PAVI: Plate-Amortized Variational Inference

Louis Rouillard, Alexandre Le Bris, Thomas Moreau, Demian Wassermann

PAVI: Plate-Amortized Variational Inference

Louis Rouillard *louis.rouillard-odera@inria.fr*
Université Paris-Saclay, Inria, CEA
Palaiseau, 91120, France

Alexandre Le Bris *alexandre.le-bris@inria.fr*
Université Paris-Saclay, Inria, CEA
Palaiseau, 91120, France

Thomas Moreau *thomas.moreau@inria.fr*
Université Paris-Saclay, Inria, CEA
Palaiseau, 91120, France

Demian Wassermann *demian.wassermann@inria.fr*
Université Paris-Saclay, Inria, CEA
Palaiseau, 91120, France

Reviewed on OpenReview: *https://openreview.net/forum?id=ulY9GDCCA6*

Abstract

Given observed data and a probabilistic generative model, Bayesian inference searches for the distribution of the model's parameters that could have yielded the data. Inference is challenging for large population studies where millions of measurements are performed over a cohort of hundreds of subjects, resulting in a massive parameter space. This large cardinality renders off-the-shelf Variational Inference (VI) computationally impractical. In this work, we design structured VI families that efficiently tackle large population studies. Our main idea is to share the parameterization and learning across the different i.i.d. variables in a generative model, symbolized by the model's *plates*. We name this concept *plate amortization*. Contrary to off-the-shelf stochastic VI, which slows down inference, plate amortization results in orders of magnitude faster to train variational distributions. Applied to large-scale hierarchical problems, PAVI yields expressive, parsimoniously parameterized VI with an affordable training time. This faster convergence effectively unlocks inference in those large regimes. We illustrate the practical utility of PAVI through a challenging Neuroimaging example featuring 400 million latent parameters, demonstrating a significant step towards scalable and expressive Variational Inference.

1 Introduction

Inference in graphical models allows to explain data in an interpretable and uncertainty-aware manner (Koller & Friedman, 2009; Gelman et al., 2004; Zhang et al., 2019). Such graphical models are leveraged in Neuroimaging to capture complex phenomena, for instance, the localization of cognitive function across the human brain cortex (Thomas Yeo et al., 2011; Kong et al., 2019). Yet, current inference methods struggle with the high dimensionality of Neuroimaging applications, which can feature hundreds of millions of latent parameters. This work unlocks inference for those large-scale applications via novel scalable VI methodologies.

Our target applications are population studies. Population studies analyze measurements over large cohorts of subjects. They allow to model population parameters —for instance, the mean weight in a population

and different groups of subjects— along with individual variability —the variation of each subject's weight compared to the group weight. These studies are ubiquitous in health care (Fayaz et al., 2016; Towsley et al., 2011), and can typically involve hundreds of subjects and individual measurements per subject. For instance, in Neuroimaging (e.g. Kong et al., 2019), measurements X can correspond to signals in thousands of locations in the brain for a thousand subjects. Given this observed data X, and a generative model that can produce X given model parameters Θ, we want to recover the parameters Θ that could have yielded the observed X. In our Neuroimaging example, Θ can be local labels for each brain location and subject, together with global parameters common to all subjects, such as the connectivity fingerprint corresponding to each label. In many real-world applications, the link between latent parameters and observed data is not deterministic. For instance, the observed signal can be noisy, or the modeled process in itself may be stochastic (Koller & Friedman, 2009). As such, several parameters Θ instances could yield the same data X. We want to model that parameter uncertainty by recovering the *distribution* of the Θ that could have produced X. Following the Bayesian formalism (Gelman et al., 2004), we cast both Θ and X as random variables (RVs), and our goal is to recover the *posterior* distribution $p(\Theta|X)$. Due to the nested structure of our applications, we focus on the case where p corresponds to a hierarchical Bayesian model (HBM) (Gelman et al., 2004). In population studies, the multitude of subjects and measurements per subject implies a large dimensionality for both Θ and X. This large dimensionality in turn creates computational hurdles that we tackle through our method.

Several inference methods for HBMs exist in the literature. Earliest works resorted to Markov chain Monte Carlo (Koller & Friedman, 2009), which become slower as dimensionality increases (Blei et al., 2017). Recent approaches coined Variational Inference (VI) cast the inference as an optimization problem (Blei et al., 2017; Zhang et al., 2019; Ruiz & Titsias, 2019). In VI, inference reduces to choosing a variational family $\mathcal{Q}$ and finding inside that family the distribution $q(\Theta;\phi) \in \mathcal{Q}$ closest to the unknown posterior $p(\Theta|X)$. Historically, VI required to manually derive and optimize over $\mathcal{Q}$, which remains an effective method where applicable (Dao et al., 2021). This *manual* VI requires technical mastery. The experimenter has to choose an appropriate family $\mathcal{Q}$ to approximate the unknown posterior closely. At the same time, the experimenter must rely on properties such as conjugacy to derive an optimizing routine (Dao et al., 2021; Kucukelbir et al., 2016). This means that time and effort must be spent not only on the *modeling* part —laying out hypothesis in a generative model— but also on the *inference* part —inferring parameters given the generative model. For many experimenters, manual VI thus features strong barriers to entry, both in time and technical skill (Kucukelbir et al., 2016). In contrast, we follow the idea of *automatic* VI: deriving an efficient family $\mathcal{Q}$ directly from the HBM p. Automatic VI reduces inference to the *modeling* part only, simplifying the experimenter's research cycle (Kucukelbir et al., 2016; Ambrogioni et al., 2021a;b). We propose an automatic VI method computationally applicable to large population studies.

Our method relies on the pervasive idea of amortization in VI. Amortization can be broadly defined as finding posteriors usable across multiple data points (Zhang et al., 2019). As such, amortization can be interpreted as the meta-learning of the inference problem (Ravi & Beatson, 2019; Iakovleva et al., 2020; Yao et al., 2019). A particular example of meta-learning is Neural Processes, which share with our method the conditioning of a density estimator by the output of a permutation-invariant encoder (Garnelo et al., 2018; Dubois et al., 2020; Zaheer et al., 2018). Though close in spirit to our work, meta-learning studies problems with a single hierarchy. One-hierarchy cases correspond to models distinguishing local parameters, representing the subjects, from global parameters, representing the population (Ravi & Beatson, 2019; Tran et al., 2017). In contrast, we study generic HBMs with an arbitrary number of hierarchies to tackle population studies. Another interesting focus is the two-hierarchy case —population, group and subject— for which Agrawal & Domke (2021) provide theoretical guarantees and explore in length the Gaussian approximation. In contrast, our focus is rather computational to enable the training of generic density approximators such as normalizing flows (Papamakarios et al., 2019). Furthermore, we propose a general method to treat any plate-enriched HBM, as opposed the particular factorization considered by Agrawal & Domke (2021).

Modern VI is effective in low-dimensional settings but does not scale up to large population studies, which can involve millions of random variables (e.g. Kong et al., 2019). In this work, we identify and tackle two challenges to enable this scale-up. A first challenge with scalability is a detrimental trade-off between expressivity and high-dimensionality (Rouillard & Wassermann, 2022). To reduce the inference gap, VI

requires the variational family $\mathcal{Q}$ to contain distributions closely approximating $p(\Theta|X)$ (Blei et al., 2017). Yet the form of $p(\Theta|X)$ is usually unknown to the experimenter. Instead of a lengthy search for a valid family, one can resort to universal density approximators: normalizing flows (Papamakarios et al., 2019). But the cost for this generality is a heavy parameterization, and normalizing flows scale poorly with the dimensionality of Θ. As a result, in large population studies, the parameterization of normalizing flows becomes prohibitive. To tackle this challenge, Rouillard & Wassermann (2022) recently proposed, via the ADAVI architecture, to partially share the parameterization of normalizing flows across the hierarchies of a generative model. We build upon this shared parameterization idea while improving the ADAVI architecture on several aspects: removing the mean-field approximation; extending from pyramidal to arbitrary graphs; generalizing to non-sample-amortized and stochastically trained schemes. Critically, while ADAVI tackled the over-parameterization of VI in population studies, it still could not perform inference in large data regimes. This is due to a second challenge with scalability: as the size of Θ increases, evaluating a single gradient over the entirety of an architecture's weights quickly requires too much memory and computation. This second challenge can be overcome using stochastic VI (SVI, Hoffman et al., 2013), which subsamples the parameters Θ inferred for at each optimization step. However, using SVI, the weights for the posterior for a local parameter $\theta \in \Theta$ are only updated when the algorithm visits θ. In the presence of hundreds of thousands of such local parameters, stochastic VI can become prohibitively slow.

This work introduces the concept of *plate amortization* (PAVI) for fast inference in large-scale HBMs. Instead of considering the inference over local parameters θ as separate problems, our main idea is to share both the parameterization and learning across those local parameters, or equivalently across a model's *plates*. We first propose an algorithm to automatically derive an expressive yet parsimoniously-parameterized variational family from a plate-enriched HBM. We then propose a hierarchical stochastic optimization scheme to train this architecture efficiently. PAVI leverages the repeated structure of plate-enriched HBMs via a novel combination of amortization and stochastic training. Through this combination, our main claim is to enable inference over arbitrarily large population studies with reduced parameterization and training time as the cardinality of the problem augments. Critically, while traditional Stochastic VI unlocked large-scale inference at the cost of slower convergence, PAVI does not feature such a detrimental trade-off. On the contrary, PAVI converges orders of magnitude faster than non-stochastic VI in large regimes. In practice, this quicker training unlocks inference in applications previously unattainable, bringing inference time down from weeks to hours. We illustrate this by applying PAVI to a challenging human brain cortex parcellation, featuring inference over a cohort of 1000 subjects with tens of thousands of measurements per subject, for a total of half a billion RVs. This demonstrates a significant step towards scalable, expressive and fast VI.

2 Problem statement: inference in large population studies

Here we introduce a motivating example for our work that we'll use as an illustration in the rest of this section. The concepts introduced in this section will be more formally defined below in Section 3.

Our objective is to perform inference in large population studies. As an example of how inference becomes impractical in this context, consider $\mathcal{M}$ in Figure 1 (top) as a model for the weight distribution in a population. $\theta_{2,0}$ denotes the mean weight across the population. $\theta_{1,0}, \theta_{1,1}, \theta_{1,2}$ denote the mean weights for 3 groups of subjects, distributed around the population mean. X_0, X_1 represent the observed weights of 2 subjects from group 0, distributed around the group mean. Given the observed subject weights X, the goal is to determine the posterior distributions of the group and population means $p(\theta_{1,0}, \theta_{1,1}, \theta_{1,2}, \theta_{2,0}|X)$.

To infer using the VI framework, we choose a variational family $\mathcal{Q}$ and search inside this family for the distribution closest to our unknown distribution of interest: $q(\Theta) \simeq p(\Theta|X)$. Applying automatic VI to the example in Figure 1, the variational family $\mathcal{Q}$ will oftentimes factorize to (Ambrogioni et al., 2021a;b):

$$q(\Theta; \Phi) = q(\theta_{2,0}; \phi_{2,0}) \prod_{n=0}^{2} q(\theta_{1,n}|\theta_{2,0}; \phi_{1,n}) \tag{1}$$

where $\Phi = \{\phi_{2,0}, \phi_{1,0}, ..., \phi_{1,3}\}$ represent the weights associated to the variational family $\mathcal{Q}$ and each factor in q will approximate the corresponding posterior distribution: as an example $q(\theta_{2,0}; \phi_{2,0}) \simeq p(\theta_{2,0}|X)$.

Compared to the mean-field factorization (Blei et al., 2017), in Equation (1) the posterior for the RVs $\theta_{1,n}$ is conditional to the RV $\theta_{2,0}$ for greater expressivity. We will follow this dependency scheme in Section 4.1. During training, we will search for the optimal weights Φ to best approximate the posterior distribution $p(\Theta|X)$.

Yet, as the number of groups and subjects per group augments, this inference problem becomes computationally intractable. On the parameterization side, each additional group in the population study requires additional weights $\phi_{1,n}$. This constitutes a first computational hurdle: the total number of weights Φ in the variational family becomes prohibitively large. On the computing side, optimizing over an increasing number of weights $\phi_{1,n}$ also requires a growing amount of calculation. This constitutes a second computational hurdle. To circumvent this issue, one must resort to Stochastic VI and only optimize over a subset of the weights Φ at a given training step. In practice, this amounts to inferring the group weight of only a subset of the subject groups at a time. Yet, stochastic training means that the weights $\phi_{1,n}$ corresponding to a given subject group will only be optimized for a fraction of the training time. Consequently, the larger the population study becomes, the slower the inference. Our goal is to keep inference computationally tractable as the cardinality of the problem augments.

To keep the inference tractable, we will build upon the natural factorization of the problem into subjects and groups of subjects. This factorization is symbolized by a model's plates as detailed in Section 3. First, we will harness this natural factorization in the parameterization of our variational family –to reduce its number of weights. This strategy, coined *plate amortization*, will be detailed in Section 4.1. Second, we will reflect this factorization in the design of a stochastic training scheme –to control the computing required during the training. This will be detailed in Section 4.2. Critically, our contribution does not lie only in adding those two items. Combining shared parametrization and stochastic training, PAVI yields orders-of-magnitude speedups in inference. Consequently, contrary to Stochastic VI, inference does not become prohibitively slow as the problem's cardinality augments. In practice, this unlocks inference in very large population studies. We illustrate this speed-up in our experiments in Section 5 and the practical utility of our method in a challenging Neuroimaging setup in Section 5.4.

3 Generative modeling

3.1 Hierarchical Bayesian models (HBMs), templates and plates

Here we define more formally the inference problem and the associated notations.

Population studies can be compactly represented via plate-enriched directed acyclic graph (DAG) templates $\mathcal{T}$ (plates are defined in chapter 8.1 in Bishop, 2006; Gilks et al., 1994; Koller & Friedman, 2009, describes plates and template variables in chapter 6.3). $\mathcal{T}$ feature RV *templates* that symbolise multiple similar *ground* RVs. As an example, the RV template θ_1 in Figure 1 (left) represents a generic group of subjects, of which there would be as many instances as the cardinality of the plate $\mathcal{P}_1$. The plate structure in $\mathcal{T}$ also denotes that the multiple ground RVs corresponding to the same RV template are conditionally i.i.d. As an example, two different group weights could be two Gaussian perturbations of the same population weight. The two groups weight RVs thus have the same conditional distribution, and are independent given the population weight. The plate structure in $\mathcal{T}$ thus symbolizes a strong symmetry in the problem that we exploit in our method.

We denote the template $\mathcal{T}$'s vertices, corresponding to RV templates, as X and $\Theta = \{\theta_i\}_{i=1..I}$. X denotes the RVs observed during inference, and Θ the parameters we infer: our goal is to approximate the posterior distribution $p(\Theta|X)$. We denote $\mathcal{T}$'s plates as $\{\mathcal{P}_p\}_{p=0..P}$, and the plates θ_i belongs to as $\text{Plates}(\theta_i)$. I and P respectively denote the number of latent RV templates and plates in $\mathcal{T}$, which are in general not equal. In the toy example from Figure 1, there are two latent RV templates: θ_1 and θ_2, respectively the group and population mean weights. $\mathcal{T}$ also features two plates $\mathcal{P}_1, \mathcal{P}_0$, which respectively denote groups in the population and the subjects in each group. Graphically, we can see that $\text{Plates}(\theta_2) = \emptyset$, whereas $\text{Plates}(\theta_1) = \{\mathcal{P}_1\}$ and $\text{Plates}(X) = \{\mathcal{P}_0, \mathcal{P}_1\}$. To understand how we could have $P \neq I$, one could keep the plates $\mathcal{P}_1, \mathcal{P}_0$, but add two more RV templates to symbolize the population and group heights, in addition to their weights. In this case, we would have $P = 2$ and $I = 4$.

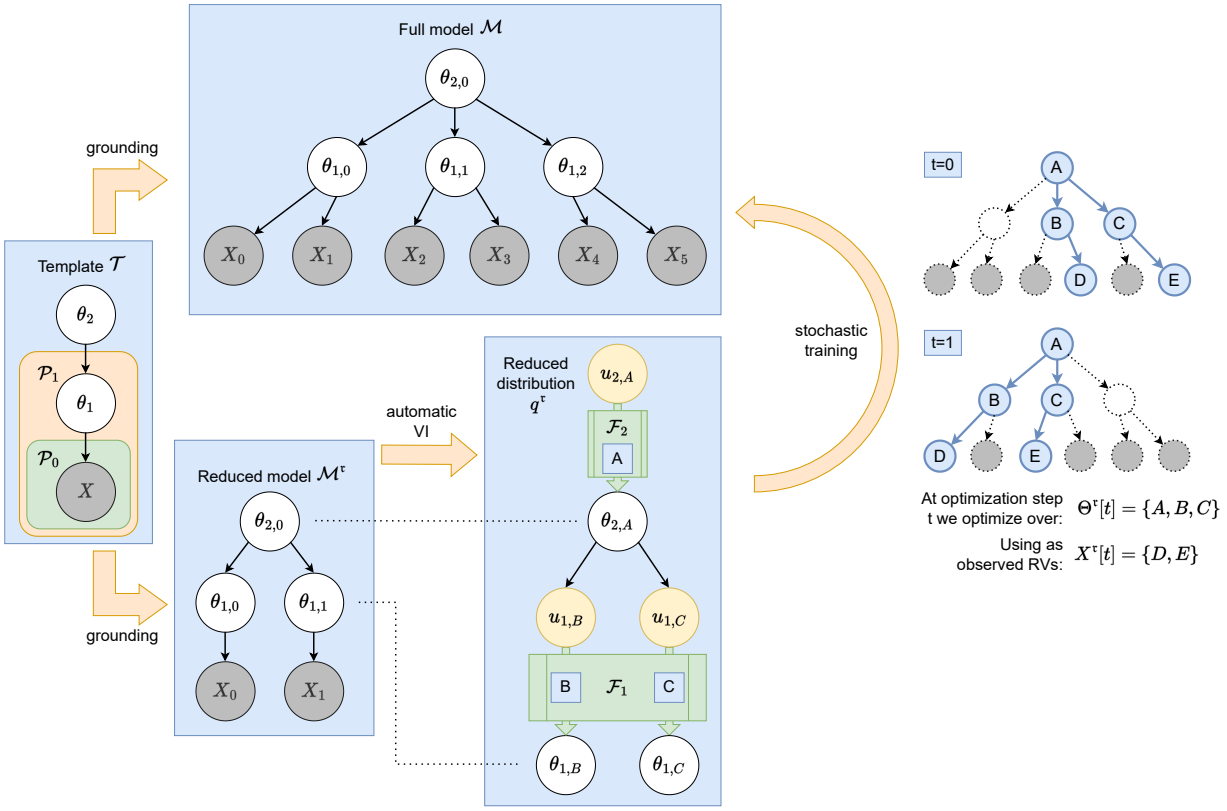


Figure 1: **PAVI working principle:** The template $\mathcal{T}$ (left) can be *grounded* into the *full* model $\mathcal{M}$ (top). We aim to perform inference over $\mathcal{M}$. Yet, $\mathcal{M}$ can feature large cardinalities in population studies. As an example, instead of $\theta_{1,0}, ..., \theta_{1,2}$, $\mathcal{M}$ can feature $\theta_{1,0}, ..., \theta_{1,1000}$ –corresponding to a thousand different subject groups. This can make inference over $\mathcal{M}$ computationally intractable. To circumvent this issue, we will train over $\mathcal{M}$ stochastically. To this end, we instantiate $\mathcal{M}$'s template into a smaller replica: the reduced model $\mathcal{M}^{\mathfrak{r}}$ (bottom left). Following the automatic VI framework, we derive the reduced distribution $q^{\mathfrak{r}}$ (bottom right) directly from $\mathcal{M}^{\mathfrak{r}}$. The reduced distribution $q^{\mathfrak{r}}$ features 2 conditional normalizing flows $\mathcal{F}_1$ and $\mathcal{F}_2$ respectively associated to the RV templates θ_1 and θ_2. During the stochastic training (right), $q^{\mathfrak{r}}$ is instantiated over different branchings of the full model $\mathcal{M}$ –highlighted in blue. The branchings have $\mathcal{M}^{\mathfrak{r}}$'s cardinalities and change at each stochastic training step t. The branching determine the encodings $\mathbf{E}$ conditioning the flows $\mathcal{F}$ –as symbolised by the letters A, B, C– and the observed data slice –as symbolised by the letters D, E.

By instantiating the repeated structures symbolized by the plates $\mathcal{P}$ in $\mathcal{T}$, we obtain a heavier graph representation: the hierarchical Bayesian model (HBM) $\mathcal{M}$. This instantiation is visible in Figure 1, where we go from the template $\mathcal{T}$ (left) to the model $\mathcal{M}$ (top). In the context of Neuroimaging (Kong et al., 2019), $\mathcal{M}$ could feature millions of RVs, corresponding to thousands of subjects and measurements per subject, making it a much less compact representation. We build upon the interplay between the template $\mathcal{T}$ and the HBM $\mathcal{M}$. To go from one representation to the other, $\mathcal{T}$ can be *grounded* into $\mathcal{M}$ given some plate cardinalities $\{\text{Card}(\mathcal{P}_p)\}_{p=0..P}$ (Koller & Friedman, 2009). $\text{Card}(\mathcal{P})$ represents the number of elements in the plate $\mathcal{P}$, for instance the number of groups in the study. Going from $\mathcal{T}$ to $\mathcal{M}$, a RV *template* θ_i is instantiated into multiple *ground* RVs $\{\theta_{i,n}\}_{n=0..N_i}$ with the same parametric form, where $N_i = \prod_{\mathcal{P}\in\text{Plates}(\theta_i)} \text{Card}(\mathcal{P})$. Note that when RV templates belong to multiple plates, they symbolize as many ground RVs as the product of the cardinalities of those plates. In Figure 1, the RV template θ_1 is grounded into the ground RVs $\theta_{1,0}, \theta_{1,1}, \theta_{1,2}$. There are as many ground RVs X_i as the product of the number of groups on the study $\text{Card}(\mathcal{P}_1)$ times the number of subjects per group $\text{Card}(\mathcal{P}_0)$. We denote as $\pi(\theta_{i,n})$ the (potentially empty) set of parents of the RV $\theta_{i,n}$ in the ground graph corresponding to the model $\mathcal{M}$. $\pi(\theta_{i,n})$ are the RVs whose value condition the distribution of $\theta_{i,n}$. For instance, in Figure 1, a subject's weight —the child RV— is a perturbation of the group's weight —the parent RV, conditioning the child RV's distribution. This is denoted as $\pi(X_0) = \{\theta_{1,0}\}$.

The benefit of the template $\mathcal{T}$ and the plates $\mathcal{P}$ is thus to symbolize the natural factorization of the graphical model $\mathcal{M}$ into i.i.d. RVs. This factorization de-clutters the representation of hierarchical models. PAVI exploits this simplifying factorization: we consider the inference over the ground RVs $\theta_{i,n}$ as symmetrical problems, over which we share parameterization and learning, as described in Section 4.

3.2 Full model

The *full* model $\mathcal{M}$ is associated with the density p. In p, the plate structure indicates that a RV template θ_i is associated to a conditional distribution p_i shared across all ground RVs $\theta_{i,n}$:

$$\begin{aligned}\log p(\Theta, X) &= log p(X|\Theta) + \log p(\Theta) \\ &= \sum_{n=0}^{N_X} \log p_X(x_n|\pi(x_n)) + \sum_{i=1}^{I}\sum_{n=0}^{N_i} \log p_i(\theta_{i,n}|\pi(\theta_{i,n}))\end{aligned} \tag{2}$$

where $\pi(\theta_{i,n})$ is the (potentially empty) set of parents of the RV $\theta_{i,n}$, which condition its distribution. We denote with a $\bullet_X$ index all variables related to the observed RVs X. Exploiting the factorization visible in Equation (2), our goal is to obtain a variational distribution $q(\Theta)$ usable to approximate the unknown posterior $p(\Theta|X)$ for the target model $\mathcal{M}$.

4 Methods

4.1 PAVI architecture

Full variational family design Here we define the variational distribution q corresponding to the full model $\mathcal{M}$. To derive q, we push forward the prior $p(\Theta)$ using trainable normalizing flows, denoted as $\mathcal{F}$ (Papamakarios et al., 2019). A normalizing flow is an invertible and differentiable neural network, also known as a parametric diffeomorphism. A normalizing flow transforms samples from a simple base distribution into samples from an arbitrarily complex target distribution. To every ground RV $\theta_{i,n}$, we associate the learnable flow $\mathcal{F}_{i,n}$ to approximate its posterior distribution:

$$\begin{aligned}\log q(\Theta) &= \sum_{i=1}^{I}\sum_{n=0}^{N_i} \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n})) \\ \theta_{i,n} &= \mathcal{F}_{i,n}(u_{i,n}) \\ u_{i,n} &\sim p_i(u_{i,n}|\pi(\theta_{i,n}))\end{aligned} \tag{3}$$

where $q_{i,n}$ is the push-forward of the prior p_i through the flow $\mathcal{F}_{i,n}$. This push-forward is illustrated in Figure 1, where flows $\mathcal{F}$ push RVs u into θ. Since it replicates the prior conditional dependencies, $q(\Theta)$ in

Equation (3) mimics the right-hand term $p(\Theta)$ in Equation (2). This *cascading* scheme was first introduced by Ambrogioni et al. (2021b). In the 2 plates case, this dependency scheme is equivalent to the branch approximation coined by Agrawal & Domke (2021). In this shallow case, Agrawal & Domke (2021) show that modeling only the prior dependencies does not limit the expressivity of the variational family compared to a fully joint factorization. However, this result does not generalize to cases featuring more plates, as detailed in Appendix C.5.

Focus on normalizing flows Expressivity is essential to make VI experimentally viable: the variational $q(\Theta)$ should be a good proxy for the true $p(\Theta|X)$ and not an artifact due to $q(\Theta)$'s limited expressivity (Blei et al., 2017). Though PAVI could be applied to simpler density estimators —such as the Gaussian approximations explored in depth by Agrawal & Domke (2021)— we focus on normalizing flows which can model a large range of distributions. As detailed in the next paragraph, PAVI allows the experimenter to always default to the most expressive flow architecture possible —such as Masked Autoregressive Flows (Papamakarios et al., 2018)— without sacrificing the variational family's scalability. This helps the experimenter minimize the inference's approximation gap (Cremer et al., 2018). Nonetheless, in cases where the experimenter has a good sense of the form the terms $q_{i,n}$ should take in the posterior —due for instance to conjugacy— the usage of very expressive flows $\mathcal{F}_{i,n}$ can be relaxed. Optimizing a lower number of weights in $\mathcal{F}_{i,n}$ can then further improve the inference speed.

Though maximizing q's expressivity, using fully parameterized flows would not scale to the context of large hierarchical models —featuring millions of RVs. This justifies the need for plate amortization, as detailed in the next paragraph.

Plate amortization Here, we introduce plate amortization: sharing the parameterization of density estimators across a model's plates. Plate amortization reduces the number of weights in a variational family as the cardinality of the inference problem augments. In Section 4.2, we show that plate amortization also results in faster inference.

VI searches for a distribution $q(\Theta;\phi)$ that best approximates the posterior of Θ given a value $\mathbf{X}_0$ for X: $q(\Theta;\phi_0) \simeq p(\Theta|X=\mathbf{X}_0)$. When presented with a new data point $\mathbf{X}_1$, optimization has to be performed again to search for the weights ϕ_1, such that $q(\Theta;\phi_1) \simeq p(\Theta|X=\mathbf{X}_1)$. Instead, *sample amortized* (sa) inference (Zhang et al., 2019; Cremer et al., 2018) infers in the general case, regressing the weights ϕ using an *encoder* f of the observed data $\mathbf{X}$: $q(\Theta;\phi=f(\mathbf{X}_i)) \simeq p(\Theta|X=\mathbf{X}_i)$. The cost of learning the encoder weights is *amortized* since inference over any new sample $\mathbf{X}$ requires no additional optimization. In Section 5.3, we compare PAVI to several *sample* amortized baselines, which will be denoted by the suffix *(sa)*, and should not be confused with *plate* amortized methods. A more in-depth discussion about amortization can be found in Appendix C.1. We propose to exploit the concept of amortization but to apply it at a different granularity, leading to our notion of *plate amortization*.

Similar to amortizing across the different data samples $\mathbf{X}$, we amortize across the different ground RVs $\{\theta_{i,n}\}_{n=0..N_i}$ corresponding to the same RV template θ_i. Instead of casting every flow $\mathcal{F}_{i,n}$, defined in Equation (3), as a separate, fully-parameterized flow, we will share some parameters across the $\mathcal{F}_{i,n}$. To the template θ_i, we associate a conditional flow $\mathcal{F}_i(\,\cdot\,;\phi_i,\bullet)$ with weights ϕ_i shared across all the $\{\theta_{i,n}\}_{n=0..N_i}$. The flow $\mathcal{F}_{i,n}$ associated with a given ground RV $\theta_{i,n}$ will be an instance of this conditional flow, conditioned by an encoding $\mathbf{E}_{i,n}$:

$$\mathcal{F}_{i,n} = \mathcal{F}_i(\,\cdot\,;\phi_i,\mathbf{E}_{i,n}) \quad \text{yielding} \quad q_{i,n} = q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\phi_i,\mathbf{E}_{i,n}) \tag{4}$$

The distributions $q_{i,n}$ thus have 2 sets of weights, ϕ_i and $\mathbf{E}_{i,n}$, creating a parameterization trade-off. Concentrating all of $q_{i,n}$'s parameterization into ϕ_i results in all the ground RVs $\theta_{i,n}$ having the same posterior distribution. On the contrary, concentrating all of $q_{i,n}$'s parameterization into $\mathbf{E}_{i,n}$ allows the $\theta_{i,n}$ to have completely different posterior distributions. But in a large cardinality setting, this freedom can result in a massive number of weights, proportional to the number of ground RVs times the encoding size. This double parameterization is therefore efficient when the majority of the weights of $q_{i,n}$ is concentrated into ϕ_i. Using normalizing flows $\mathcal{F}_i$, the burden of approximating the correct parametric form for the posterior is placed onto ϕ_i, while the $\mathbf{E}_{i,n}$ encode lightweight summary statistics specific to each $\theta_{i,n}$. For instance,

$\mathcal{F}_i$ could learn to model a Gaussian mixture distribution, while the $\mathbf{E}_{i,n}$ would encode the location and variance of each mode for each ground RV. Encodings $\mathbf{E}_{i,n}$ allow to individualize for $\theta_{i,n}$ only the strictly necessary information necessary to approximate $\theta_{i,n}$'s posterior. A more in-depth analysis of the impact of plate amortization on the expressivity of the variational family $\mathcal{Q}$ can be found in Appendix A.4.

Intermediate summary This section defined q, the variational distribution to approximate full model $\mathcal{M}$'s posterior. q features *plate amortization*, which helps maintain a tractable number of weights as the cardinality of $\mathcal{M}$ augments. The next section introduces a stochastic scheme to train q. Critically, combining shared parameterization and stochastic training does not simply result in combining the advantages of both, but features synergies resulting in faster training, as further explained in Section 4.2.

4.2 PAVI stochastic training

Reduced model Our goal is to train the variational distribution $q(\Theta)$, defined in Equation (3), that approximates the posterior $p(\Theta|X)$. q corresponds to the *full* model $\mathcal{M}$. $\mathcal{M}$ typically features large plate cardinalities $\text{Card}(\mathcal{P})$ —with many subject groups and subjects per group— and thus many ground RVs, making it computationally intractable. We will therefore train q stochastically, over smaller subsets of RVs at a time. In this section, we interpret stochastic training as the training over a *reduced* model $\mathcal{M}^{\mathfrak{r}}$.

Instead of inferring directly over $\mathcal{M}$, we will train over a smaller replica of $\mathcal{M}$. To this end, we instantiate the template $\mathcal{T}$ into a second HBM $\mathcal{M}^{\mathfrak{r}}$, the *reduced* model, of tractable plate cardinalities $\text{Card}^{\mathfrak{r}}(\mathcal{P}) \ll \text{Card}(\mathcal{P})$. $\mathcal{M}^{\mathfrak{r}}$ has the same template as $\mathcal{M}$, meaning the same dependency structure and the same parametric form for its distributions. The only difference lies in $\mathcal{M}^{\mathfrak{r}}$'s smaller cardinalities, resulting in fewer ground RVs, as visible in Figure 1.

Reduced distribution and loss Here we define the distribution $q^{\mathfrak{r}}$ used in stochastic training. $q^{\mathfrak{r}}$ features the smaller cardinalities of the reduced model $\mathcal{M}^{\mathfrak{r}}$, making it computationally tractable. At each optimization step t, we randomly choose inside $\mathcal{M}$ paths of reduced cardinality, as visible in Figure 1. Selecting paths is equivalent to selecting from X a subset $X^{\mathfrak{r}}[t]$ of size $N_X^{\mathfrak{r}}$, and from Θ a subset $\Theta^{\mathfrak{r}}[t]$. For a given θ_i, we denote as $\mathcal{B}_i[t]$ the batch of selected ground RVs, of size $N_i^{\mathfrak{r}}$. $\mathcal{B}_X[t]$ equivalently denotes the batch of selected observed RVs. Inferring over $\Theta^{\mathfrak{r}}[t]$, we will simulate training over the full distribution q:

$$\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t]) = \sum_{i=1}^{I} \frac{N_i}{N_i^{\mathfrak{r}}} \sum_{n \in \mathcal{B}_i[t]} \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n})) \tag{5}$$

where the factor $N_i/N_i^{\mathfrak{r}}$ emulates the observation of as many ground RVs as in $\mathcal{M}$ by repeating the RVs from $\mathcal{M}^{\mathfrak{r}}$ (Hoffman et al., 2013). Similarly, the loss used at step t is the reduced ELBO constructed using $X^{\mathfrak{r}}[t]$ as observed RVs:

$$\begin{aligned} \text{ELBO}^{\mathfrak{r}}[t] &= \mathbb{E}_{\Theta^{\mathfrak{r}} \sim q^{\mathfrak{r}}} \left[\log p^{\mathfrak{r}}(X^{\mathfrak{r}}[t], \Theta^{\mathfrak{r}}[t]) - \log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] \\ \log p^{\mathfrak{r}}(X^{\mathfrak{r}}[t], \Theta^{\mathfrak{r}}[t]) &= \frac{N_X}{N_X^{\mathfrak{r}}} \sum_{n \in \mathcal{B}_X[t]} \log p_X(x_n|\pi(x_n)) + \sum_{i=1}^{I} \frac{N_i}{N_i^{\mathfrak{r}}} \sum_{n \in \mathcal{B}_i[t]} \log p_i(\theta_{i,n}|\pi(\theta_{i,n})) \end{aligned} \tag{6}$$

This scheme can be viewed as the instantiation of $\mathcal{M}^{\mathfrak{r}}$ over batches of $\mathcal{M}$'s ground RVs. In Figure 1, we see that $q^{\mathfrak{r}}$ has the cardinalities of $\mathcal{M}^{\mathfrak{r}}$ and replicates its conditional dependencies. This training is analogous to stochastic VI (Hoffman et al., 2013), generalized with multiple hierarchies, dependencies in the posterior, and mini-batches of RVs.

Sharing learning across plates Here we detail how our shared parameterization, detailed in Section 4.1, combined with our stochastic training scheme, results in faster inference. In stochastic VI (SVI, Hoffman et al., 2013), every $\theta_{i,n}$ corresponding to the same template θ_i is associated with individual weights. Those weights are trained only when the algorithm visits $\theta_{i,n}$, that is to say, at step t when $n \in \mathcal{B}_i[t]$. As plates become larger, this event becomes rare. If $\theta_{i,n}$ is furthermore associated with a highly-parameterized density estimator —such as a normalizing flow— many optimization steps are required for $q_{i,n}$ to converge. The

combination of those two items leads to slow training and makes inference impractical in contexts such as Neuroimaging, which can feature millions of RVs. With plate amortization, we aim to unlock inference in those large regimes by reducing the training time.

Instead of treating the ground RVs $\theta_{i,n}$ independently, we share the learning across plates. Due to the problem's plate structure, we consider the inference over the $\theta_{i,n}$ as different instances of a common density estimation task. In PAVI, a large part of the parameterization of the estimators $q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}); \phi_i, \mathbf{E}_{i,n})$ is mutualized via the plate-wide-shared weights ϕ_i. This means that most of the weights of the flows $\mathcal{F}_{i,n}$, concentrated in ϕ_i, are trained at every optimization step across all the selected batches $\mathcal{B}_i[t]$. This results in drastically faster convergence than SVI, as seen in experiment 5.1.

4.3 Encoding schemes

PAVI-F and PAVI-E schemes PAVI shares the parameterization and learning of density estimators across an HBM's plates. In practice the distributions $q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}); \phi_i, \mathbf{E}_{i,n})$ from Equation (4) with different n only differ through the value of the encodings $\mathbf{E}_{i,n}$. We detail two schemes to derive those encodings:

Free plate encodings (PAVI-F) In our core implementation, $\mathbf{E}_{i,n}$ are free weights. We define encoding arrays with the cardinality of the full model $\mathcal{M}$, one array $\mathbf{E}_i = [\mathbf{E}_{i,n}]_{n=0..N_i}$ per template θ_i. This means that an additional ground RV —for instance, adding a subject in a population study— requires an additional encoding vector. The associated increment in the total number of weights is much lighter than adding a fully parameterized normalizing flow, as would be the case in the non-plate-amortized regime. The PAVI-F scheme cannot be sample amortized: when presented with an unseen $\mathbf{X}$, though ϕ_i can be kept as an efficient warm start, the optimal values for the encodings $\mathbf{E}_{i,n}$ have to be searched again.

During training, the encodings $\mathbf{E}_{i,n}$ corresponding to $n \in \mathcal{B}_i[t]$ are sliced from the arrays $\mathbf{E}_i$ and are optimized for along with ϕ_i. In the toy example from Figure 1, at $t = 0$, $\mathcal{B}_1[0] = \{1, 2\}$ and the trained encodings are $\{\mathbf{E}_{1,1}, \mathbf{E}_{1,2}\}$, and at $t = 1$ $\mathcal{B}_1[1] = \{0, 1\}$ and we train $\{\mathbf{E}_{1,0}, \mathbf{E}_{1,1}\}$.

Deep set encoder (PAVI-E) The parameterization of PAVI-F scales lightly but linearly with $\text{Card}(\mathcal{P})$. Though lighter than the non-plate-amortized case, this scaling could still become unaffordable in large population studies. We thus propose an alternate scheme, PAVI-E, with a parameterization independent of cardinalities. In this more experimental scheme, free encodings are replaced by an encoder f with weights η applied to the observed data: $\mathbf{E} = f(\mathbf{X}; \eta)$. As encoder f we use a *deep-set* architecture, detailed in Appendix A.1.3 (Zaheer et al., 2018; Lee et al., 2019). Due to the plate structure, the observed X features multiple permutation invariances —across data points corresponding to i.i.d. RVs. Deep sets are attention architectures that can model generic permutation invariant functions. As such, they constitute a natural design choice to incorporate the problem's invariances. The PAVI-E scheme allows for *sample amortization* across different data samples $\mathbf{X}_0, \mathbf{X}_1, ...$, as described in Section 4.1. Note that an encoder will be used to generate the encodings whether the inference is sample amortized or not. Our encoder architecture is similar to the one from Agrawal & Domke (2021), considering only the case of fixed-cardinality plates, and generalizing to an arbitrary number of plates.

During training, shared learning is further amplified as all the architecture's weights —ϕ_i and η— are trained at every step t. To collect the encodings to plug into $q^{\mathfrak{r}}$, we build up on a property of f: *set size generalization* (Zaheer et al., 2018). Instead of encoding the full-sized data $\mathbf{X}$, f is applied to the slice $\mathbf{X}^{\mathfrak{r}}[t]$. This amounts to aggregating summary statistics across a subset of the observed data instead of the full data (Lee et al., 2019; Agrawal & Domke, 2021). The PAVI-E scheme has an even greater potential in the sample amortized context: we train a sample amortized family over the lightweight model $\mathcal{M}^{\mathfrak{r}}$ and use it "for free" to infer over the heavyweight model $\mathcal{M}$.

Stochastic training and bias A key consideration is a potential bias introduced by our stochastic scheme. Namely, training stochastically over a variational family $\mathcal{Q}$, we want to converge to the same solution q^* as if we trained over the entirety of $\mathcal{Q}$. In this section, we show that the PAVI-F scheme is unbiased. In contrast, the PAVI-E scheme is theoretically biased —though we seldom noticed any negative impact of that bias in practice. A more thorough analysis can be found in Appendix A.2. Note that in this manuscript the term

bias systematically refers to the stochastic training scheme. A different form of bias consists in the limited expressivity of the variational family $\mathcal{Q}$ which may not contain the true posterior $p(\Theta|X)$. We refer to this other bias as the variational family's *gap*, as further detailed in Appendix A.4.

To show that our training scheme is unbiased, we need to prove that the expectation of our stochastic loss is equal to the non-stochastic loss. This amounts to showing that:

$$\begin{aligned} \mathbb{E}_{\text{paths}}\left[\text{ELBO}^{\mathfrak{r}}[t]\right] &= \text{ELBO} \\ &= \mathbb{E}_{\Theta\sim q}\left[\log p(X,\Theta) - \log q(\Theta)\right] \end{aligned} \tag{7}$$

where $\mathbb{E}_{\text{paths}}$ loosely refers to the paths that are sampled stochastically into the full model $\mathcal{M}$'s graph, as defined in Section 4.2. Equation (7) means that the expectation of the reduced ELBO over all the stochastic paths that can be sampled inside the full model's graph is equal to the full ELBO. To prove Equation (7), it is sufficient to prove that:

$$\mathbb{E}_{\text{paths}}\left[\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] = \log q(\Theta) \tag{8}$$

meaning that the expectation of the reduced distribution $q^{\mathfrak{r}}$ over the stochastic paths equals the full distribution q. In Appendix A.2 we show that:

$$\mathbb{E}_{\text{paths}}\left[\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] = \sum_{i=1}^{I}\sum_{n=0}^{N_i}\mathbb{E}_{\text{paths}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\mathbf{E}_{i,n})\right] \tag{9}$$

an expression that mimics the definition of q in Equation (3) —albeit the expectations over paths. The rest of the bias analysis depends on the encoding scheme.

Free plate encodings (PAVI-F) In the PAVI-F scheme, the encodings $\mathbf{E}_{i,n}$ are free weights. As a consequence, their value does not depend on the paths that are selected by the stochastic algorithm. This means that $\mathbb{E}_{\text{paths}}\left[\log q_{i,n}(\theta_{i,n};\mathbf{E}_{i,n})\right] = \log q_{i,n}(\theta_{i,n};\mathbf{E}_{i,n})$, proving Equation (8) and the unbiasedness of the PAVI-F scheme. With the PAVI-F scheme, training over $\mathcal{M}^{\mathfrak{r}}$, we converge to the same distribution as if training directly over $\mathcal{M}$.

Deep set encoder (PAVI-E) In the PAVI-E scheme, encodings are obtained by applying the encoder f to a slice of the observed data. With the implementation presented in this work, the value of $\mathbf{E}_{i,n}$ will depend on which children of $\theta_{i,n}$ the stochastic algorithm selects. For instance, the encoding for a group will depend on which subjects are selected inside the group to compute the group's summary statistics. Though computationally efficient, our implementation is thus theoretically biased. The negative impact of this bias on PAVI-E's performance was seldom noticeable and always marginal throughout our experiments.

Technical summary In Section 4.1, we derived an architecture sharing its parameterization across a model's plates. This allows performing inference in large cardinality regimes without incurring an exploding number of weights. In Section 4.2 we derived a stochastic scheme to train this architecture over batches of data. Stochastic training allows large-scale inference without incurring an exploding memory and computing. Off-the-shelf stochastic VI would however result in significantly slower inference. Our novelty lies in our original combination of amortization and stochastic training, which avoids this slow-down, as demonstrated in the following experiments.

5 Results and discussion

In this section, we show how PAVI unlocks hierarchical Bayesian model inference for large-scale problems by matching the inference quality of SOTA methods while providing faster convergence and lighter parameterization. Our experiments also highlight the differences between our two encoding schemes PAVI-E and PAVI-F. In summary:

1. Section 5.1 shows how plate amortization results in faster convergence compared to non-plate-amortized SVI;
2. Section 5.2 illustrates the role of the encodings $\mathbf{E}_{i,n}$ as summary statistics in inference;

3. Section 5.3 exemplifies –against baselines– PAVI's favorable scaling as a problem's cardinality augments;

4. Section 5.4 showcases the practical utility of PAVI by applying the method to a challenging population study.

Supplemental experiments In Appendix B.3, we evaluate the impact of the reduced model cardinalities on performance. In Appendix B.4, we compare our method against baselines over a mixture model; over a model featuring the aggregation of higher-order summary statistics; and over a smaller version of our Neuroimaging model used in Section 5.4. Those models complement the analysis of our main text, which is mostly concentrated on Gaussian models. Gaussian models constitute a standard model featuring an approximate closed-form posterior which we use for validation.

ELBO metric Throughout this section, we use the ELBO as a proxy for the KL divergence between the variational posterior and the unknown true posterior (Blei et al., 2017). ELBO is measured across 20 samples $\mathbf{X}$, with 5 repetitions per sample. The ELBO allows us to compare the relative performance of different architectures on a given inference problem. In Appendix B.2, we provide sanity checks to assess the quality of the results.

Gaussian random effects (GRE) model In our experiments Section 5.1, 5.2, 5.3, we focus on a standard hierarchical model: a Gaussian random effects model (GRE)(Diggle et al., 2013; Gelman et al., 2004). The GRE model constitutes an intuitive inference problem —inferring group and population means given observations— and features an approximate closed-form solution, as described in the next paragraph. This closed-form solution helps us display theoretical bounds for the ELBO, for instance in Figure 2.

The GRE model can be described using the following set of equations:

$$\begin{aligned} X_{n_1,n_0}|\theta_{1,n_1} &\sim \mathcal{N}(\theta_{1,n_1}, \sigma_x^2) \quad {}^{\forall n_1=1..\,\mathrm{Card}(\mathcal{P}_1)}_{\forall n_0=1..\,\mathrm{Card}(\mathcal{P}_0)} \\ \theta_{1,n_1}|\theta_{2,0} &\sim \mathcal{N}(\theta_{2,0}, \sigma_1^2) \quad {}^{\forall n_1=1..\,\mathrm{Card}(\mathcal{P}_1)} \qquad \theta_{2,0} \sim \mathcal{N}(\vec{0}_D, \sigma_2^2) \ , \end{aligned} \tag{10}$$

where D represents the data $\mathbf{X}$'s feature size, with group means θ_1 and population means θ_2 as D-dimensional Gaussians. The GRE model features two nested plates: the group plate $\mathcal{P}_1$ and the sample plate $\mathcal{P}_0$ as in Figure 1. Taking our introductory example from Section 3, X_{n_1,n_0} represents the weight for subject n_0 in group n_1. θ_{1,n_1} represents the mean weight in the group n_1. $\theta_{2,0}$ represents the mean weight in the population. Inferring over the GRE model, the objective is to retrieve the posterior distribution of the group and population means given the observed sample.

Asymptotic closed-form ELBO As a baseline for comparison, we compare the ELBO of various methods to an approximate closed-form baseline's ELBO. Though a closed-form posterior cannot be derived in the 3-level case, a good approximation can be constructed using Gaussian distributions centered on the empirical group and population means. This asymptotic ELBO is represented using dashed lines in Figures 2 and 3.

5.1 Plate amortization and convergence speed

In this experiment, we illustrate how plate amortization results in faster training.

We use the GRE model, described in Equation (10), setting $D = 8$, $\mathrm{Card}(\mathcal{P}_1) = 100$ and $\mathrm{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 2$. In this experiment, we set $\mathrm{Card}^{\mathfrak{r}}(\mathcal{P}_1) \ll \mathrm{Card}(\mathcal{P}_1)$. This emulates a regime in which SVI is slow because only a small fraction —of size $\mathrm{Card}^{\mathfrak{r}}(\mathcal{P}_1)$— of a large parameter space —of size $\mathrm{Card}(\mathcal{P}_1)$— gets optimized at a given stochastic training step. We compare our PAVI architecture to a baseline with the same architecture, trained stochastically with SVI (Hoffman et al., 2013), but without plate amortization. The only difference is that ground RVs $\theta_{i,n}$ are associated in the baseline to individual fully-parameterized flows $\mathcal{F}_{i,n}$ instead of sharing the same conditional flow $\mathcal{F}_i$, as further described in Section 4.1.

Figure 2 (left) displays the evolution of the ELBO across training steps for the baseline and PAVI with free encoding (PAVI-F) and deep-set encoders (PAVI-E). We see that both plate-amortized methods reach

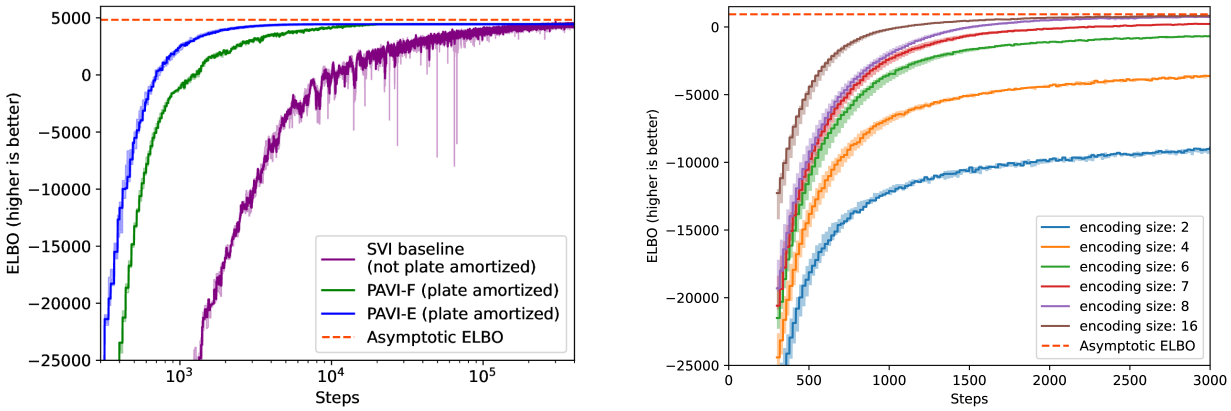


Figure 2: **Left panel: Plate amortization increases convergence speed** Plot of the ELBO (higher is better) as a function of the optimization steps (log-scale) for our methods PAVI-F (in green) and PAVI-E (in blue) versus a non-plate-amortized baseline (in purple). Standard deviation across repetitions is displayed as a shaded area. A dashed line denotes the asymptotic closed-form performance. Due to plate amortization, our method converges ten to a hundred times faster to the same asymptotic ELBO as its non-plate-amortized counterpart.; **Right panel: Encodings as ground RVs summary statistics** Plot of the ELBO (higher is better) as a function of the optimization steps for the PAVI-F architecture with increasing encoding sizes. Standard deviation across repetitions is displayed as a shaded area. A dashed line denotes the asymptotic closed-form performance. As the encoding size augments, so does the asymptotic performance until reaching the dimensionality of the posterior's sufficient statistics ($D = 8$), after which performance plateaus. Encoding size allows for a clear trade-off between memory and inference quality.

asymptotic ELBO equal to the non-plate-amortized baseline's but with orders of magnitudes faster convergence and more numerical stability. This stems from the individual flows $\mathcal{F}_{i,n}$ in the baseline only being trained when the stochastic algorithm visits the corresponding $\theta_{i,n}$. In contrast, our shared flow $\mathcal{F}_i$ is updated at every optimization step in PAVI. Intuitively, the PAVI-E scheme should converge faster than PAVI-F by sharing the training not only of the conditional flows but also of the encoder across the different optimization steps. However, the computation required to derive the encodings from the observed data results in longer optimization steps and in slower inference, as illustrated in Section 5.3. The usage of an encoder nonetheless allows for *sample amortization* with the PAVI-E scheme, which is impossible with the PAVI-F scheme.

5.2 Impact of encoding size

Here we illustrate the role of encodings as ground RV posterior's summary statistics, as further described in Section 4.1. We use the GRE HBM detailed in Equation (10), using $D = 8$, $\text{Card}(\mathcal{P}_1) = 20$ and $\text{Card}^{\mathsf{r}}(\mathcal{P}_1) = 2$. We use a single PAVI-F architecture, varying the size of the encodings $\mathbf{E}_{i,n}$, which are defined in Section 4.3.

Due to plate amortization, encodings determine how much individual information each RV $\theta_{i,n}$'s posterior is associated with. The encoding size —varying from 2 to 16— is to be compared with the problem's dimensionality, $D = 8$. In GRE, $D = 8$ corresponds to the size of the sufficient statistics needed to reconstruct the posterior of a group mean since all other statistics, such as the variance, are shared between the group means.

Figure 2 (right) shows how the asymptotic performance steadily increases when the encoding size augments and plateaus once reaching the sufficient summary statistic size $D = 8$. Interestingly, increasing the encoding size also leads to faster convergence: redundancy can likely be exploited in the optimization. Increasing the encoding size also leads experimentally to diminishing returns in terms of performance. This property can be exploited in large settings to drastically reduce the memory footprint of inference while maintaining acceptable performance by choosing the encoding size approximately equal to the expected size of the sufficient statistics. Encoding size appears as an unequivocal hyperparameter allowing to trade inference quality for computational efficiency.

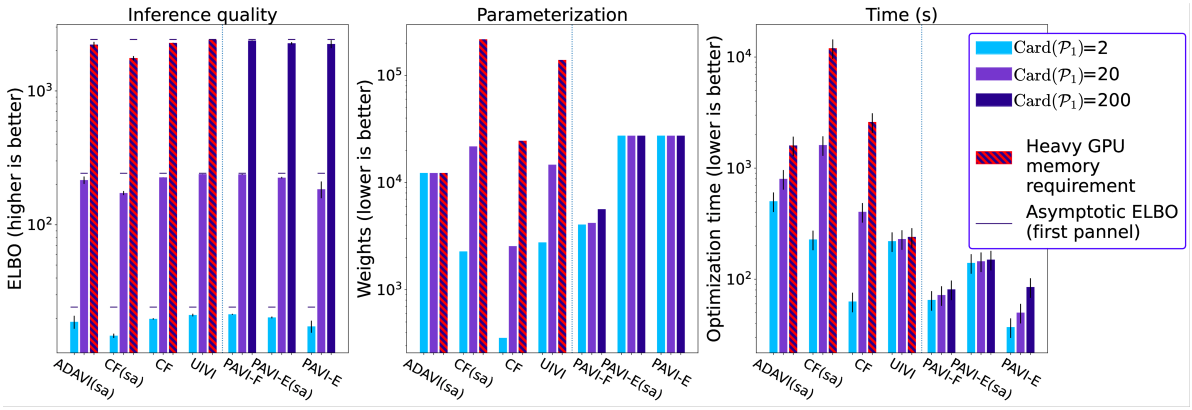


Figure 3: **PAVI provides favorable parameterization and training time as the cardinality of the target model augments** Baselines are compared in each panel, with the suffix (sa) indicating *sample amortization*. Our architecture PAVI is displayed on the right of each panel. We augment the cardinality $\text{Card}(\mathcal{P}_1)$ of the GRE model, which is described in Equation (10). While doing so, we compare three different metrics. *In the first panel:* inference quality, as measured by the ELBO. An asymptotic closed-form ELBO is displayed using a dark blue dash. None of the presented SOTA architecture's performance degrades as the cardinality of the problem augments. *In the second panel:* parameterization, comparing the number of trainable weights of each architecture. PAVI –similar to ADAVI– displays a constant number of weights as the cardinality of the problem increases —or almost constant for PAVI-F. *Third panel:* GPU training time. Benefiting from learning across plates, PAVI has a short and almost constant training time as the cardinality of the problem augments. At $\text{Card}(\mathcal{P}_1) = 200$, CF, UIVI, and ADAVI required large GPU memory, a constraint absent from PAVI due to its stochastic training. For convenience, the results used to plot this figure are reproduced in supplemental Table B.1.

5.3 Scaling with plate cardinalities

Here we put in perspective the gains from plate amortization when scaling up an inference problem's cardinality. We consider the GRE model in Equation (10) with $D = 2$ and augment the plate cardinalities $(\text{Card}(\mathcal{P}_1), \text{Card}^{\mathfrak{r}}(\mathcal{P}_1)) : \ (2, 1) \to (20, 5) \to (200, 20)$. In doing so, we augment the number of parameters $\Theta : \ 6 \to 42 \to 402$.

Baselines We compare our PAVI architecture against three state-of-the-art automatic VI baselines.

- **Cascading Flows (CF)** (Ambrogioni et al., 2021b) is a non-plate-amortized structured VI architecture improving on the baseline presented in Section 5.1. CF pushes the prior p into the posterior q using *Highway Flows*. CF follows a cascading dependency structure complemented by a backward auxiliary coupling. CF thus consists in a structured baseline that does not pay particular attention to scalability to large cardinalities;

- **Automatic Dual Amortized VI (ADAVI)** (Rouillard & Wassermann, 2022) is a structured VI architecture with constant parameterization with respect to a problem's cardinality but large training times and memory. ADAVI has several limitations compared to PAVI. First, ADAVI implements a Mean Field approximation (Blei et al., 2017) while PAVI implements a cascading flow. This means that, contrary to PAVI, ADAVI can be biased in cases where the posterior features strong dependencies. Second, ADAVI is limited to pyramidal HBMs, while PAVI tackles generic plate-enriched HBMs, making it more general. Third, ADAVI is limited to a full-model sample-amortized variant, which ultimately caps the cardinality of problems it can be used on. ADAVI thus consists in a structured baseline that tackles the parameterization hurdle of large cardinalities, but not the computational efficiency hurdle;

- **Unbiased Implicit VI (UIVI)** is an unstructured implicit VI architecture. UIVI infers over the full parameter space Θ, without any SVI-amenable factorization —contrary to CF, ADAVI, and

PAVI. To do so, UIVI reparameterizes a base distribution with a stochastic transform. UIVI does not explicitly define a density q and relies on optimization steps intertwined with MCMC runs. UIVI thus consists in a non-structured VI baseline that does not pay particular attention to scalability to a large parameter space This means that UIVI could not be applied above a certain cardinality due to its impossibility to be stochastically trained.

For all architectures, we indicate with the suffix *(sa) sample amortization*, as defined in Section 4.1. More implementation details can be found in Appendix B.5.

As the cardinality of the problem augments, Figure 3 shows how PAVI maintains a state-of-the-art inference quality while being more computationally attractive.

Parameterization In terms of parameterization, both ADAVI and PAVI-E provide a heavyweight but constant parameterization as the cardinality $\text{Card}(\mathcal{P}_1)$ of the problem augments. This is due to both methods' usage of an encoder of the observed data, which makes their parameterization independent of the problem's cardinality. Comparatively, both CF and PAVI-F's parameterization scale linearly with $\text{Card}(\mathcal{P}_1)$, but with a drastically lighter augmentation for PAVI-F. The difference can be explained by the additional weights required by each architecture for an additional ground RV. CF requires an additional fully parameterized normalizing flow, whereas PAVI-F only requires an additional lightweight encoding vector. In detail, PAVI-F's parameterization due to the plate-wide-shared ϕ_1 represents a constant $\approx 2k$ weights, while the part due to the encodings $\mathbf{E}_{1,n}$ grows linearly from 16 to 160 to $1.6k$ weights. UIVI's parameterization scales quadratically with the size of the parameter space Θ. This is due to UIVI's usage of a neural network to regress the weights of a transform applied to a base distribution with the size of Θ. As the cardinality augments, UIVI's quadratic weight scaling would be limiting before CF and PAVI-F's linear scaling, which would be limiting before ADAVI and PAVI-E's constant scaling.

Memory Regarding computational budget, PAVI's stochastic training allows for controlled GPU memory during optimization. This removes the need for a larger memory as the cardinality of the problem augments, a hardware constraint that can become unaffordable at very large cardinalities. CF could be trained stochastically to remove this memory constraint but, without plate amortization, would suffer from slower inference, as illustrated in Section 5.1. In contrast, UIVI could not be trained stochastically, as it infers over the full parameter space Θ at once instead of factorizing it. As a result, UIVI would be ultimately limited by memory to infer over larger problems.

Speed Regarding convergence speed, PAVI benefits from plate amortization to have orders of magnitude faster convergence compared to structured VI baselines CF and ADAVI. This means that a stochastically-trained architecture (PAVI) trains faster than non-stochastic baselines (CF, ADAVI) This result is opposite to the result we would have obtained without plate amortization since SVI slows down inference. For UIVI, as the cardinality augments, training amounts to evaluating a neural network with increasingly larger layers. GPU training time is thus constant, but this property would not translate to larger problems as the GPU memory would become insufficient. Plate amortization is particularly significant for the PAVI-E(sa) scheme, in which a sample-amortized variational family is trained over a dataset of reduced cardinality yet performs "for free" inference over an HBM of large cardinality. Maintaining $\text{Card}^{\mathbf{r}}(\mathcal{P}_1)$ constant while $\text{Card}(\mathcal{P}_1)$ augments allows for a constant parameterization *and training time* as the cardinality of the problem augments This property is not limited to any maximum cardinality, contrary to UIVI. This is a novel result with strong future potential. The effect of plate amortization is particularly noticeable at $\text{Card}(\mathcal{P}_1) = 200$ between the PAVI-E(sa) and CF(sa) architectures, where PAVI performs sample-amortized inference with $10\times$ fewer weights and $100\times$ lower training time.

Scaling even higher the cardinality of the problem —$\text{Card}(\mathcal{P}_1) = 2000$ for instance— renders ADAVI, CF and UIVI computationally intractable In contrast, PAVI maintains a light memory footprint and a short training time, as exemplified in the next experiment.

Using PAVI on small scale problems By design, PAVI is meant to tackle large cardinality problems effectively. The use of PAVI on smaller problems is thus an open question. As Figure 3 underlines, in the $\text{Card}(\mathcal{P}_1) = 2$ case, PAVI-F matches the ELBO and training speed of CF but features a heavier parameterization. In addition, plate amortizing a variational family theoretically reduces its expressivity, as further

explained in Appendix A.4. In theory, a simpler, non-plate-amortized baseline such as CF could thus be preferred for small problems. However, in practice, VI amounts to an optimization problem, and the most expressive variational family will not necessarily yield the best performance on a complex problem (Bottou & Bousquet, 2007). Our comparative experiments in Appendix B.4 exemplify this. In Appendix B.4, PAVI's repeated parameterization oftentimes eases the optimization and yields a better ELBO. We leave for future work the analysis of the impact of PAVI's repeated parameterization —which could be considered as an inductive bias— on properties such as inference calibration (Talts et al., 2020). In conclusion, when its parameterization is affordable, PAVI remains a relevant choice for small plate-enriched problems.

5.4 Neuroimaging application: full cortex probabilistic parcellation for a cohort of 1,000 subjects

To illustrate its usefulness, we apply PAVI to a challenging population study for full-cortex functional *parcellation*.

A *parcellation* clusters brain vertices into different *connectivity networks*: fingerprints describing co-activation with the rest of the brain, as measured using functional Magnetic Resonance Imaging (fMRI). A parcellation is essentially a clustering of a subject's cortex vertices based on their connectivity with the rest of the brain. Different subjects exhibit a strong variability, as visible in Figure 4. However, fMRI is costly to acquire: few noisy measurements are usually made for a given subject. It is thus essential to combine information across the different measurements for a given subject and across subjects and to display uncertainty in the results. Those 2 points motivate hierarchical Bayesian modelling and VI in Neuroimaging (Kong et al., 2019): we search the posteriors of connectivity networks and vertex labels, measuring fMRI over a large cohort of subjects.

We use the HCP dataset (Van Essen et al., 2012): 2 acquisitions from a 1000 subjects, with millions of measures per acquisition, and *over 400 million parameters* Θ to infer. We use a model with three plates: subjects, sessions and brain vertices. We cluster the brain vertices into 7 clusters, following Thomas Yeo et al. (2011). None of the baselines presented in Section 5.3 —CF, ADAVI, UIVI— can computationally tackle this high cardinality problem. We nevertheless show superior performance over those baselines over a tractable problem size with 2,000 parameters in Appendix B.4.3.

Despite the massive dimensionality of the problem, thanks to plate amortization, PAVI converges in less than 5 hours of GPU time. Results are visible in Figure 4. Qualitatively, the recovered networks match existing expert knowledge on the brain's functional organization. For instance, we recover networks responsible for motor control or vision (Heim et al., 2009; Zhang et al., 2020). Quantitatively, using the individual subject's parcellation as feature for a cognitive score regression task, we obtain better scores compared to state-of-the-art methods (Thomas Yeo et al., 2011; Gordon et al., 2017; Wang et al., 2015; Kong et al., 2019). This means that a subject's brain spatial organization —where each function is "located" in the brain— can be used to predict the subject's cognitive ability, such as memory, reading, or spatial orientation.

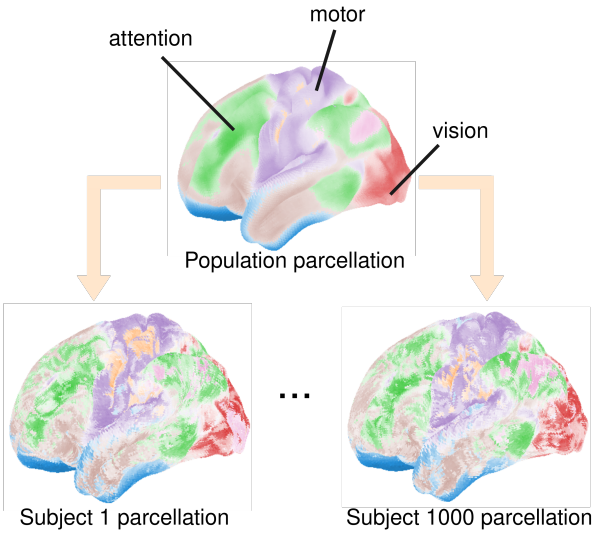
Using PAVI, we obtain state-of-the-art functional parcellations for a thousand subjects, capturing the uncertainty in this challenging task via VI while maintaining a manageable training time despite a 400-million-sized parameter space.

5.5 Conclusion

In this work we present the novel PAVI architecture, combining a structured variational family and a stochastic training scheme. PAVI is based the on concept of plate amortization, allowing to share parameterization and learning across a model's plates. We demonstrated the positive impact of plate amortization on training speed and scaling to large plate cardinality regimes, making a significant step towards scalable, expressive Variational Inference.

Reproducibility statement

All experiments were performed in Python using the Tensorflow Probability library (Dillon et al., 2017). All experiments were conducted on computational cluster nodes equipped with a Tesla V100-16Gb GPU and 4 AMD EPYC 7742 64-Core processors. VRAM intensive experiments in Figure 3 were performed on an



Method	Accuracy for 13 cognitive measures (Higher is better)
MS-HBM from Kong et al. (2019)	0.1321 (±0.0053)
YeoBackProject from Calhoun & Adali (2012)	0.1057 (±0.0060)
Gordon2017 from Gordon et al. (2017)	0.0545 (±0.0062)
Wang2015 from Wang et al. (2015)	0.1202 (±0.0054)
Ours (PAVI)	**0.1645** (±0.0047)

Figure 4: **Probabilistic full cortex parcellation** PAVI can be applied in the challenging context of Neuroimaging population studies. For a cohort of 1000 subjects, 2 of which are represented here (in the bottom 2 items) we cluster 60,000 cortex vertices according to their connectivity with the rest of the brain. On the left, we show the obtained probabilistic *parcellations*. Each color in the parcellation corresponds to one of 7 functional *network* (Thomas Yeo et al., 2011). Networks represent groups of neurons that co-activate in the brain and can be associated with certain cognitive functions, such as vision or motor control. Through our method, we recover each subject's parcellation (at the bottom), which are i.i.d. perturbations of the population's parcellation (at the top). Our method also models uncertainty: coloring represents the dominant label for each vertex, and the level of white increases with the uncertainty in the labeling. We can use the subject parcellations as features for a cognitive score prediction task, as visible in the table on the right. The table shows the mean predictive accuracy across 13 cognitive measures, including memory, pronunciation, processing speed or spatial orientation. The baseline methods scores are reproduced from Kong et al. (2019)'s implementation. Our method produces individual maps that are predictive of the subject's cognitive ability.

Ampere 100 PCIE-40Gb GPU. Appendix B.5 lists implementation details for all our synthetic experiments. Appendix B.6 is related to our Neuroimaging experiment 5.4, and details both our data pre-processing steps and our implementation. As part of our submission we furthermore packaged and release the code associated to our experiments.

Broader Impact Statement

Our Neuroimaging data come from the Human Connectome Project dataset (Van Essen et al., 2012). All data in the HCP is strongly anonymized, as per the HCP protocols. We used in this paper only Open Access imaging data data, following the HCP data use terms.

Acknowledgments

This work was supported by the ERC-StG NeuroLang ID:757672.

This work was performed using HPC resources from GENCI–IDRIS (Grant 2022-102043).

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Mueller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gael Varoquaux. Machine learning for neuroimaging with scikit-learn. *Frontiers in Neuroinformatics*, 8, 2014. ISSN 1662-5196. doi: 10.3389/fninf.2014.00014. URL https://www.frontiersin.org/article/10.3389/fninf.2014.00014.

Abhinav Agrawal and Justin Domke. Amortized Variational Inference for Simple Hierarchical Models. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 21388–21399. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/b28d7c6b6aec04f5525b453411ff4336-Paper.pdf.

Luca Ambrogioni, Kate Lin, Emily Fertig, Sharad Vikram, Max Hinne, Dave Moore, and Marcel van Gerven. Automatic structured variational inference. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 676–684. PMLR, 13–15 Apr 2021a. URL https://proceedings.mlr.press/v130/ambrogioni21a.html.

Luca Ambrogioni, Gianluigi Silvestri, and Marcel van Gerven. Automatic variational inference with cascading flows. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 254–263. PMLR, 18–24 Jul 2021b. URL https://proceedings.mlr.press/v139/ambrogioni21a.html.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

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, April 2017. ISSN 0162-1459, 1537-274X. doi: 10.1080/01621459.2017.1285773. URL http://arxiv.org/abs/1601.00670. arXiv: 1601.00670.

Léon Bottou and Olivier Bousquet. The Tradeoffs of Large Scale Learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper/2007/file/0d3180d672e08b4c5312dcdafdf6ef36-Paper.pdf.

Vince D. Calhoun and Tülay Adali. Multisubject Independent Component Analysis of fMRI: A Decade of Intrinsic Networks, Default Mode, and Neurodiagnostic Discovery. *IEEE Reviews in Biomedical Engineering*, 5:60–73, 2012. ISSN 1937-3333, 1941-1189. doi: 10.1109/RBME.2012.2211076. URL http://ieeexplore.ieee.org/document/6268324/.

Chris Cremer, Xuechen Li, and David Duvenaud. Inference Suboptimality in Variational Autoencoders. *arXiv:1801.03558 [cs, stat]*, May 2018. URL http://arxiv.org/abs/1801.03558. arXiv: 1801.03558.

Kamalaker Dadi, Gaël Varoquaux, Antonia Machlouzarides-Shalit, Krzysztof J. Gorgolewski, Demian Wassermann, Bertrand Thirion, and Arthur Mensch. Fine-grain atlases of functional modes for fMRI analysis. *NeuroImage*, 221:117126, 2020. ISSN 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2020.117126. URL https://www.sciencedirect.com/science/article/pii/S1053811920306121.

Viet-Hung Dao, David Gunawan, Minh-Ngoc Tran, Robert Kohn, Guy E. Hawkins, and Scott D. Brown. Efficient selection between hierarchical cognitive models: Cross-validation with variational bayes, 2021. URL https://arxiv.org/abs/2102.06814.

Peter Diggle, Patrick Heagerty, Kung-Yee Liang, and Scott Zeger. *Analysis of Longitudinal Data*. OUP Oxford, March 2013. ISBN 978-0-19-967675-0. Google-Books-ID: ur0BlXPuOukC.

Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A. Saurous. TensorFlow Distributions. *arXiv:1711.10604 [cs, stat]*, November 2017. URL http://arxiv.org/abs/1711.10604. arXiv: 1711.10604.

Yann Dubois, Jonathan Gordon, and Andrew YK Foong. Neural process family. http://yanndubs.github.io/Neural-Process-Family/, September 2020.

A Fayaz, P Croft, RM Langford, LJ Donaldson, and GT Jones. Prevalence of chronic pain in the uk: a systematic review and meta-analysis of population studies. *BMJ open*, 6(6):e010364, 2016.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes, 2018. URL https://arxiv.org/abs/1807.01622.

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition, 2004.

W. R. Gilks, A. Thomas, and D. J. Spiegelhalter. A Language and Program for Complex Bayesian Modelling. *The Statistician*, 43(1):169, 1994. ISSN 00390526. doi: 10.2307/2348941. URL https://www.jstor.org/stable/10.2307/2348941?origin=crossref.

Evan M. Gordon, Timothy O. Laumann, Adrian W. Gilmore, Dillan J. Newbold, Deanna J. Greene, Jeffrey J. Berg, Mario Ortega, Catherine Hoyt-Drazen, Caterina Gratton, Haoxin Sun, Jacqueline M. Hampton, Rebecca S. Coalson, Annie L. Nguyen, Kathleen B. McDermott, Joshua S. Shimony, Abraham Z. Snyder, Bradley L. Schlaggar, Steven E. Petersen, Steven M. Nelson, and Nico U. F. Dosenbach. Precision Functional Mapping of Individual Human Brains. *Neuron*, 95(4):791–807.e7, 2017. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2017.07.011. URL https://www.sciencedirect.com/science/article/pii/S089662731730613X.

Stefan Heim, Simon B. Eickhoff, Anja K. Ischebeck, Angela D. Friederici, Klaas E. Stephan, and Katrin Amunts. Effective connectivity of the left BA 44, BA 45, and inferior temporal gyrus during lexical and phonological decisions identified with DCM. *Human Brain Mapping*, 30(2):392–402, February 2009. ISSN 10659471. doi: 10.1002/hbm.20512.

Matthew D. Hoffman and David M. Blei. Structured Stochastic Variational Inference. *arXiv:1404.4114 [cs]*, November 2014. URL http://arxiv.org/abs/1404.4114. arXiv: 1404.4114.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(4):1303–1347, 2013. URL http://jmlr.org/papers/v14/hoffman13a.html.

Ekaterina Iakovleva, Jakob Verbeek, and Karteek Alahari. Meta-learning with shared amortized variational inference. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4572–4582. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/iakovleva20a.html.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. Adaptive computation and machine learning. MIT Press, Cambridge, MA, 2009. ISBN 978-0-262-01319-2.

Ru Kong, Jingwei Li, Csaba Orban, Mert Rory Sabuncu, Hesheng Liu, Alexander Schaefer, Nanbo Sun, Xi-Nian Zuo, Avram J. Holmes, Simon B. Eickhoff, and B. T. Thomas Yeo. Spatial topography of individual-specific cortical networks predicts human cognition, personality, and emotion. *Cerebral cortex*, 29 6:2533–2551, 2019.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M. Blei. Automatic Differentiation Variational Inference. *arXiv:1603.00788 [cs, stat]*, March 2016. URL http://arxiv.org/abs/1603.00788. arXiv: 1603.00788.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/lee19d.html.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *CoRR*, abs/1611.00712, 2016. URL http://arxiv.org/abs/1611.00712.

George Papamakarios and Iain Murray. Fast ϵ-free Inference of Simulation Models with Bayesian Conditional Density Estimation. *arXiv:1605.06376 [cs, stat]*, April 2018. URL http://arxiv.org/abs/1605.06376. arXiv: 1605.06376.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation. *arXiv:1705.07057 [cs, stat]*, June 2018. URL http://arxiv.org/abs/1705.07057. arXiv: 1705.07057.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing Flows for Probabilistic Modeling and Inference. *arXiv:1912.02762 [cs, stat]*, December 2019. URL http://arxiv.org/abs/1912.02762. arXiv: 1912.02762.

Sachin Ravi and Alex Beatson. Amortized bayesian meta-learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rkgpy3C5tX.

Louis Rouillard and Demian Wassermann. ADAVI: Automatic Dual Amortized Variational Inference Applied To Pyramidal Bayesian Models. In *ICLR 2022*, Virtual, France, April 2022. URL https://hal.archives-ouvertes.fr/hal-03267956.

Francisco Ruiz and Michalis Titsias. A contrastive divergence for combining variational inference and MCMC. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5537–5545. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/ruiz19a.html.

Hojjat Salehinejad, Julianne Baarbe, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *CoRR*, abs/1801.01078, 2018. URL http://arxiv.org/abs/1801.01078.

Stephen M Smith, Christian F Beckmann, Jesper Andersson, Edward J Auerbach, Janine Bijsterbosch, Gwenaëlle Douaud, Eugene Duff, David A Feinberg, Ludovica Griffanti, Michael P Harms, et al. Resting-state fmri in the human connectome project. *Neuroimage*, 80:144–168, 2013.

Sean Talts, Michael Betancourt, Daniel Simpson, Aki Vehtari, and Andrew Gelman. Validating Bayesian Inference Algorithms with Simulation-Based Calibration, October 2020. URL http://arxiv.org/abs/1804.06788. arXiv:1804.06788 [stat].

B. T. Thomas Yeo, Fenna M. Krienen, Jorge Sepulcre, Mert R. Sabuncu, Danial Lashkari, Marisa Hollinshead, Joshua L. Roffman, Jordan W. Smoller, Lilla Zöllei, Jonathan R. Polimeni, Bruce Fischl, Hesheng Liu, and Randy L. Buckner. The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *Journal of Neurophysiology*, 106(3):1125–1165, 2011. doi: 10.1152/jn.00338.2011. URL https://doi.org/10.1152/jn.00338.2011. PMID: 21653723.

Michalis K. Titsias and Francisco Ruiz. Unbiased implicit variational inference. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 167–176. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/titsias19a.html.

Kayle Towsley, Michael I Shevell, Lynn Dagenais, Repacq Consortium, et al. Population-based study of neuroimaging findings in children with cerebral palsy. *European journal of paediatric neurology*, 15(1): 29–35, 2011.

Dustin Tran, Rajesh Ranganath, and David M. Blei. Hierarchical Implicit Models and Likelihood-Free Variational Inference. *arXiv:1702.08896 [cs, stat]*, November 2017. URL http://arxiv.org/abs/1702.08896. arXiv: 1702.08896.

D. C. Van Essen, K. Ugurbil, E. Auerbach, D. Barch, T. E. Behrens, R. Bucholz, A. Chang, L. Chen, M. Corbetta, S. W. Curtiss, S. Della Penna, D. Feinberg, M. F. Glasser, N. Harel, A. C. Heath, L. Larson-Prior, D. Marcus, G. Michalareas, S. Moeller, R. Oostenveld, S. E. Petersen, F. Prior, B. L. Schlaggar, S. M. Smith, A. Z. Snyder, J. Xu, and E. Yacoub. The Human Connectome Project: a data acquisition perspective. *Neuroimage*, 62(4):2222–2231, Oct 2012.

Danhong Wang, Randy L Buckner, Michael D Fox, Daphne J Holt, Avram J Holmes, Sophia Stoecklein, Georg Langs, Ruiqi Pan, Tianyi Qian, Kuncheng Li, Justin T Baker, Steven M Stufflebeam, Kai Wang, Xiaomin Wang, Bo Hong, and Hesheng Liu. Parcellating cortical functional networks in individuals. *Nature Neuroscience*, 18(12):1853–1860, December 2015. ISSN 1097-6256, 1546-1726. doi: 10.1038/nn.4164. URL http://www.nature.com/articles/nn.4164.

Stefan Webb, Adam Golinski, Robert Zinkov, N. Siddharth, Tom Rainforth, Yee Whye Teh, and Frank Wood. Faithful Inversion of Generative Models for Effective Amortized Inference. *arXiv:1712.00287 [cs, stat]*, November 2018. URL http://arxiv.org/abs/1712.00287. arXiv: 1712.00287.

Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 3(1):9, May 2016. ISSN 2196-1115. doi: 10.1186/s40537-016-0043-6. URL https://doi.org/10.1186/s40537-016-0043-6.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2020. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2020.2978386. URL http://arxiv.org/abs/1901.00596. arXiv: 1901.00596.

Huaxiu Yao, Ying Wei, Junzhou Huang, and Zhenhui Li. Hierarchically structured meta-learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7045–7054. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/yao19b.html.

Mingzhang Yin and Mingyuan Zhou. Semi-implicit variational inference. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5660–5669. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/yin18b.html.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep Sets. *arXiv:1703.06114 [cs, stat]*, April 2018. URL http://arxiv.org/abs/1703.06114. arXiv: 1703.06114.

Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. Advances in Variational Inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):2008–2026, August 2019. ISSN 0162-8828, 2160-9292, 1939-3539. doi: 10.1109/TPAMI.2018.2889774. URL https://ieeexplore.ieee.org/document/8588399/.

Yizhen Zhang, Kuan Han, Robert Worth, and Zhongming Liu. Connecting concepts in the brain by mapping cortical representations of semantic relations. *Nature Communications*, 11(1):1877, April 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-15804-w.

Appendix

A Supplemental methods

A.1 PAVI implementation details

A.1.1 Plate branchings and stochastic training

As exposed in Section 4.2, at each optimization step t we randomly select branchings inside the full model $\mathcal{M}$, branchings over which we *instantiate* the reduced model $\mathcal{M}^{\mathfrak{r}}$. In doing so, we define batches $\mathcal{B}_i[t]$ for the RV templates θ_i. Those batches have to be coherent with one another: they have to respect the conditional dependencies of the original model $\mathcal{M}$. As an example, if a ground RV is selected as part of $\mathcal{M}^{\mathfrak{r}}$, then its parent RV needs to be selected as well. To ensure this, during the stochastic training we do not sample RVs directly but plates:

1. For every plate $\mathcal{P}_p$, we sample without replacement $\text{Card}^{\mathfrak{r}}(\mathcal{P}_p)$ indices amongst the $\text{Card}(\mathcal{P}_p)$ possible indices.
2. Then, for every RV template θ_i, we select the ground RVs $\theta_{i,n}$ corresponding to the sampled indices for the plates $\text{Plates}(\theta_i)$.
3. The selected ground RVs $\theta_{i,n}$ constitute the set $\Theta^{\mathfrak{r}}[t]$ of parameters appearing in Equation (5). The same procedure yields the observed RV subset $X^{\mathfrak{r}}[t]$ and the data slice $\mathbf{X}^{\mathfrak{r}}[t]$.

For instance, in the toy example from Figure 1, X_2 will be chosen iff the index 1 is selected as part of sub-sampling $\mathcal{P}_1$ and the index 0 is selected as part of sub-sampling $\mathcal{P}_0$. Less formally, this is equivalent to going *middle*, then *left* in the full graph representing $\mathcal{M}$. This stochastic choice is illustrated in Figure 1 at $t = 1$ where X_2 corresponds to the node E. This stochastic strategy also applies to the selected encoding scheme —described in Section 4.3&4.2— as detailed in the next sections.

A.1.2 PAVI-F details

In Section 4.3 we refer to encodings $\mathbf{E}_i = [\mathbf{E}_{i,n}]_{n=0..N}$ corresponding to RV templates θ_i. In practice, we have some amount of sharing for those encodings: instead of defining separate encodings for every RV template, we define encodings for every *plate level*. A plate level is a combination of plates with at least one parameter RV template θ_i belonging to it:

$$\text{PlateLevels} = \{(\mathcal{P}_k..\mathcal{P}_l) = \text{Plates}(\theta_i)\}_{\theta_i \in \Theta} \tag{A.1}$$

For every plate level, we construct a large encoding array with the cardinalities of the full model $\mathcal{M}$:

$$\begin{aligned} \text{Encodings} &= \{(\mathcal{P}_k..\mathcal{P}_l) \mapsto \mathbb{R}^{\text{Card}(\mathcal{P}_k)\times..\times\text{Card}(\mathcal{P}_l)\times D}\}_{(\mathcal{P}_k..\mathcal{P}_l)\in\text{PlateLevels}} \\ \mathbf{E}_i &= \text{Encodings}(\text{Plates}(\theta_i)) \end{aligned} \tag{A.2}$$

Where D is an encoding size that we kept constant to de-clutter the notation but can vary between plate levels. The encodings for a given ground RV $\theta_{i,n}$ then correspond to an element from the encoding array $\mathbf{E}_i$.

A.1.3 PAVI-E details

In the PAVI-E scheme, encodings are not free weights but the output of en encoder $f(\,\cdot\,, \eta)$ applied to the observed data $\mathbf{X}$. In this section we detail the design of this encoder.

As in the previous section, the role of the encoder will be to produce one encoding per plate level. We start from a dependency structure for the plate levels:

$$\begin{aligned} &\forall (\mathcal{P}_a..\mathcal{P}_b) \in \text{PlateLevels}\ , \\ &\forall (\mathcal{P}_c..\mathcal{P}_d) \in \text{PlateLevels}\ , \\ (\mathcal{P}_a..\mathcal{P}_b) \in \pi((\mathcal{P}_c..\mathcal{P}_d)) &\Leftrightarrow {}^{\exists \theta_i /\, \text{Plates}(\theta_i)=(\mathcal{P}_a..\mathcal{P}_b)}_{\exists \theta_j /\, \text{Plates}(\theta_j)=(\mathcal{P}_c..\mathcal{P}_d)} / \theta_j \in \pi(\theta_i) \end{aligned} \tag{A.3}$$

note that this dependency structure is in the *backward* direction: a plate level will be the parent of another plate level, if the former contains a RV who has a child in the latter. We therefore obtain a plate level dependency structure that *reverts* the conditional dependency structure of the graph template $\mathcal{T}$. To avoid redundant paths in this dependency structure, we take the maximum branching of the obtained graph.

Given the plate level dependency structure, we will recursively construct the encodings, starting from the observed data:

$$\forall x \in X : \quad \text{Encodings}(\text{Plates}(x)) = \rho(\mathbf{x}) \tag{A.4}$$

where $\mathbf{x}$ is the observed data for the RV x, and ρ is a simple encoder that processes every observed ground RV's value independently through an identical multi-layer perceptron. Then, until we have exhausted all plate levels, we process existing encodings to produce new encodings:

$$\begin{aligned} &\forall (\mathcal{P}_k..\mathcal{P}_l) \in \text{PlateLevels} \,/\, \nexists x \in X, \text{Plates}(x) = (\mathcal{P}_k..\mathcal{P}_l) : \\ &\text{Encodings}((\mathcal{P}_k..\mathcal{P}_l)) = g(\text{Encodings}(\pi(\mathcal{P}_k..\mathcal{P}_l))) \end{aligned} \tag{A.5}$$

where g is the composition of attention-based deep-set networks called *Set Transformers* (Lee et al., 2019; Zaheer et al., 2018). For every plate $\mathcal{P}_p$ present in the parent plate level but absent in the child plate level, g will compute summary statistics *across* that plate, effectively contracting the corresponding batch dimensionality in the parent encoding (Rouillard & Wassermann, 2022).

In the case of multiple observed RVs, we run this "backward pass" independently for each observed data —with one encoder per observed RV. We then concatenate the resulting encodings corresponding to the same plate level.

For more precise implementation details, we invite the reader to consult the codebase released with this supplemental material.

A.2 Analysis of bias in the stochastic training

A key concern in our stochastic training scheme is its unbiasedness: we want our stochastic optimization to converge to the same variational posterior as if we trained over the full model directly —without any stochasticity. In this section we first show that the PAVI-F is unbiased. Second, we identify strategies to obtain an unbiased PAVI-E scheme, while showing how the approximations we do in practice can theoretically result in biased training. As an important note, the negative impact of this bias on the performance PAVI-E remained limited throughout our experiments —as seen in experiments 5.3, B.3, B.2 and B.4.

A.2.1 General derivation (applicable to both the PAVI-F and PAVI-E schemes)

We first formalize the *plate sampling* strategy described in Appendix A.1.1. To every plate $\mathcal{P}$ we associate the RV $I_{\mathcal{P}}$ corresponding to the $\text{Card}^{\mathfrak{r}}(\mathcal{P})$-sized set of indices sampled without replacement from the $\text{Card}(\mathcal{P})$ possible index values. As an example, with a plate $\mathcal{P}_0$ with $\text{Card}(\mathcal{P}_0) = 4$ and $\text{Card}^{\mathfrak{r}}(\mathcal{P}_0) = 2$, $\{0, 2\}$ or $\{2, 3\}$ can be 2 different samples from $I_{\mathcal{P}_0}$. At a given optimization step t, we sample independently from the RVs $\{I_{\mathcal{P}_p}\}_{p=0..P}$. This defines the batches $\mathcal{B}_i[t]$ and the distribution $q^{\mathfrak{r}}$ in Equation (5).

To check the unbiasedness of our stochastic training, we need to show that:

$$\mathbb{E}_{I_{\mathcal{P}_0}} \dots \mathbb{E}_{I_{\mathcal{P}_P}} \left[\text{ELBO}^{\mathfrak{r}}[t]\right] = \text{ELBO} \tag{A.6}$$

Where:

$$\text{ELBO} = \mathbb{E}_{\Theta \sim q} \left[\log p(X, \Theta) - \log q(\Theta)\right] \tag{A.7}$$

And $\text{ELBO}^{\mathfrak{r}}[t]$ is defined in Equation (6). In that expression, q and p have symmetrical roles. As the ELBO amounts to the difference between the logarithms of distributions p and q, we can prove the equality in Equation (A.6) if we prove that the expectation of each reduced distribution is equal to the corresponding full distribution. To prove the equality in Equation (A.6), a sufficient condition is therefore to prove that:

$$\mathbb{E}_{I_{\mathcal{P}_0}} \dots \mathbb{E}_{I_{\mathcal{P}_P}} \left[\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] = \mathbb{E}_{I_{\mathcal{P}}} \left[\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] = \log q(\Theta) \tag{A.8}$$

where to de-clutter the notations we denote the expectation over the collection of RVs $\{I_{\mathcal{P}_p}\}_{p=0..P}$ as $\mathbb{E}_{I_\mathcal{P}}$.

Consider a given ground RV $\theta_{i,n}$ corresponding to the RV template θ_i and to the plates $\text{Plates}(\theta_i)$. At a given stochastic step t, $\theta_{i,n}$ will be chosen if and only if its corresponding *branching* is chosen. Recall that when sampling equiprobably without replacement a set of k elements from a population of n elements, a given element will be present in the set with probability k/n. We can apply this reasoning to the choice of *branching* corresponding to a given ground RV. For instance, in Figure 1, X_2 will be chosen iff the index 1 is selected as part of sub-sampling $\mathcal{P}_1$ and the index 0 is selected as part of sub-sampling $\mathcal{P}_0$. As $\text{Card}^\mathfrak{r}(\mathcal{P}_1) = 2$ indices are chosen inside the plate $\mathcal{P}_1$ of full cardinality 3, and $\text{Card}^\mathfrak{r}(\mathcal{P}_0) = 1$ indices are chosen inside the plate $\mathcal{P}_0$ of full cardinality 2, X_2 is therefore chosen with probability $2/3 \times 1/2$. More formally, for a ground RV $\theta_{i,n}$ we have:

$$\begin{aligned} \forall n = 0..N_i \ : \ \mathbb{P}(\theta_{i,n} \in \mathcal{B}_i[t]) &= \prod_{\mathcal{P} \in \text{Plates}(\theta_i)} \frac{\text{Card}^\mathfrak{r}(\mathcal{P})}{\text{Card}(\mathcal{P})} \\ &= \frac{N_i^\mathfrak{r}}{N_i} \end{aligned} \tag{A.9}$$

Applying this reasoning to every RV template θ_i, we have that:

$$\begin{aligned} \mathbb{E}_{I_\mathcal{P}}\left[\log q^\mathfrak{r}(\Theta^\mathfrak{r}[t])\right] &= \sum_{i=1}^{I} \frac{N_i}{N_i^\mathfrak{r}} \mathbb{E}_{I_\mathcal{P}}\left[\sum_{n \in \mathcal{B}_i[t]} \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}))\right] \\ &= \sum_{i=1}^{I} \frac{N_i}{N_i^\mathfrak{r}} \mathbb{E}_{I_\mathcal{P}}\left[\sum_{n=0}^{N_i} \mathbb{1}_{n \in \mathcal{B}_i[t]} \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}))\right] \\ &= \sum_{i=1}^{I} \frac{N_i}{N_i^\mathfrak{r}} \sum_{n=0}^{N_i} \mathbb{E}_{I_\mathcal{P}}\left[\mathbb{1}_{n \in \mathcal{B}_i[t]} \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}); \phi_i, \mathbf{E}_{i,n})\right] \end{aligned} \tag{A.10}$$

where we exploited the fact that the expectation of the sum of RVs is the sum of the expectations, even in the case of dependent RVs. The term $\mathbb{1}_{n \in \mathcal{B}_i[t]} \times \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}); \phi_i, \mathbf{E}_{i,n})$ is the product of 2 RVs —related to the stochastic choice of plate indices:

- the RV $\mathbb{1}_{n \in \mathcal{B}_i[t]}$ is an indicator that $\theta_{i,n}$'s *branching* has been chosen via the stochastic sampling of plate indices. By construction, this RV depends only on the indices of the plates $\mathcal{P} \in \text{Plates}(\theta_i)$.
- the RV $\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}); \phi_i, \mathbf{E}_{i,n})$ depends on $\mathbf{E}_{i,n}$, whose construction depends on the encoding scheme:
  - In the PAVI-F scheme, $\mathbf{E}_{i,n}$ is a constant.
  - In the PAVI-E scheme, $\mathbf{E}_{i,n}$ results of the application of an encoder to the observed data of a subset of $\theta_{i,n}$'s descendants. By construction, this subset will only depend on the indices of plates containing θ_i's descendants, but not containing θ_i. The value of $\mathbf{E}_{i,n}$ therefore only depends on the indices of plates $\mathcal{P} \notin \text{Plates}(\theta_i)$

As an example of this reasoning, consider the model $\mathcal{M}$ illustrated in Figure 1. We can evaluate both terms for the ground RV $\theta_{1,2}$ in the PAVI-E scheme:

- $\mathbb{1}_{2 \in \mathcal{B}_1[t]}$ depends on whether the index 2 is chosen as part of sub-sampling the plate $\mathcal{P}_1$, and therefore only depends on the RV $I_{\mathcal{P}_1}$. In this case, the associated probability is 2/3;
- to evaluate $\log q_{1,2}(\theta_{1,2}|\theta_{2,0}; \phi_1, \mathbf{E}_{1,2})$, the value of $\mathbf{E}_{1,2}$ will result from the application of the encoder f over the value of either X_4 or X_5. This choice depends on whether the index 0 or 1 is chosen as part of sub-sampling the plate $\mathcal{P}_0$. Therefore, the value of the term $\log q_{1,2}$ only depends on the RV $I_{\mathcal{P}_0}$.

As a result, in both PAVI-F and PAVI-E, the terms $\mathbb{1}_{n\in\mathcal{B}_i[t]}$ and $\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\phi_i,\mathbf{E}_{i,n})$ depend on the sampled indices of disjoint sets of plates, and are therefore independent. This means that the expectation of their product can be rewritten as the product of their expectations:

$$\begin{aligned}\mathbb{E}_{I_\mathcal{P}}\left[\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] &= \sum_{i=1}^{I}\frac{N_i}{N_i^{\mathfrak{r}}}\sum_{n=0}^{N_i}\mathbb{E}_{I_\mathcal{P}}\left[\mathbb{1}_{n\in\mathcal{B}_i[t]}\right]\mathbb{E}_{I_\mathcal{P}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}))\right]\\ &= \sum_{i=1}^{I}\frac{N_i}{N_i^{\mathfrak{r}}}\sum_{n=0}^{N_i}\frac{N_i^{\mathfrak{r}}}{N_i}\mathbb{E}_{I_\mathcal{P}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n}))\right]\\ &= \sum_{i=1}^{I}\sum_{n=0}^{N_i}\mathbb{E}_{I_\mathcal{P}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\mathbf{E}_{i,n})\right]\end{aligned} \tag{A.11}$$

This equality can be further simplified in the PAVI-F case —proving its unbiasedness— but not in the PAVI-E case, as detailed in the sections below.

A.2.2 Unbiasedness of the PAVI-F scheme

In the PAVI-F scheme, detailed in Section 4.3, the encodings $\mathbf{E}_{i,n}$ are constants with respect to the branching choice, therefore we have:

$$\begin{aligned}\mathbb{E}_{I_\mathcal{P}}\left[\log q^{\mathfrak{r}}(\Theta^{\mathfrak{r}}[t])\right] &= \sum_{i=1}^{I}\sum_{n=0}^{N_i}\mathbb{E}_{I_\mathcal{P}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\mathbf{E}_{i,n})\right]\\ &= \sum_{i=1}^{I}\sum_{n=0}^{N_i}\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\mathbf{E}_{i,n})\\ &= \log q(\Theta)\end{aligned} \tag{A.12}$$

which proves Equation (A.8) and Equation (A.6). In the above example of $\theta_{1,2}$ in $\mathcal{M}$, in the PAVI-F scheme the expression $\mathbb{E}_{I_\mathcal{P}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\phi_i,\mathbf{E}_{i,n})\right]$ can be evaluated into $\log q_{1,2}(\theta_{1,2}|\theta_{2,0};\phi_1,\mathbf{E}_{1,2})$. This demonstrates that the PAVI-F scheme is unbiased: training over stochastically chosen sub-graphs for $q^{\mathfrak{r}}$ is in expectation equal to training over the full graph of q.

A.2.3 Approximations in the PAVI-E scheme

In the PAVI-E scheme, detailed in Section 4.2, the encodings $\mathbf{E}_{i,n}$ are computed from the observed data X. Specifically, considering the ground RV $\theta_{i,n}$, we have $\mathbf{E}_{i,n} = f(\mathbf{X}_{i,n}^{\mathfrak{r}}[t])$ where $\mathbf{X}_{i,n}^{\mathfrak{r}}[t]$ corresponds to the observed data of a subset of $\theta_{i,n}$'s descendants. Depending on the chosen branching *downstream* of $\theta_{i,n}$, the value of $\mathbf{E}_{i,n}$ can therefore vary. This means we cannot further simplify Equation (A.11): the terms $\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\mathbf{E}_{i,n})$ are not constants with respect to the RVs $I_\mathcal{P}$. In the above example of $\theta_{1,2}$ in $\mathcal{M}$, in the PAVI-E scheme the expression $\mathbb{E}_{I_\mathcal{P}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\phi_i,\mathbf{E}_{i,n})\right]$ can be evaluated into:

$$\frac{1}{2}(\log q_{1,2}(\theta_{1,2}|\theta_{2,0};\phi_1,f(\mathbf{X}_4)) + \log q_{1,2}(\theta_{1,2}|\theta_{2,0};\phi_1,f(\mathbf{X}_5)))$$

How could the PAVI-E scheme be made unbiased? Specifically, by making the value of $\mathbf{E}_{i,n}$ independent of the choice of downstream branching. A possibility would be to parameterize $\mathbf{E}_{i,n}$ as an average —an expectation— over all the possible sub-branchings downstream of $\theta_{i,n}$. Yet, in practical cases, the cardinalities of the reduced model are much inferior to the ones of the full model: $\mathrm{Card}^{\mathfrak{r}}(\mathcal{P}) \ll \mathrm{Card}(\mathcal{P})$. This means that numerous $\mathrm{Card}^{\mathfrak{r}}(\mathcal{P})$-sized subsets can be chosen inside the $\mathrm{Card}(\mathcal{P})$ possible descendants. In order to average over all those subset choices to compute $\mathbf{E}_{i,n}$, numerous encoding calculations would be required at each stochastic training step. For large-scale cases, we deemed this possibility impractical. Other possibilities could exist, all revolving around the problem of aggregating collections of stochastic estimators into one general estimator —in an unbiased and efficient manner. To our knowledge, this is a complex and still open research question, whose advancement could much benefit our applications.

Practical approximation for the PAVI-E scheme In practice, we compute the encoding $\mathbf{E}_{i,n}$ based on the single downstream branching corresponding to the sampling of the RVs $I_{\mathcal{P}}$. Compared to the previous paragraph, this amounts to estimating the expectation of $\mathbf{E}_{i,n}$ —over all downstream branchings— using a single one of those branchings. Note that, even if this encoding estimate was unbiased, $\log q_{i,n}$ would remain a highly non-linear function of $\mathbf{E}_{i,n}$. As a consequence, we need to rely on the approximation:

$$\mathbb{E}_{I_{\mathcal{P}}}\left[\log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\phi_i, f(\mathbf{X}^{\mathfrak{r}}_{i,n}[t]))\right] \simeq \log q_{i,n}(\theta_{i,n}|\pi(\theta_{i,n});\phi_i, f(\mathbf{X}_{i,n})) \tag{A.13}$$

which can theoretically introduce some bias in our gradients. The approximation Equation (A.13) can be interpreted as follow: "the expectation of the density of $\theta_{i,n}$ when collecting summary statistics over a stochastic subset of $\theta_{i,n}$'s descendants is approximately equal to the density of $\theta_{i,n}$ when collecting summary statistics over the entirety of $\theta_{i,n}$'s descendants". Another interpretation is that the distribution associated with the summary of the full data can be approximated by annealing the distributions associated with summaries of subsets of this data. In practice, this approximation did not yield significantly worse performance for the PAVI-E scheme over the generative models we tested. At the same time, computing the encodings over a single branching allows the computation of all the $\mathbf{E}_{i,n}$ encodings in a single lightweight pass over the data $\mathbf{X}^{\mathfrak{r}}[t]$. This simple solution thus provided a substantial increase in training speed with seldom noticeable bias. Yet, we do not bar the existence of pathological generative HBMs where this approximation would become coarse. Experimenters should bear in mind this possibility when using the PAVI-E scheme. In practice, using the PAVI-F scheme as a sanity check over synthetic, toy-dimension implementations of the considered generative models is a good way to validate the PAVI-E scheme —before moving on to the real problem instantiating the same generative model with a larger dimensionality.

A.3 PAVI algorithms

More technical details can be found in the codebase provided with this supplemental material.

A.3.1 Architecture build

Algorithm 1: PAVI architecture build

Input: Graph template $\mathcal{T}$, plate cardinalities $\{(\text{Card}(\mathcal{P}_p), \text{Card}^{\mathfrak{r}}(\mathcal{P}_p))\}_{p=0..P}$, encoding scheme
Output: q distribution
for $i = 1..I$ **do**
  Construct conditional flow $\mathcal{F}_i$;
  Define conditional posterior distributions $q_{i,n}$ as the push-forward of the prior via $\mathcal{F}_i$, following Equation (3);
Combine the $q_{i,n}$ distributions following the cascading flows scheme, as in Section 4.1 (Ambrogioni et al., 2021b) ;
if *PAVI-F encoding scheme* **then**
  Construct encoding arrays $\{\mathbf{E}_i = [\mathbf{E}_{i,n}]_{n=0..N_i}\}_{i=1..I}$ as in Appendix A.1.2 ;
else if *PAVI-E encoding scheme* **then**
  Construct encoder f as in Appendix A.1.3 ;

A.3.2 Stochastic training

Algorithm 2: PAVI stochastic training

Input: Untrained architecture q, observed data $\mathbf{X}$, encoding scheme, number of steps T
Output: trained architecture q
for $t = 0..T$ **do**
  Sample plate indices to define the batches $\mathcal{B}_i[t]$, the latent $\Theta^{\mathfrak{r}}[t]$ and the observed $X^{\mathfrak{r}}[t]$ and $\mathbf{X}^{\mathfrak{r}}[t]$, following Appendix A.1.1 ;
  Define reduced distribution $p^{\mathfrak{r}}$;
  if *PAVI-F encoding scheme* **then**
    Collect encodings $\mathbf{E}_{i,n}$ by slicing from the arrays $\mathbf{E}_i$ the elements corresponding to the batches $\mathcal{B}_i[t]$;
  else if *PAVI-E encoding scheme* **then**
    Compute encodings as $\mathbf{E} = f(\mathbf{X}^{\mathfrak{r}}[t])$;
  Feed obtained encodings into $q^{\mathfrak{r}}$;
  Compute reduced ELBO as in Equation (6), back-propagate its gradient ;
  Update conditional flow weights $\{\phi_i\}_{i=1..I}$;
  if *PAVI-F encoding scheme* **then**
    Update encodings $\{\mathbf{E}_{i,n}\}_{i=1..I, n\in\mathcal{B}_{i,t}}$;
  else if *PAVI-E encoding scheme* **then**
    Update encoder weights η;

A.3.3 Inference

Algorithm 3: PAVI inference

Input: trained architecture q, observed data $\mathbf{X}$, encoding scheme
Output: approximate posterior distribution
if *PAVI-F encoding scheme* **then**
  Collect full encoding arrays $\mathbf{E}_i$;
else if *PAVI-E encoding scheme* **then**
  Compute encodings as $\mathbf{E} = f(\mathbf{X})$ using set size generalization ;
Feed obtained encodings into q ;

A.4 Inference gaps

In terms of inference quality, the impact of our architecture can be formalized following the *gaps* terminology (Cremer et al., 2018). Consider a joint distribution $p(\Theta, X)$, and a value $\mathbf{X}$ for the RV template X. We pick a variational family $\mathcal{Q}$, and in this family look for the parametric distribution $q(\Theta;\phi)$ that best approximates $p(\Theta|X = \mathbf{X})$. Specifically, we want to minimize the Kulback-Leibler divergence (Blei et al., 2017) between our variational posterior and the true posterior, that Cremer et al. (2018) refer to as the *gap* $\mathcal{G}$:

$$\begin{aligned}\mathcal{G} &= \mathrm{KL}(q(\Theta;\phi)||p(\Theta|X)) \\ &= \log p(X) - \mathrm{ELBO}(q;\phi)\end{aligned} \tag{A.14}$$

We denote $q^*(\Theta;\phi^*)$ the optimal distribution inside $\mathcal{Q}$ that minimizes the KL divergence with the true posterior:

$$\begin{aligned}\mathcal{G}_{\text{approx}}(\mathcal{Q};\phi^*) &= \log p(X) - \mathrm{ELBO}(q^*;\phi^*) \\ &\geq 0 \\ \mathcal{G}_{\text{vanilla VI}} &= \mathcal{G}_{\text{approx}}\end{aligned} \tag{A.15}$$

The *approximation gap* $\mathcal{G}_{\text{approx}}$ depends on the expressivity of the variational family $\mathcal{Q}$, specifically whether $\mathcal{Q}$ contains distributions arbitrarily close to the posterior —in the KL sense.

Note: $\mathcal{G}_{\text{approx}}$ is a property of the variational family $\mathcal{Q}$. $\mathcal{G}_{\text{approx}}$ is an asymptotic bound for the KL divergence between any distribution $q \in \mathcal{Q}$ and the true posterior. This gap is therefore a form of bias but is not to be mistaken with the stochasticity-induced bias studied in Appendix A.2. The bias in Appendix A.2 relates to whether q^* can be found by training stochastically over $\mathcal{Q}$, whereas $\mathcal{G}_{\text{approx}}$ relates to the the bias between q^* and the true posterior.

Cremer et al. (2018) demonstrate that, in the case of sample amortized inference, when the weights ϕ no longer are free but the output of an encoder $f \in \mathcal{F}$, inference cannot be better than in the non-sample-amortized case, and a positive *amortization gap* is introduced:

$$\begin{aligned} \mathcal{G}_{\text{sa}}(\mathcal{Q}, \mathcal{F}; \eta^*) &= \mathcal{G}_{\text{approx}}(\mathcal{Q}; f(\mathbf{X}, \eta^*)) - \mathcal{G}_{\text{approx}}(\mathcal{Q}; \phi^*) \\ &\geq 0 \\ \mathcal{G}_{\text{sample amortized VI}} &= \mathcal{G}_{\text{approx}} + \mathcal{G}_{\text{sa}} \end{aligned} \tag{A.16}$$

Where we denote as η^* the optimal weights for the encoder f inside the function family $\mathcal{F}$. The gap terminology can be interpreted as follow: "theoretically, sample amortization cannot be beneficial in terms of KL divergence for the inference over a given sample $\mathbf{X}$."

Using the same gap terminology, we can define gaps implied by our PAVI architecture. Instead of picking the distribution q inside the family $\mathcal{Q}$, consider picking q from the *plate-amortized* family $\mathcal{Q}_{\text{pa}}$ corresponding to $\mathcal{Q}$. Distributions in $\mathcal{Q}_{\text{pa}}$ are distributions from $\mathcal{Q}$ with the additional constraints that some weights have to be equal. Consequently, $\mathcal{Q}_{\text{pa}}$ is a subset of $\mathcal{Q}$:

$$\mathcal{Q}_{\text{pa}} \subset \mathcal{Q} \tag{A.17}$$

As such, looking for the optimal distribution inside $\mathcal{Q}_{\text{pa}}$ instead of inside $\mathcal{Q}$ cannot result in better performance, leading to a *plate amortization gap*:

$$\begin{aligned} \mathcal{G}_{\text{pa}}(\mathcal{Q}, \mathcal{Q}_{\text{pa}}; \psi^*, \phi^*) &= \mathcal{G}_{\text{approx}}(\mathcal{Q}_{\text{pa}}; \psi^*) - \mathcal{G}_{\text{approx}}(\mathcal{Q}; \phi^*) \\ &\geq 0 \\ \mathcal{G}_{\text{PAVI-F}} &= \mathcal{G}_{\text{approx}} + \mathcal{G}_{\text{pa}} \end{aligned} \tag{A.18}$$

Where we denote as ψ^* the optimal weights for a variational distribution q inside $\mathcal{Q}_{\text{pa}}$ —in the KL sense. The equation A.18 is valid for the PAVI-F scheme —see Section 4.3. We can interpret it as follow: "theoretically, plate amortization cannot be beneficial in terms of KL divergence for the inference over a given sample $\mathbf{X}$".

Now consider that encodings are no longer free parameters but the output of an encoder f. Similar to the case presented in Equation (A.16), using an encoder cannot result in better performance, leading to an *encoder gap*:

$$\begin{aligned} \mathcal{G}_{\text{encoder}}(\mathcal{Q}_{\text{pa}}, \mathcal{F}; \psi^*, \eta^*) &= \mathcal{G}_{\text{approx}}(\mathcal{Q}_{\text{pa}}; f(\mathbf{X}, \eta^*)) - \mathcal{G}_{\text{approx}}(\mathcal{Q}_{\text{pa}}; \psi^*) \\ &\geq 0 \\ \mathcal{G}_{\text{PAVI-E}} &= \mathcal{G}_{\text{approx}} + \mathcal{G}_{\text{pa}} + \mathcal{G}_{\text{encoder}} \end{aligned} \tag{A.19}$$

The equation Equation (A.19) is valid for the PAVI-E scheme —see Section 4.3.

The most complex case is the PAVI-E(sa) scheme, where we combine both plate and sample amortization. Our argument cannot account for the resulting $\mathcal{G}_{\text{PAVI-E(sa)}}$ gap: both the PAVI-E and PAVI-E(sa) schemes rely upon the same encoder f. In the PAVI-E scheme, f is overfitting over a dataset composed of the slices of a given data sample $\mathbf{X}$. In the PAVI-E(sa) scheme, the encoder is trained over the whole distribution of the samples of the reduced model $\mathcal{M}^{\mathfrak{r}}$. Intuitively, it is likely that the performance of PAVI-E(sa) will always be dominated by the performance of PAVI-E, but —as far as we understand it— the gap terminology cannot account for this discrepancy.

Comparing previous equations, we therefore have:

$$\mathcal{G}_{\text{vanilla VI}} \leq \mathcal{G}_{\text{PAVI-F}} \leq \mathcal{G}_{\text{PAVI-E}} \tag{A.20}$$

Note that those are *theoretical* results, that do not necessarily pertain to optimization in practice. In particular, in Section 5.1&5.3, this theoretical performance loss is not observed empirically over the studied

examples. On the contrary, in practice, our results can actually be better than non-amortized baselines, as is the case for the PAVI-F scheme in Figure 3 or experiments B.4. We interpret this as a result of a simplified optimization problem due to plate amortization —with fewer parameters to optimize for, and mini-batching effects across different ground RVs. A better framework to explain those discrepancies could be the one from Bottou & Bousquet (2007): performance in practice is not only the reflection of an *approximation error* but also of an *optimization error*. A less expressive architecture —using plate amortization— may in practice yield better performance. Furthermore, for the experimenter, the theoretical gaps $\mathcal{G}_{\text{pa}}, \mathcal{G}_{\text{encoder}}$ are likely to be well "compensated for" by the lighter parameterization and faster convergence entitled by plate amortization.

B Supplemental results

In this section, we present various supplemental experiments that further compare the PAVI-E and -F between themselves and against baselines. Overall, we found the PAVI-F scheme faster to train and to yield better inference quality than the PAVI-E scheme. When its parameterization is affordable, PAVI-F should be preferred. PAVI-E nevertheless opens up promising research directions, with the potential for parameterization-constant, time-constant sample-amortized inference as the problem cardinality augments. Though degraded compared to PAVI-F's, PAVI-E's performance is still on par or beats baselines in a variety of inference tasks —see experiments 5.3&B.4.

B.1 Tabular results for experiment 5.3

For convenience, we reproduce the results of our comparative experiment from Figure 3 in Table B.1.

B.2 GRE results sanity check

As exposed in the introduction of Section 5, in this work we focused on the usage of the ELBO as an inference performance metric (Blei et al., 2017):

$$\text{ELBO}(q) = \log p(X) - \text{KL}(q(\Theta)||p(\Theta|X)) \tag{B.21}$$

Given that the likelihood term $\log p(X)$ does not depend on the variational family q, differences in ELBOs directly transfer to differences in KL divergence, and provide a straightforward metric to compare different variational posteriors. Nonetheless, the ELBO doesn't provide an absolute metric of quality. As a sanity check, we want to assert the quality of the results presented in Section 5.3 —that are transferable to Section 5.1&5.2, based on the same model. In Figure B.1 we plot the posterior samples of various methods against approximate closed-form ground truths, using the $\text{Card}(\mathcal{P}_1) = 20$ case. All the method's results are aligned with the closed-form ground truth, with differences in ELBO translating meaningful qualitative differences in terms of inference quality.

B.3 Effect of the reduced model cardinalities on the training efficiency

In Figure B.2 we show the impact of the augmentation of $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ on the efficiency of the variational posterior's optimization.

In practice, we noticed that the most efficient choice in the case of the PAVI-F scheme was to maximize the cardinalities of the reduced model given the memory constraints of the GPU. Indeed, training over larger cardinalities does make each optimization step slightly slower but also makes the ELBO gradient estimates less noisy and allows to train more encoding vectors $\mathbf{E}_{i,n}$ at a given optimization step.

In the case of the PAVI-E scheme, the training speed is constant with respect to $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$. This is due to the computing of the encodings being vectorized across plates in our deep-set encoder —see Appendix A.1.3 (Lee et al., 2019). We observe a slight if barely noticeable reduction of the inference bias when augmenting $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$. Similar to the PAVI-F scheme, $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ should be maximized with respect to the GPU memory constraint. The intuition behind this choice is that the generalization of the learning of summary statistics

	Architecture	ELBO (10^1)	Parameterization	Optimization time (s)
$\text{Card}(\mathcal{P}_1) = 2$	ADAVI (sa)	1.88 (± 0.21)	12,280	503 (± 50)
	CF (sa)	1.48 (± 0.05)	2,275	228 (± 23)
	CF	1.98 (± 0.02)	355	63 (± 6)
	UIVI	2.11 (± 0.05)	2,764	220 (± 22)
	PAVI-F	2.14 (± 0.02)	4,058	65 (± 6)
	PAVI-E (sa)	2.03 (± 0.03)	27,394	140 (± 14)
	PAVI-E	1.74 (± 0.18)	27,394	37 (± 4)
	Asymptotic ELBO	2.20		
$\text{Card}(\mathcal{P}_1) = 20$	ADAVI (sa)	21.6 (± 1.36)	12,280	800 (± 80)
	CF (sa)	17.3 (± 0.60)	21,751	1,612 (± 161)
	CF	22.6 (± 0.01)	2,551	404 (± 40)
	UIVI	23.9 (± 0.02)	14,676	230 (± 23)
	PAVI-F	23.8 (± 0.02)	4,202	72 (± 7)
	PAVI-E (sa)	22.3 (± 0.30)	27,394	145 (± 14)
	PAVI-E	18.4 (± 2.63)	27,394	50 (± 5)
	Asymptotic ELBO	24.1		
$\text{Card}(\mathcal{P}_1) = 200$	ADAVI (sa)	221.7 (± 10.93)	12,280	1,600 (± 160)
	CF (sa)	175.7 (± 0.52)	216,511	12,000 (± 1,200)
	CF	228.6 (± 0.52)	24,511	2,600 (± 260)
	UIVI	241.2 (± 0.14)	139,300	240 (± 24)
	PAVI-F	237.7 (± 0.95)	5,642	81 (± 8)
	PAVI-E (sa)	227.0 (± 4.76)	27,394	150 (± 15)
	PAVI-E	224.2 (± 15.34)	27,394	85 (± 9)
	Asymptotic ELBO	243.4		

Table B.1: **Scaling with plate cardinalities** Reproduction of the results from Figure 3

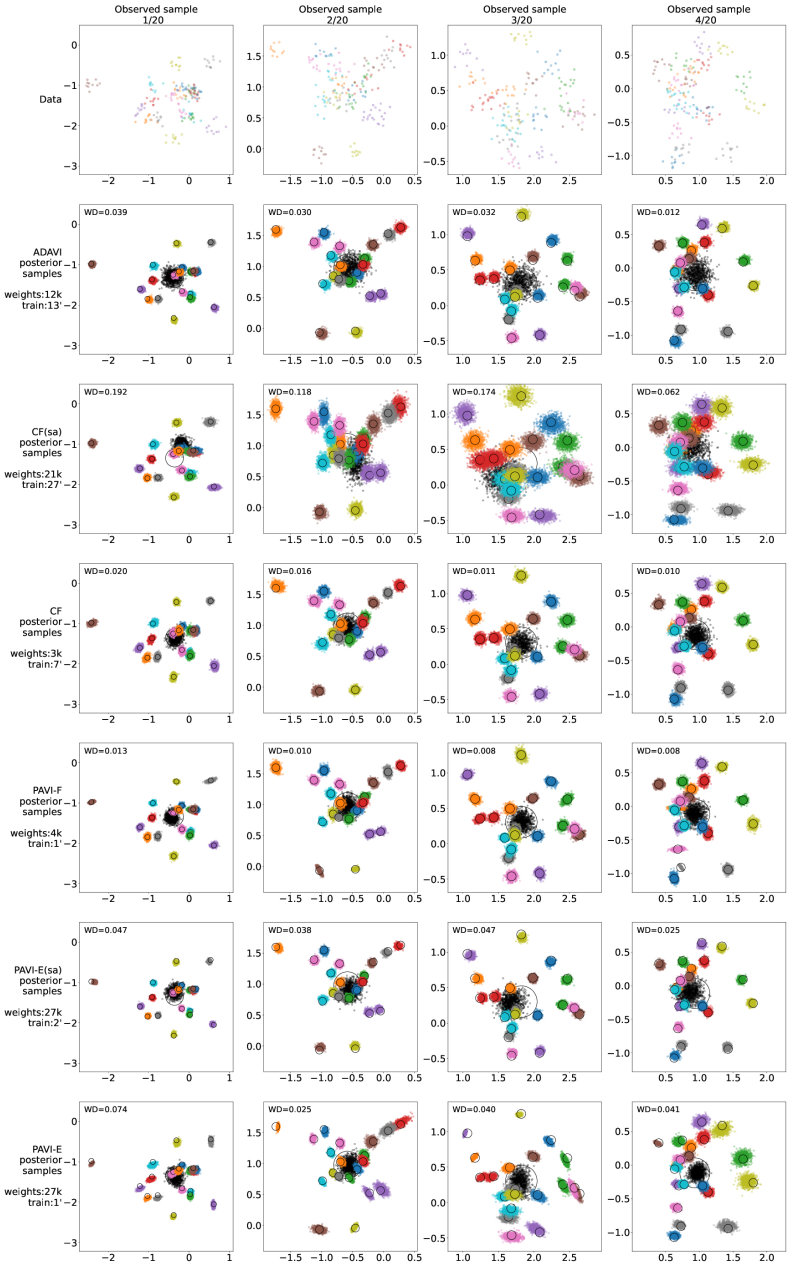


Figure B.1: **GRE Sanity check** Inference methods present qualitatively correct results, making ELBO comparisons relevant in our experiments. *On the topmost line*, we represent 4 different $\mathbf{X}$ samples for the GRE model described in Equation (10) with $\text{Card}(\mathcal{P}_1) = 20$. Each set of colored points represents the $\mathbf{X}_{n_1,\bullet}$ points belonging to one of the 20 groups. *Bottom lines* represent the posterior samples for the methods used in Section 5.3. Colored points are sampled from the posterior of the groups means θ_1, whereas black points are samples from the population mean θ_2. We represent as black circles a closed-form ground truth, centered on the correct posterior mean, and with a radius equal to 2 times the closed-form posterior's standard deviation. **Correct posterior samples should be centered on the same point as the corresponding black circle, and 95% of the points should fall within the black circle**. Associated with each posterior sample is a Wasserstein Distance (WD) computed with a sample from an approximate closed-form posterior (Flamary et al., 2021). PAVI is represented on the 3 last lines. Some minor bias can be observed for the PAVI-E scheme, but this approximation error is marginal compared to the optimization error that can be observed for unbiased methods, such as CF(sa) (Bottou & Bousquet, 2007). We can observe a superior quality for the PAVI-F scheme, rivaling ADAVI and CF's performance with orders of magnitude fewer parameters and training time, as visible in Figure 3.

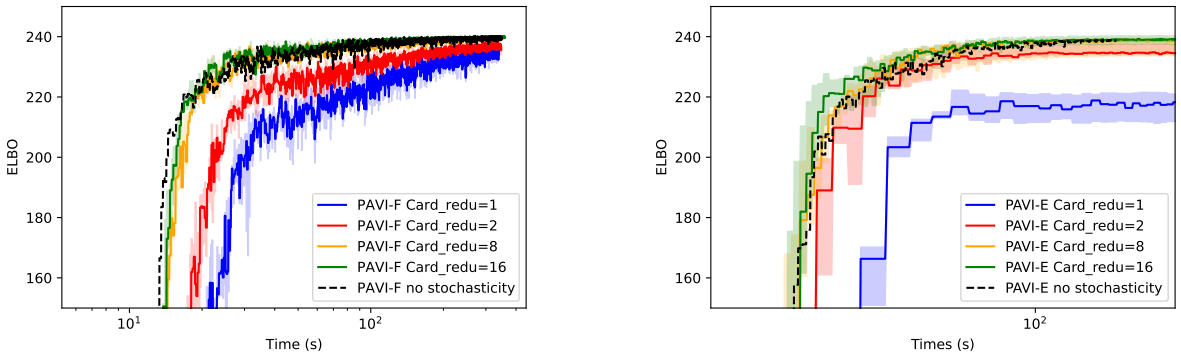


Figure B.2: **Both panels: Effect of** $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ **on the training efficiency** Experiment performed on the GRE model (see Equation (10)). We increase the cardinality of the reduced model $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ from 1 to 20 while keeping $\text{Card}(\mathcal{P}_1) = 20$ fixed. At $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 20$, there is no stochasticity in the training, meaning we train directly on $\mathcal{M}$. This experiment is interesting to evaluate the bias introduced in the stochastic training, as detailed in Appendix A.2. **Left panel: PAVI-F scheme** As $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ augments, the training gets faster and less noisy, likely due to less stochasticity in the gradient estimates and more encodings vectors $\mathbf{E}_{i,n}$ being trained at once. This increase in speed quickly caps, and the speed is approximately the same between $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 8$ and $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 16$. This experiment also illustrates the unbiasedness of the stochastic training: at $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 20$ there is no stochasticity in the training, and the asymptotic performance is the same as for the stochastic training (see Appendix A.2). **Right panel: PAVI-E scheme** The case of $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 1$ is pathological: the encoder "learns" to collect summary statistics across a set of 1 element, and —not surprisingly— the learned function doesn't generalize well on sets of size 20. In all of the other cases, the ELBO converges approximately to the same values as with $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 20$, that is to say when there is no stochasticity in the training (black curve). We furthermore observe a slight reduction of the inference bias as $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ augments. This illustrates how the theoretical bias of the PAVI-E scheme identified in Appendix A.2 does not translate in significantly worse empirical results —albeit in the pathological case of $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1) = 1$. Interestingly, the training speed is approximately constant across all values of $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$. This feature is essential in the PAVI-E scheme: we can train over a reduced version of our model —with a light memory footprint— and apply the obtained architecture to the full model. Note that this feature wouldn't necessarily be present when training on CPU: computing summary statistics over larger sets would make the training *slower* when $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ augments.

Architecture	ELBO (10^1)	Optimization time (s)
CF	- 16.2 ($\pm$ 2.9)	3,100
ADAVI	- 20.0 ($\pm$ 2.3)	3,000
UIVI	- 26.3 ($\pm$ 1.2)	500
PAVI-E (ours)	- 15.2 ($\pm$2.8)	470
PAVI-F (ours)	**- 11.1 ($\pm$1.1)**	**300**

Table B.2: **Analytical performance over a Gaussian mixture HBM** PAVI shows superior performance. PAVI converges in a fraction of CF's optimization time —the second-best-performing architecture.

across sets of data points is easier the closer the reduced set size is to the full set size. This constant training speed allows for a controlled memory footprint of the stochastic training: contrary to the PAVI-F scheme, $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ could be maintained at a value manageable by the GPU without reducing the training speed. This example also displays a pathological choice for $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$: the encoder fails to learn to compute the correct summary statistics over sets of size 1. However trivial, this example underlines that $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ should be chosen at a value logical with respect to the posterior's sufficient statistics. When generalizing to moments of a higher order —such as the variance— there is therefore a theoretical lower bound to consider when fixating the value of $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$. Our intuition is that the more complex to estimate the statistic is, the larger $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ should be.

Note that in practice non-stochastic VI is intractable for large-scale models, because of its memory requirement. In large-scale experiments such as the one presented in Figure 4, stochastic training is necessary. Our claim is to be faster than non-plate-amortized stochastic VI in those large-scale contexts —but not necessarily to be faster than non-stochastic VI in the small-scale regime of this experiment. For PAVI-F we nonetheless obtain similar training speed compared to non-stochastic VI with $\text{Card}^{\mathfrak{r}}(\mathcal{P}_1)$ as small as 8. For PAVI-E, the stochastic training is as fast as the non-stochastic one.

B.4 Additional comparative experiments

B.4.1 Gaussian Mixture model

In this experiment, we test out various baselines over a challenging model: a Gaussian mixture with random effects.

$$
\begin{aligned}
& D, \quad \text{Card}(\mathcal{P}_1), \quad \text{Card}(\mathcal{P}_0) = 2, \quad 20, \quad 10 \\
\substack{\forall n_1 = 1..\,\text{Card}(\mathcal{P}_1) \\ \forall n_0 = 1..\,\text{Card}(\mathcal{P}_0)} \quad & X_{n_1,n_0} | \theta_{1,n_1}, \pi_{n_1} \sim \text{Mixture}(\left[\mathcal{N}(\theta^1_{1,n_1}, \sigma_x^2), \ldots, \mathcal{N}(\theta^L_{1,n_1}, \sigma_x^2)\right], \pi_{n_1}) \\
\forall n_1 = 1..\,\text{Card}(\mathcal{P}_1) \quad & \pi_{n_1} \sim \text{Dirichlet}(1 \times \vec{1}_L) \\
\substack{\forall n_1 = 1..\,\text{Card}(\mathcal{P}_1) \\ \forall l = 1..L} \quad & \theta^l_{1,n_1} | \theta^l_{2,0} \sim \mathcal{N}(\theta^l_{2,0}, \sigma_1^2) \\
\forall l = 1..L \quad & \theta^l_{2,0} \sim \mathcal{N}(\vec{0}_D, \sigma_2^2) \ ,
\end{aligned}
\tag{B.22}
$$

where $\text{Mixture}([\mathcal{D}_1, \ldots, \mathcal{D}_L], \pi)$ denotes a mixture between the distributions $[\mathcal{D}_1, \ldots, \mathcal{D}_L]$ with mixture weights π. Results are visible in Table B.2, where PAVI displays the best asymptotic ELBO, as well as the shortest optimization time. On that note, we underline that stochastic training over a mixture distribution is challenging, as —due to the sub-sampling of points— only a fraction of the mixture components could be expressed at a given step, requiring the architecture to dynamically cluster the data points across time.

B.4.2 Hierarchical Variance model

In this experiment, we test out our architectures over a non-canonical model, in which parent RVs play the role of variance for the distribution of their children. Our goal is in particular to evaluate a potential empirical bias for the PAVI-E scheme —as studied in Appendix A.2. We will refer to the following HBM as

Architecture	ELBO (10^2)	Optimization time (s)
CF	- 11.2 ($\pm$ 2.9)	500
UIVI	-6.7 ($\pm$ 0.8)	100
PAVI-E (ours)	-6.7 ($\pm$1.0)[1]	90
PAVI-F (ours)	**- 6.7 ($\pm$0.8)**	**20**

Table B.3: **Analytical performance over a Hierarchical Variance HBM** In this setting with hard-to-estimate summary statistics, PAVI-E doesn't show any empirical bias. [1]: PAVI-E suffers from numerical instability in this example, with runs degenerating into NaN results.

Architecture	ELBO (10^4)	Optimization time (s)
CF	- 139.7 ($\pm$ 21)	3,200
ADAVI	—	—
UIVI	—	—
PAVI-F (ours)	**- 8.2 ($\pm$ 0.1)**	**1,600**

Table B.4: **Analytical performance over our Neuroimaging HBM** This HBM is not pyramidal and consequently cannot be processed by the ADAVI architecture. Due to the large parameter space (approximately 2k parameters) and the complex density involved, despite intensive efforts we did not manage to obtain a numerically stable UIVI optimization. PAVI yields a higher ELBO than CF, in half of CF's optimization time.

the Hierarchical Variance model:

$$\begin{aligned} & D, \quad \mathrm{Card}(\mathcal{P}_1), \quad \mathrm{Card}(\mathcal{P}_0) = 2, \quad 15, \quad 15 \\ \substack{\forall n_1 = 1..\,\mathrm{Card}(\mathcal{P}_1) \\ \forall n_0 = 1..\,\mathrm{Card}(\mathcal{P}_0)} & \quad \log X_{(n_1,n_0)} | \theta_{1,n_1} \sim \mathcal{N}(0, \theta_{1,n_1}) \\ \forall n_1 = 1..\,\mathrm{Card}(\mathcal{P}_1) & \quad \log \theta_{1,(n_1)} | \theta_{2,0} \sim \mathcal{N}(0, \theta_{0,0}) \\ & \quad \log \theta_{2,0} \sim \mathcal{N}(\vec{0}_D, 1) \end{aligned} \tag{B.23}$$

In this model, the encoder —used in the PAVI-E scheme— has to collect non-trivial summary statistics: empirical variances across variable subsets of the observed data. This is the context in which we would expect to observe the most bias due to the approximation in Equation (A.13). The performance of PAVI-E remains nonetheless competitive. This illustrates how PAVI-E's theoretical bias —introduced in the stochastic training— does not result in significantly worse inference compared to state-of-the-art architectures. PAVI also displays the best ELBO, in conjunction with UIVI, but in a 5 times shorter optimization time.

B.4.3 Small dimension version of our Neuroimaging model

In this experiment, we validate the performance of our architecture on synthetic data generated using a small dimension version of the model presented in Equation (B.24). To allow the use of the comparative baselines, we fixate $S, S^{\mathfrak{r}} = 5, 3$, $T, T^{\mathfrak{r}} = 6, 3$, $N, N^{\mathfrak{r}} = 12, 3$, $D = 4$, $L = 7$. Results are visible in Table B.4: PAVI can deal with HBMs unavailable to the ADAVI architecture (Rouillard & Wassermann, 2022), and so so with a performance superior to the CF baseline (Ambrogioni et al., 2021b). Despite intensive efforts, we did not manage to obtain a numerically stable UIVI optimization. This is likely due to the complexity of this inference problem: a large dimensionality (2k parameters) combined with complex dependencies between RVs in the posterior. In this context, automatic structured VI baselines such as CF or PAVI exploit the parametric form of the prior p to help with inference. PAVI's plate amortization likely facilitates the inference, resulting in a reduced *optimization error* (Bottou & Bousquet, 2007).

B.5 Experimental details - analytical examples

All experiments were performed in Python, using the *Tensorflow Probability* library (Dillon et al., 2017). Through this section, we refer to *Masked Autoregressive Flows* (Papamakarios et al., 2018) as *MAF*. All experiments are performed using the Adam optimizer (Kingma & Ba, 2015). At training, the ELBO was estimated using a Monte Carlo procedure with 8 samples. All architectures were evaluated over a fixed set of 20 samples $\mathbf{X}$, with 5 seeds per sample. Non-sample-amortized architectures were trained and evaluated on each of those points. Sample amortized architectures were trained over a dataset of $20,000$ samples separate from the 20 validation samples, then evaluated over the 20 validation samples.

Termination condition Our termination condition amounts to plateau detection across the statistics of multiple runs. In details, we ran the inference with a large, conservative number of epochs for all methods and collected the evolution of the ELBO for each run. We then took as runtime the time to reach an ELBO close to the one at convergence for the method. We averaged over 100 repetitions to avoid stochastic noise due to Monte Carlo estimation of the ELBO. The order of magnitude difference between the results is clearly above the potential noise effects due to our procedure.

B.5.1 Plate amortization and convergence speed (5.1)

All 3 architectures (baseline, PAVI-F, PAVI-E) used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units;
- as encoding size, 128

For the encoder f in the PAVI-E scheme, we used a multi-head architecture with 4 heads of 32 units each, 2 ISAB blocks with 64 inducing points.

B.5.2 Impact of encoding size (5.2)

All architectures used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units, after an affine block with triangular scaling matrix.
- as encoding size, a value varying from 2 to 16

B.5.3 Scaling with plate cardinalities (5.3)

ADAVI (Rouillard & Wassermann, 2022) we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units, after an affine block with triangular scaling matrix.
- for the encoder, an encoding size of 8 with a multi-head architecture with 2 heads of 4 units each, 2 ISAB blocks with 32 inducing points.

Cascading Flows (Ambrogioni et al., 2021b) we used:

- a mean-field distribution over the auxiliary variables r
- as auxiliary size, a fixed value of 8
- as flows, *Highway Flows* as designed by the Cascading Flows authors

PAVI-F we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units, after an affine block with triangular scaling matrix.
- an encoding size of 8

PAVI-E we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units, after an affine block with triangular scaling matrix.
- for the encoder, an encoding size of 16 with a multi-head architecture with 2 heads of 8 units each, 2 ISAB blocks with 64 inducing points.

UIVI we used as $\text{Card}(\mathcal{P}_1) = 2 \to 20 \to 200$:

- as base distribution, a standard Gaussian with dimensionality $6 \to 42 \to 402$
- as transform h, an affine transform with diagonal scale
- as embedding distribution, a standard Gaussian with dimensionality $3 \to 6 \to 9$
- as transform weights regressor, an MLP with hidden units $[32, 32] \to [64, 64] \to [128, 128]$
- to sample uncorrelated samples ϵ, an HMC run with 5 burn-in steps and 5 samples
- an Adam optimizer with exponential learning rate decay, starting at $1e-2$, $\times 0.9$ every 300 steps

B.5.4 Gaussian mixture (B.4.1)

ADAVI (Rouillard & Wassermann, 2022) we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32]$ hidden units, after an affine block with diagonal scaling matrix.
- for the encoder, an encoding size of 16 with a multi-head architecture with 2 heads of 8 units each, 2 ISAB blocks with 8 inducing points.

Cascading Flows (Ambrogioni et al., 2021b) we used:

- a mean-field distribution over the auxiliary variables r
- as auxiliary size, a fixed value of 16
- as flows, *Highway Flows* as designed by the Cascading Flows authors

PAVI-F we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32]$ hidden units, after an affine block with diagonal scaling matrix.
- an encoding size of 16
- $\text{Card}^{\mathbf{r}}(\mathcal{P}_1) = 5$

PAVI-E we used:

- for the flows $\mathcal{F}_i$, a MAF with $[128, 128]$ hidden units, after an affine block with triangular scaling matrix.

- for the encoder, an encoding size of 128 with a multi-head architecture with 4 heads of 32 units each, 2 ISAB blocks with 128 inducing points.
- $\text{Card}^{\mathbf{r}}(\mathcal{P}_1) = 5$

UIVI we used:

- as base distribution, a standard Gaussian with dimensionality 82
- as transform h, an affine transform with diagonal scale
- as embedding distribution, a standard Gaussian with dimensionality 6
- as transform weights regressor, a MLP with hidden units $[64, 64]$
- to sample uncorrelated samples ϵ, an HMC run with 5 burn-in steps and 5 samples
- an Adam optimizer with exponential learning rate decay, starting at $1e-2$, $\times 0.9$ every 300 steps

B.5.5 Hierarchical Variances (B.4.2)

Cascading Flows (Ambrogioni et al., 2021b) we used:

- a mean-field distribution over the auxiliary variables r
- as auxiliary size, a fixed value of 32
- as flows, *Highway Flows* as designed by the Cascading Flows authors

PAVI-F we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units, after an affine block with diagonal scaling matrix.
- an encoding size of 32
- $\text{Card}^{\mathbf{r}}(\mathcal{P}_1) = 3$
- $\text{Card}^{\mathbf{r}}(\mathcal{P}_0) = 3$

PAVI-E we used:

- for the flows $\mathcal{F}_i$, a MAF with $[128, 128]$ hidden units, after an affine block with diagonal scaling matrix.
- for the encoder, an encoding size of 128 with a multi-head architecture with 4 heads of 32 units each, 2 ISAB blocks with 128 inducing points.
- $\text{Card}^{\mathbf{r}}(\mathcal{P}_1) = 3$
- $\text{Card}^{\mathbf{r}}(\mathcal{P}_0) = 3$

UIVI we used:

- as base distribution, a standard Gaussian with dimensionality 32
- as transform h, an affine transform with diagonal scale
- as embedding distribution, a standard Gaussian with dimensionality 3
- as transform weights regressor, an MLP with hidden units $[64, 64]$
- to sample uncorrelated samples ϵ, an HMC run with 5 burn-in steps and 5 samples
- an Adam optimizer with exponential learning rate decay, starting at $1e-2$, $\times 0.9$ every 300 steps

B.5.6 Small Neuroimaging example (B.4.3)

Cascading Flows (Ambrogioni et al., 2021b) we used:

- a mean-field distribution over the auxiliary variables r
- as auxiliary size, a fixed value of 32
- as flows, *Highway Flows* as designed by the Cascading Flows authors

PAVI-F we used:

- for the flows $\mathcal{F}_i$, a MAF with $[32, 32]$ hidden units, after an affine block with diagonal scaling matrix.
- an combination of encoding sizes of 32 and 8

UIVI we used:

- as base distribution, a standard Gaussian with dimensionality $1,802$
- as transform h, an affine transform with diagonal scale
- as embedding distribution, a standard Gaussian with dimensionality 16
- as transform weights regressor, an MLP with hidden units $[128, 256, 512, 1024]$
- to sample uncorrelated samples ϵ, an HMC run with 5 burn-in steps and 5 samples
- an Adam optimizer with exponential learning rate decay, starting at $1e-3$, $\times 0.9$ every 300 steps

Optimization systematically degenerated into NaN results after around 200 optimization steps.

B.6 Details about our Neuroimaging experiment (5.4)

B.6.1 Data description

In this experiment, we use data from the *Human Connectome Project (HCP)* dataset (Van Essen et al., 2012). We randomly select a cohort of $S = 1,000$ subjects from this dataset, each subject is associated with $T = 2$ resting-state fMRI sessions (Smith et al., 2013). We minimally pre-process the signal using the `nilearn` python library (Abraham et al., 2014):

1. removing high variance confounds
2. detrending the data
3. band-filtering the data (0.01 to 0.1 Hz), with a repetition time of 0.74 seconds
4. spatially smoothing the data with a 4mm Full-Width at Half Maximum

For every subject, we extract the surface Blood Oxygenation Level Dependent (BOLD) signal of $N = 59,412$ vertices across the whole cortex. We compare this signal with the extracted signal of $D = 64$ DiFuMo components: a dictionary of brain spatial maps allowing for an effective fMRI dimensionality reduction (Dadi et al., 2020). Specifically, we compute the one-to-one Pearson's correlation coefficient of every vertex with every DiFuMo component. The resulting connectome, with S subjects, T sessions, N vertices and a connectivity signal with D dimensions, is of shape $(S \times T \times N \times D)$. We project this data —correlation coefficients lying in $]-1; 1[$— in an unbounded space using an inverse sigmoid function.

B.6.2 Model description

We use a model inspired by the work of Kong et al. (2019). We hypothesize that every vertex in the cortex belongs to either one of $L = 7$ functional networks. This number is inspired by the work of Thomas Yeo et al. (2011) and Kong et al. (2019), in which authors identify 7 general networks on resting-state fMRI data.

Each network corresponds to a pattern of connectivity with the brain, represented as the correlation of the BOLD signal with the signal from the $D = 64$ DiFuMo components. We define $L = 7$ functional networks at the population level. Each network l corresponds to the connectivity fingerprint μ_l.

At the population level, for a vertex n we denote as $\text{logits}_n \in \mathrm{R}^L$ the logits of its probability to belong to each network l. Each subject s is associated with the logits $\text{logits}_{s,n}$, which are considered as a perturbation of the population logits. This creates a regularization across subjects: the same vertex n is encouraged to have the same label across all subjects. The variable γ_l controls the inter-subject spatial variability across all subjects, for the network l.

For every subject s, session t, and vertex n, we denote as $X_{s,t,n}$ the observed connectivity. This connectivity is modeled via a mixture model: $X_{s,t,n}$ is a perturbation of either one of the μ_l's connectivity fingerprints. $\text{label}_{s,n}$ denotes the label in the mixture (that depends on the subject-specific logits). κ_l controls the variability between $X_{s,t,n}$ and the mixture component $\mu_{\text{label}_{s,n}}$.

The resulting model can be described as:

$$
\begin{aligned}
S, T, N, D, L &= 1000, 2, 59412, 64, 7 \\
\forall l{=}1..L: \quad \mu_l &\sim \mathcal{N}(0 \times \vec{1}_D, 6) \\
\forall l{=}1..L: \quad \log \kappa_l &\sim \mathcal{N}(0 \times \vec{1}_L, 1) \\
\forall n{=}1..N: \quad \text{logits}_n &\sim \mathcal{N}(0 \times \vec{1}_L, 6) \\
\forall l{=}1..L: \quad \log \gamma_l &\sim \mathcal{N}(0 \times \vec{1}_L, 1) \\
\forall s{=}1..S, \forall n{=}1..N: \quad \text{logits}_{s,n} \,|\, \text{logits}_n, [\gamma_l]_{l=1..L} &\sim \mathcal{N}(\text{logits}_n, [\gamma_1 ... \gamma_L]) \\
\forall s{=}1..S, \forall n{=}1..N: \quad \text{labels}_{s,n} \,|\, \text{logits}_{s,n} &\sim \text{Categorical}(\text{logits}_{s,n}) \\
\forall s{=}1..S, \forall t{=}1..T, \forall n{=}1..N: \quad X_{s,t,n} | [\mu_l]_{l=1..L}, [\kappa_l]_{l=1..L}, \text{label}_{s,n} &\sim \mathcal{N}(\mu_{\text{label}_{s,n}}, \kappa_{\text{label}_{s,n}})
\end{aligned}
\tag{B.24}
$$

The model contains 4 plates: the *network* plate of full cardinality L (that we did not exploit in our implementation), the *subject* plate of full cardinality S, the *session* plate of full cardinality T and the *vertex* plate of full cardinality N.

Our goal is to recover the posterior distribution of the networks μ and the labels label —represented as the parcellation in Figure 4— given the observed connectome described in Appendix B.6.1.

B.6.3 PAVI implementation

We used in this experiment the PAVI-F scheme, using:

- for the RVs μ_l:
  - for the flows $\mathcal{F}_i$, a MAF with $[64, 128, 64]$ hidden units, following an affine block with diagonal scale
  - for the encoding size: 16
- for the RVs κ_l, γ_l:
  - a MAP estimator using a regressor with $[64, 128, 64]$ hidden units
  - for the encoding size: 8
- for the RVs $\text{logits}_n, \text{logits}_{s,n}$:
  - for the flows $\mathcal{F}_i$, an affine block with diagonal scale regressed using a $[64, 128, 64]$ MLP

  - for the encoding size: 8

- for the reduced model, we used $S^{\mathfrak{r}} = 128$, $T^{\mathfrak{r}} = 1$ and $N^{\mathfrak{r}} = 128$.

To allow for the optimization over the discrete $\text{label}_{s,n}$ RV, we used the Gumbell-Softmax trick, using a fixed temperature of 1.0 (Jang et al., 2017; Maddison et al., 2016).

B.6.4 Cognitive scores prediction

For the cognitive scores prediction in Figure 4, we reproduced the standard methodology from Kong et al. (2019).

We perform a 20-fold cross-validation across 1,000 subjects. We start from the $\text{logits}_{s,n}$ associated with each subject. We use PCA to project the features from the 19 training folds to their 100 first components (with 33% explained variance). We then train a linear regression to predict each of the 13 cognitive scores from the training-fold PCA features. We compute the test performance on the test fold —using the training set PCA and linear regression— averaged across the 13 cognitive measures. This process is reproduced 100 times. The reported scores are the triple average across folds, measures, and repetition, while the standard deviation is computed across the 100 repetitions only.

C Supplemental discussion

C.1 Plate amortization as a generalization of sample amortization

In Section 4.1, we introduced plate amortization as applying the generic concept of amortization to the granularity of plates. There is actually an even stronger connection between sample amortization and plate amortization.

An HBM p models the distribution of a given observed RV X —jointly with the parameters Θ. Different samples $\mathbf{X}_0, \mathbf{X}_1, ...$ of the model p are i.i.d. draws from the distribution $p(X)$. p can thus be considered as the model for "one sample". Instead of p, consider a "macro" model for the whole *population* of samples one could draw from p. The observed RV of that macro model would be the infinite collection of samples drawn from the same distribution $p(X)$. In that light, the i.i.d. sampling of different X values from p could be interpreted as a plate of the macro model. Thus, we could consider sample amortization as an instance of plate amortization for the "sample plate". Or equivalently: plate amortization can be seen as the generalization of amortization beyond the particular case of sample amortization.

C.2 Alternate formalism for SVI — PAVI-E(sa) scheme

In this work, we propose a different formalism for SVI, based around the concept of full HBM $\mathcal{M}^{\text{full}}$ versus reduced HBM $\mathcal{M}^{\text{redu}}$ sharing the same template $\mathcal{T}$. This formalism is helpful to set up GPU-accelerated stochastic VI (Dillon et al., 2017), as it entitles a fixed computation graph -with the cardinality of the reduced model $\mathcal{M}^{\text{redu}}$- in which encodings are "plugged in" -either sliced from larger encoding arrays or as the output of an encoder applied to a data slice, see Section 4.1&4.2. Particularly, our formalism doesn't entitle a control flow over models and distributions, which can be hurtful in the context of *compiled* computation graphs such as in *Tensorflow* (Abadi et al., 2015).

The reduced model formalism is also meaningful in the PAVI-E(sa), where we train and amortized variational posterior over $\mathcal{M}^{\text{redu}}$ and obtain "for free" a variational posterior for the full model $\mathcal{M}^{\text{full}}$ —see Section 4.2. In this context, our scheme is no longer a different take on hierarchical, batched SVI: the cardinality of the full model is truly independent of the cardinality of the training and is only simulated as a scaling factor in the stochastic training —see Section 4.2. We have the intuition that fruitful research directions could stem from this concept.

C.3 Benefiting from structure in inference

Our contributions can be abstracted through the concept of plate amortization -see Section 4.1. Plate amortization is particularly useful in the context of heavily parameterized density approximators such as normalizing flows, but is not tied to it: plate-amortized Mean Field (Blei et al., 2017), ASVI (Ambrogioni et al., 2021a), or implicit (Yin & Zhou, 2018; Titsias & Ruiz, 2019) schemes are also possible to use. Plate amortization can be viewed as the amortization of common density approximators across different sub-structures of a problem. This general concept could have applications in other highly-structured problem classes such as graphs or sequences (Wu et al., 2020; Salehinejad et al., 2018).

Central to our design, PAVI adjoins an encoding structure to an inference problem. The encodings $\mathbf{E}_{i,n}$ embed all the "individualized" information in the problem, while the shared conditional density estimators $\mathcal{F}_i$ translate those encodings into posterior distributions. We believe there is potential in externalizing part of the inference problem into an embedding. For instance, embeddings could be learned separately from density estimators, opening up the possibility for transfer learning (Weiss et al., 2016). Or the encodings could integrate some known symmetry of the problem, such as convolution-based encodings in the case of Random Fields (Bishop, 2006).

C.4 Connection with meta-learning

In Section 4.1, we introduced plate amortization: sharing the parameterization and learning across a model's plates. This section discusses the connection between plate amortization and meta-learning(Ravi & Beatson, 2019; Iakovleva et al., 2020; Yao et al., 2019).

Supervised learning can be seen as the mapping from a given context set $\mathcal{C} = \{(x, y)\}$ to a predictive function f (Bishop, 2006) such that $f(x) = y$. Meta-learning —or "learning to learn"— instead recovers this mapping $\mathcal{C} \mapsto f$ in the general case. Once the meta-training is completed, a predictive function f conditioned by an unseen context $\mathcal{C}$ can be obtained in a single forward pass —without any training done on $\mathcal{C}$. As an instance of meta-learning, the Neural Process Family encodes the context $\mathcal{C}$ via a deep set encoder (Garnelo et al., 2018; Dubois et al., 2020; Zaheer et al., 2018). The encoded context, along with the data point x is then used to condition an estimator for the density $q(y|x, \mathcal{C})$.

This framework is similar to the PAVI-E scheme, where a combination of a deep set encoder and a normalizing-flow-based density estimator output the posterior probability of a ground RV $\theta_{i,n}$. This encoder-estimator pair is repeatedly used across a plate. This is analogous to meta-learning to solve the inference problem across the different elements of a plate —such as the different subjects of a population study. In PAVI-E, the encoding $\mathbf{E}_{i,n}$ at a lower hierarchy plays a role similar to the context $\mathcal{C}$ in meta-leaning. A few differences, however, exist between the two frameworks:

- meta-learning is typically concatenated to the 1-plate regime, whereas PAVI is designed for the multi-hierarchy scenario;
- meta-learning typically considers a set of i.i.d. tasks, whereas in PAVI the inference of different ground RVs $\theta_{i,n}$ are conditionally dependent through the hierarchical model p;
- meta-learning is trained using the forward KL loss, that is to say in the sample-amortized regime, maximizing the probability $q(y)$ of samples from the underlying generative process. In contrast, PAVI —though possible to train using the forward KL loss— is trained via the reverse KL, which requires to evaluate the density of the generative process explicitly.

We suspect there would be interesting applications for the PAVI architecture in hierarchical meta-learning scenarios.

C.5 Conditional dependencies modeled in the variational family

Consider an inference problem with three latent parameters θ_1, θ_2, θ_3, and a model $p(\Theta, X)$ that factorizes as $p(\theta_1, \theta_2, \theta_3) = p(\theta_1)p(\theta_2)p(\theta_3|\theta_1, \theta_2)p(X|\theta_3)$. To approximate the posterior $p(\Theta|X)$, different conditional

dependencies can be modeled in the chosen variational family $\mathcal{Q}$. The mean-field (MF) approximation fully factorizes the variational distribution as $q(\theta_1, \theta_2, \theta_3) = q(\theta_1)q(\theta_2)q(\theta_3)$. MF was originally introduced in VI to facilitate computation, allowing dedicated optimization schemes (Blei et al., 2017). Nonetheless, this approximation assumes independence between RVs in the posterior and ultimately limits the expressivity of the variational family. To remove this approximation, modeling complex conditional dependencies in the variational family is an open research subject (Ambrogioni et al., 2021b; Webb et al., 2018). In the PAVI design, we inherit our statistical dependency structure from the prior distribution p, as detailed in Equation (3). This amounts to factorizing $q(\theta_1, \theta_2, \theta_3) = q(\theta_1)q(\theta_2)q(\theta_3|\theta_1, \theta_2)$. This choice of dependencies follows the line of research from Hoffman & Blei (2014) and Ambrogioni et al. (2021a). As pointed out by Ambrogioni et al. (2021b), when modeling only those *forward* dependencies, the modeling of colliders can be an issue. More generally, one would like to model arbitrary dependencies in the variational posterior. As an example, the factorization $q(\theta_1, \theta_2, \theta_3) = q(\theta_3)q(\theta_2|\theta_3)q(\theta_1|\theta_2, \theta_3)$ would more faithfully represent the conditional dependencies that can arise in the posterior. Modeling arbitrary dependencies reduces the inference gap but can become computationally intractable as the number of RVs increases. In PAVI, we strike a particular balance between expressivity and computational tractability, as further motivated in this section.

In the case of a single plate —the 2-level case— Agrawal & Domke (2021) demonstrate that modeling only the *forward* dependencies does not result in reduced expressivity compared to the modeling of the full dependencies. Yet this result does not hold in the n-level case, as Webb et al. (2018) shows that faithful inversion features conditional dependencies in the posterior between ground RVs of the same template. We can dub those dependencies as *horizontal* dependencies across RVs in the same plate. Like Structured SVI (Hoffman & Blei, 2014) or Automatic SSVI (Ambrogioni et al., 2021a), the PAVI design thus results in reduced expressivity when stacking multiple plates in the model p. This issue can be partially alleviated with the usage of a *backward* encoding scheme —going in the reverse direction compared to the prior's dependencies— as in Cascading Flows (Ambrogioni et al., 2021b).

In practice, though limiting our expressivity, *horizontal* dependencies are difficult to inject back into our architecture. Critically, in the PAVI design, the use of a common density estimator across the ground RVs of the same template (see Section 4.1) and the stochastic training over batches of those RVs (see Section 4.2) prevent the direct modeling *horizontal* dependencies. Put differently, the fact that we consider the inference over different ground RVs as conditionally independent inference problems is central to our design and adverse to the modeling of *horizontal* dependencies.

Injecting *horizontal* dependencies back into our variational family is therefore a non-trivial research direction, that is not at the core of this paper. This opens up promising research directions: how could arbitrary conditional dependencies be modeled in the variational posterior in the context of stochastic training?

C.6 Towards user-friendly Variational Inference

By re-purposing the concept of amortization at the plate level, we aim to propose clear computation versus precision trade-offs in VI. Hyper-parameters such as the encoding size —as illustrated in Figure 2 (right)— allow to clearly trade inference quality in exchange for a reduced memory footprint. On the contrary, in classical VI, changing $\mathcal{Q}$'s parametric form —for instance, switching from Gaussian to Student distributions— can have a strong and complex impact on the number of weights and inference quality (Blei et al., 2017). By allowing the usage of normalizing flows in very large cardinality regimes, our contribution aims at disentangling approximation power and computational feasibility. In particular, having access to expressive density approximators for the posterior can help experimenters diversify the proposed HBMs, removing the need for properties such as conjugacy to obtain meaningful inference (Gelman et al., 2004). Combining clear hyper-parameters and scalable yet universal density approximators, we tend towards a user-friendly methodology in the context of large population studies VI.

In its current implementation, PAVI requires to set up a rather large number of hyper-parameters. This includes choosing a normalizing flow architecture, the size of the encodings at different plate levels, and the reduced model cardinalities. Those hyper-parameters are in addition to generic ones such as the number of samples from the variational family used to estimate the ELBO, or the choice of an optimizer. However,

our experiments suggest that those hyper-parameters could default to conservative values. As shown in Section 5.2, the encoding size should be maximized with respect to the available memory, as should the reduced cardinalities, as detailed in Appendix B.3. As for the normalizing flow architecture, since the latter is shared across plates, the user can always default to the most expressive architecture available without incurring an exploding number of weights. As an example, in our experiments, we stuck to the MAF architecture (Papamakarios & Murray, 2018). Since all hyper-parameters can be set conservatively, a user-friendly automatic VI API can be designed, where the user only provides the system with the generative model p, and some observed value $\mathbf{X}$. The variational family q can then automatically be derived from p, and directly optimized over $\mathbf{X}$. This is the design principle we implemented in the codebase adjoined to this publication.