKDOWN DATASET

The city of New York releases data on the length of delays to school bus journeys [DOE, 2023]. We model the length of the delay in terms of the type of journey, the school year in which the delay occurred, the borough the delay occurred in and the ID of the bus that was delayed.

To model delay time we use the model outlined in Appendix 3.4. Because the dataset can be stratified into a hierarchy with three levels (Year, Borough and ID) we want our model to reflect this and, inspired by attempts to use hierarchical regression to model radon levels indoor radon levels [Price et al., 1996], we use a similar multi-level regression with three levels. This model first samples a variance and mean for each year, then uses these to sample a borough mean for each year. A variance is then sampled for each borough, which together with the year level borough mean is used to sample an ID mean for each year and borough. Finally, a variance is sampled and used to sample two weight vectors, $\mathbf{C}_i$ which has length "Number of bus companies" and $\mathbf{J}_i$ which has length "Number of types of journeys". These are used to weight covariates that indicate which bus company was running a given ID's route and which type of journey was being undertaken respectively. These are then summed with the sampled ID mean for that year and borough to get the logits for a negative binomial distribution that then gives the predicted delay for the i-th ID in the j-th borough in the m-th year. A corresponding graphical model can be seen in Appendix 3.5.

We evaluate this model using a training dataset with 270 observations: $I = 30$ Ids from $J = 3$ Boroughs in $M = 3$ Years. We perform RWS for 75k iterations, and evaluate the predictive log likelihood on a held out test set the same size as the training set.

Results are shown in Fig. 3. Again we see that massively parallel RWS outperforms global RWS for all K.

5.3 COMPARING MP VI WITH TMC

Even though our main contribution is in developing massively parallel RWS, our derivations also allow for slightly more general massively parallel approaches to VI. In particular, our derivations allow us to couple the proposal for the K samples of the ith latent variable, $z_i^1, \ldots, z_i^K$, while TMC [Aitchison, 2019] forces these K samples to be IID. This coupling in massively parallel methods allows us to introduce variance-reduction strategies inspired by methods for reducing particle degeneracy in particle filters [Carpenter et al., 1999, Li et al., 2012, 2014, Zhou et al., 2016, Wang et al., 2017] (see Appendix 2 for further details).

To highlight these advantages, we considered two toy timeseries models: a single observation and a multi- observation model.

5.3.1 Single Observation

In the single observation model, there is a latent timeseries $z_1, \ldots, z_{30}$ (we use $N = 30$), and an observation, x, only at the last timestep,

$$\begin{aligned} z_1 &= 0, \\ z_i &\sim \mathcal{N}(z_{i-1}, 1/N), \\ x &\sim \mathcal{N}(z_N, 1) \end{aligned} \tag{26}$$

We use the prior to define the proposal (see Appendix 2). Results can be seen in figure 4a. For large K, all methods converge to the same value, as the ELBOs are all bounded by

the true model evidence. To compare the methods, we therefore need to consider their relative performance for smaller values of K. We can see that the TMC (orange) [Aitchison, 2019] performs considerably worse than massively parallel VI (red) and IWAE (blue) [Burda et al., 2015]. We believe that TMC is performing poorly because of particle degeneracy [Carpenter et al., 1999, Li et al., 2012, 2014, Zhou et al., 2016, Wang et al., 2017]. In particular, the TMC proposal for z_i is given by a mixture of the prior, conditioned particles from the previous timestep, z_{i-1}. In sampling from this mixture, in essence, we first sample a parent particle, $z_{i-1}^{k_{i-1}}$, then we sample from the prior, conditioned on that parent sample, $\mathrm{P}\left(z_i^{k_i}|z_{i-1}^{k_{i-1}}\right)$. In TMC, we choose these parent sample IID, which means that one parent particle, $z_{i-1}^{k_{i-1}}$ may have zero, one or multiple children. This is problematic: whenever a parent sample has zero children, then this reduces diversity in the samples of z_i, and this issue builds up over timesteps. Massively parallel methods circumvent this issue by ensuring that each parent sample has one and only one child sample (which requires us to couple the distribution over $z_i^1, \ldots, z_i^K$), and IWAE avoids the issue by simply sampling z_i^k conditioned on z_{i-1}^k. Massively parallel is comparable to IWAE in this setting due to conditioning only a single scalar value at the end of the timeseries. These methods separate when we consider multiple observations (next).

5.3.2 Multiple Observations

Next, we considered a more standard timeseries with multiple observations. We are implementing these methods in the context of a new probabilistic programming language. This language currently has limitations on the number of latent variables that are inherited from the opt-einsum implementation. As such, we were not able to do the obvious thing of having one observation at every timestep. Instead, we had an observation every third timestep.

$$\begin{aligned} &z_1 \sim \mathcal{N}(0,1), \\ &z_i \sim \mathcal{N}((1-\tfrac{1}{\tau})z_{i-1}, 2/\tau), \\ &x_i \sim \mathcal{N}(z_i, 1) \qquad \text{if } i \text{ divisible by 3.} \end{aligned} \tag{27}$$

Again, we use $N = 30$. Results can be seen in Fig. 4. Again, the methods converge as K increases, but this time, massively parallel VI (red) gives better performance than both alternatives for lower values of K.

6 CONCLUSION

We introduced massively parallel RWS, in which we draw K samples for n latent variables, and efficiently consider all K^n combinations by exploiting conditional independencies in the generative model. We showed that massively parallel RWS represents a considerable improvement over previous RWS methods that draw K samples from the full joint latent space.

References

Laurence Aitchison. Tensor Monte Carlo: particle methods for the GPU era. *Advances in Neural Information Processing Systems*, 32, 2019.

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

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. *arXiv preprint arXiv:1406.2751*, 2014.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

James Carpenter, Peter Clifford, and Paul Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings-Radar, Sonar and Navigation*, 146(1):2–7, 1999.

Sourav Chatterjee and Persi Diaconis. The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135, 2018.

Chris Cremer, Quaid Morris, and David Duvenaud. Reinterpreting importance-weighted autoencoders. *arXiv preprint arXiv:1704.02916*, 2017.

G Daniel, Johnnie Gray, et al. Opt_einsum-a python package for optimizing contraction order for einsum-like expressions. *Journal of Open Source Software*, 3(26):753, 2018.

DOE. Bus breakdown and delays, 2023. url https://data.cityofnewyork.us/Transportation/Bus-Breakdown-and-Delays/ez4e-fazm Accessed on: 05.05.2023.

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

Adam Gayoso, Zoë Steier, Romain Lopez, Jeffrey Regier, Kristopher L Nazor, Aaron Streets, and Nir Yosef. Joint probabilistic modeling of single-cell multi-omic data with totalVI. *Nature methods*, 18(3):272–282, 2021.

Tomas Geffner and Justin Domke. Variational inference with locally enhanced bounds for hierarchical models. *arXiv preprint arXiv:2203.04432*, 2022.

Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE proceedings F (radar and signal processing)*, volume 140, pages 107–113. IET, 1993.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2): 183–233, 1999.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

Jinlin Lai, Justin Domke, and Daniel Sheldon. Variational marginal particle filters. In *International Conference on Artificial Intelligence and Statistics*, pages 875–895. PMLR, 2022.

Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. Auto-encoding sequential monte carlo. *arXiv preprint arXiv:1705.10306*, 2017.

Tuan Anh Le, Adam R Kosiorek, N Siddharth, Yee Whye Teh, and Frank Wood. Revisiting reweighted wake-sleep for models with stochastic control flow. In *Uncertainty in Artificial Intelligence*, pages 1039–1049. PMLR, 2020.

Tiancheng Li, Tariq Pervez Sattar, and Shudong Sun. Deterministic resampling: unbiased sampling to avoid sample impoverishment in particle filters. *Signal Processing*, 92 (7):1637–1645, 2012.

Tiancheng Li, Shudong Sun, Tariq Pervez Sattar, and Juan Manuel Corchado. Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches. *Expert Systems with applications*, 41(8): 3944–3954, 2014.

Fredrik Lindsten, Adam M Johansen, Christian A Naesseth, Bonnie Kirkpatrick, Thomas B Schön, JAD Aston, and Alexandre Bouchard-Côté. Divide-and-conquer with sequential monte carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458, 2017.

David JC MacKay, David JC Mac Kay, et al. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Chris J Maddison, John Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Teh. Filtering variational objectives. *Advances in Neural Information Processing Systems*, 30, 2017.

Christian Naesseth, Scott Linderman, Rajesh Ranganath, and David Blei. Variational sequential monte carlo. In *International conference on artificial intelligence and statistics*, pages 968–977. PMLR, 2018.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

Phillip N Price, Anthony V Nero, and Andrew Gelman. Bayesian prediction of mean indoor radon concentrations for minnesota counties. *Health Physics*, 71(LBL-35818-Rev), 1996.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2): 1–305, 2008.

Xuedong Wang, Tiancheng Li, Shudong Sun, and Juan M Corchado. A survey of recent advances in particle filters and remaining challenges for multitarget tracking. *Sensors*, 17(12):2707, 2017.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

Haomiao Zhou, Zhihong Deng, Yuanqing Xia, and Mengyin Fu. A new sampling method in particle filter based on pearson correlation coefficient. *Neurocomputing*, 216: 208–215, 2016.