View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by The IT University of Copenhagen's Repository

Partially Exchangeable Networks and Architectures for Learning Summary Statistics in Approximate Bayesian Computation

Samuel Wiqvist[*1] **Pierre-Alexandre Mattei**[*2] **Umberto Picchini**[3] **Jes Frellsen**[2]

Abstract

We present a novel family of deep neural architectures, named partially exchangeable networks (PENs) that leverage probabilistic symmetries. By design, PENs are invariant to block-switch transformations, which characterize the partial exchangeability properties of conditionally Markovian processes. Moreover, we show that any block-switch invariant function has a PEN-like representation. The DeepSets architecture is a special case of PEN and we can therefore also target fully exchangeable data. We employ PENs to learn summary statistics in approximate Bayesian computation (ABC). When comparing PENs to previous deep learning methods for learning summary statistics, our results are highly competitive, both considering time series and static models. Indeed, PENs provide more reliable posterior samples even when using less training data.

1. Introduction

We propose a novel neural network architecture to ease the application of approximate Bayesian computation (ABC), a.k.a. *likelihood-free* inference. The architecture, called partially exchangeable network (PEN), uses partial exchangeability in Markovian data, allowing us to perform ABC inference for time series models with Markovian structure. Empirically, we also show that we can target non-Markovian time series data with PENs. Since the DeepSets architecture (Zaheer et al., 2017) turns out to be a special case of PEN, we can also perform ABC inference for static models. Our work is about automatically construct summary statistics of the data that are informative for model parameters. This is a main challenge in the practical application of ABC algorithms, since such summaries are often *hand-picked* (i.e. ad-hoc summaries are constructed from model domain expertise), or these are automatically constructed using a number of approaches as detailed in Section 2. Neural networks have been previously used to automatically construct summary statistics for ABC. Jiang et al. (2017) and Creel (2017) employ standard multilayer perceptron (MLP) networks for learning the summary statistics. Chan et al. (2018) introduce a network that exploits the exchangeability property in exchangeable data. Our PEN architecture is a new addition to the tools for automatic construction of summary statistics, and PEN produces competitive inference results compared to Jiang et al. (2017), which in turn was shown outperforming the semi-automatic regression method by Fearnhead & Prangle (2012). Moreover, our PEN architecture is more data efficient and when reducing the training data PEN outperforms Jiang et al. (2017), the factor of reduction being of order 10 to 10^2 depending on cases.

Our **main contributions** are:

- Introducing the partially exchangeable networks (PENs) architecture;
- Using PENs to automatically learn summary statistics for ABC inference. We consider both static and dynamic models. In particular, our network architecture is specifically designed to learn summary statistics for dynamic models.

2. Approximate Bayesian computation

Approximate Bayesian computation (ABC) is an increasingly popular inference method for model parameters θ, in that it only requires the ability to produce artificial data from a stochastic model *simulator* (Beaumont et al., 2002; Marin et al., 2012). A simulator is essentially a computer program, which takes θ, makes internal calls to a random number generator, and outputs a vector of artificial data. The implication is that ABC can be used to produce approximate inference when the likelihood function $p(y|\theta)$ underlying

[*]Equal contribution [1]Centre for Mathematical Sciences, Lund University, Lund, Sweden [2]Department of Computer Science, IT University of Copenhagen, Copenhagen, Denmark [3]Department of Mathematical Sciences, Chalmers University of Technology and the University of Gothenburg, Gothenburg, Sweden. Correspondence to: Jes Frellsen <jefr@itu.dk>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

the simulator is intractable. As such ABC methods have been applied to a wide range of disciplines (Sisson et al., 2018). The fundamental idea in ABC is to generate parameter proposals $\theta^\star$ and accept a proposal if the simulated data $y^\star$ for that proposal is similar to observed data y^{obs}. Typically this approach is not suitable for high-dimensional data, and a set of summary statistics of the data is therefore commonly introduced to break the *curse-of-dimensionality*. So, instead of comparing $y^\star$ to y^{obs}, we compare summary statistics of the simulated data $s^\star = S(y^\star)$ to those of observed data $s^{\text{obs}} = S(y^{\text{obs}})$. Then we accept the proposed $\theta^\star$ if $s^\star$ is close to s^{obs} in some metric. Using this scheme, ABC will simulate draws from the following approximate posterior of θ

$$p^{\epsilon}_{\text{ABC}}(\theta|s^{\text{obs}}) \propto \int K_\epsilon(\Delta(s^\star, s^{\text{obs}}))p(s^\star|\theta)p(\theta)ds^\star,$$

where $p(\theta)$ is the prior of θ, Δ is a distance function between observed and simulated summaries (we use a Mahalanobis distance, see the supplementary material), $K_\epsilon(\cdot)$ is a kernel, which in all our applications is the uniform kernel returning 1 if $\Delta(s^\star, s^{\text{obs}}) < \epsilon$ and 0 otherwise, and $\epsilon > 0$ is the so-called ABC-threshold. A smaller ϵ produces more accurate approximations to the true summaries posterior $p(\theta|s^{\text{obs}})$, though this implies a larger computational effort due to the increasing number of rejected proposals. An additional issue is that ideally we would like to target $p(\theta|y^{\text{obs}})$, not $p(\theta|s^{\text{obs}})$, but again unless sufficient statistics are available (impossible outside the exponential family), and since $\epsilon > 0$, we have to be content with samples from $p^{\epsilon}_{\text{ABC}}$.

In this work we do not focus on *how* to sample from $p^{\epsilon}_{\text{ABC}}(\theta|s^{\text{obs}})$ (see Sisson et al., 2018 for possibilities). Therefore, we employ the simplest (and also most inefficient) ABC algorithm, the so called "ABC rejection sampling" (Pritchard et al., 1999). We will use the "reference table" version of ABC rejection sampling (e.g. Cornuet et al., 2008), which is as follows:

- Generate $\tilde{N}$ independent proposals $\theta^i \sim p(\theta)$, and corresponding data $y^i \sim p(y|\theta^i)$ from the simulator;
- Compute the summary statistics $s^i = S(y^i)$ for each $i = 1, ..., \tilde{N}$;
- Compute the distances $\Delta(s^i, s^{\text{obs}})$ for each $i = 1, ..., \tilde{N}$;
- Retain proposals θ^i corresponding to those $\Delta(s^i, s^{\text{obs}})$ that are smaller than the x-th percentile of all distances.

The retained θ^i's form a sample from $p^{\epsilon}_{\text{ABC}}$ with ϵ given by the selected xth percentile. An advantage of this approach is that it allows to easily compare the quality of the ABC inference based on several methods for computing the summaries, under the same computational budget $\tilde{N}$. Moreover, once the "reference table" $(\theta^i, y^i)_{1\leq i\leq \tilde{N}}$ has been produced in the first step, we can recycle these simulations to produce new posterior samples using several methods for computing the summary statistics.

2.1. Learning summary statistics

Event though ABC rejection sampling is highly inefficient due to proposing parameters from the prior $p(\theta)$, this is not a concern for the purpose of our work. In fact, our main focus is *learning* the summary statistics $S(\cdot)$. This is perhaps the most serious difficulty affecting the application of ABC methodology to practical problems. In fact, we require summaries that are informative for θ, as a replacement for the (unattainable) sufficient statistics. A considerable amount of research has been conducted on how to construct informative summary statistics (see Blum et al., 2013 and Prangle, 2015 for an overview). However their selection is still challenging since no state-of-the-art methodology exists that can be applied to arbitrarily complex problems. Fearnhead & Prangle (2012) consider a regression-based approach where they also show that the best summary statistic, in terms of the minimal quadratic loss, is the posterior mean. The latter is however unknown since $p(\theta|y^{\text{obs}})$ itself is unknown. Therefore, they introduce a simulation approach based on a linear regression model

$$\theta^i_j = E(\theta_j|y^i) + \xi^i_j = b_{0_j} + b_j h(y^i) + \xi^i_j \qquad (1)$$

with ξ^i_j some mean-zero noise. Here $j = 1, ..., \dim(\theta)$ and $h(y^i)$ is a vector of (non)-linear transformations of "data" y^i (here y^i can be simulated or observed data). Therefore Fearnhead & Prangle (2012) have $\dim(\theta)$ models to fit separately, one for each component of vector θ. Of course, these fittings are to be performed *before* ABC rejection is executed, so this is a step that anticipates ABC rejection, to provide the latter with suitable summary statistics. The parameters in each regression (1) are estimated by fitting the model by least squares to a new set of N simulated data-parameter pairs $(\theta^i, y^i)_{1\leq i\leq N}$ where, same as for ABC rejection, the θ^i are generated from $p(\theta)$ and the y^i are generated from the model simulator conditionally on θ^i. To clarify the notation: N is the number of data-parameter pairs used to fit the linear regression model in (1), while $\tilde{N}$ is the number of parameter-data pair proposals used in ABC rejection sampling. However the two sets of parameter-data pairs $(\theta^i, y^i)_{1\leq i\leq N}$ and $(\theta^i, y^i)_{1\leq i\leq \tilde{N}}$ are different since these serve two separate purposes. They are generated in the same way but independently of each other. After fitting (1), estimates $(\hat{b}_{0_j}, \hat{b}_j)$ are returned and $\hat{b}_{0_j} + \hat{b}_j h(y)$ is taken as jth summary statistic, $j = 1, ..., \dim(\theta)$. We can then take $S_j(y^{\text{obs}}) = \hat{b}_{0_j} + \hat{b}_j h(y^{\text{obs}})$ as jth component of $S(y^{\text{obs}})$, and similarly take $S_j(y^\star) = \hat{b}_{0_j} + \hat{b}_j h(y^\star)$. The number of summaries is therefore equal to the size of θ.

This approach is further developed in Jiang et al. (2017) where a MLP deep neural network regression model is employed, and replaces the linear regression model in (1). Hence, Jiang et al. (2017) has the following regression model

$$\theta^i = E(\theta|y^i) + \xi^i = f_\beta(y^i) + \xi^i$$

where f_β is the MLP parametrized by the weights β. Jiang et al. (2017) estimate β from

$$\min_\beta \frac{1}{N}\sum_{i=1}^{N} \|f_\beta(y^i) - \theta^i\|_2^2, \tag{2}$$

where $(\theta^i, y^i)_{1\leq i\leq N}$ are the parameter-data pairs that the network f_β is fitted to.

The deep neuronal network with multiple hidden layers considered in Jiang et al. (2017) offers stronger representational power to approximate $E(\theta|y)$ (and hence learn an informative summary statistic), compared to using linear regression, if the posterior mean is a highly non-linear function of y. Moreover, experiments in Jiang et al. (2017) show that indeed their MLP outperforms the linear regression approach in Fearnhead & Prangle (2012) (at least for their considered experiments), although at the price of a much larger computational effort. For this reason in our experiments we compare ABC coupled with PENs with the ABC MLP from Jiang et al. (2017).

In Creel (2017) a deep neural network regression model is used. He also introduces a pre-processing step such that instead of feeding the network with the data set y^{obs}, the network is fed with a set of statistics of the data s^{obs}. This means that, unlike in Jiang et al. (2017), in Creel (2017) the statistician must already know "some kind" of initial summary statistics, used as input, and then the network returns another set of summary statistics as output, and the latter are used for ABC inference. Our PENs do not require any initial specification of summary statistics.

3. Partially exchangeable networks

Even though the likelihood function is intractable in the likelihood-free setting, we may still have insights into properties of the data generating process. To that end, given our data set $y \in \mathcal{Y}^M$ with M units, we will exploit some of the invariance properties of its prior predictive distribution $p(y) = \int_\theta p(y|\theta)p(\theta)d\theta$. As discussed in Section 2, the regression approach to ABC (Fearnhead & Prangle, 2012) involves to learn the regression function $y \mapsto E(\theta|y)$, where $E(\theta|y)$ is the posterior mean. Our goal in this section is to leverage the invariances of the Bayesian model $p(y)$ to design deep neural architectures that are fit for this purpose.

3.1. Exchangeability and partial exchangeability

The simplest form of model invariance is *exchangeability*. A model $p(y)$ is said to be exchangeable if, for all permutations σ in the symmetric group S_M, $p(y) = p(y_{\sigma(1)}, ..., y_{\sigma(M)})$. For example, if the observations are independent and identically distributed (i.i.d.) given the parameter, then $p(y)$ is exchangeable. A famous theorem of de Finetti (1929), which was subsequently generalized in various ways (see e.g. the review of Diaconis, 1988), remarkably shows that such conditionally i.i.d. models are essentially the only exchangeable models.

If the model is exchangeable, it is clear that the function $y \mapsto E(\theta|y)$ is permutation invariant. It is therefore desirable that a neural network used to approximate this function should also be permutation invariant. The design of permutation invariant neural architectures has been the subject of numerous works, dating at least back to Minsky & Papert (1988, Chap. 2) and Shawe-Taylor (1989). A renewed interest in such architectures came about recently, notably through the works of Ravanbakhsh et al. (2017), Zaheer et al. (2017), and Murphy et al. (2019)—a detailed overview of this rich line of work can be found in Bloem-Reddy & Teh (2019). Most relevant to our work is the DeepSets architecture of Zaheer et al. (2017) that we generalize to partial exchangeability, and the approach of Chan et al. (2018), who used permutation invariant networks for ABC.

However, the models considered in ABC are arising from intractable-likelihoods scenarios, which certainly are not limited to exchangeable data, quite the opposite, e.g. stochastic differential equations (Picchini, 2014), state-space models and beyond (Jasra, 2015). To tackle this limitation, we ask: *could we use a weaker notion of invariance to propose deep architectures suitable for such models?* In this paper, we answer this question for a specific class of non-i.i.d. models: Markov chains. To this end, we make use of the notion of *partial exchangeability* studied by Diaconis & Freedman (1980). This property can be seen as a weakened version of exchangeability where $p(y)$ is only invariant to a subset of the symmetric group called *block-switch transformations*. Informally, for $d \in \mathbb{N}$, a d-block-switch transformation interchanges two given disjoint blocks of $y \in \mathcal{Y}^M$ when these two blocks start with the same d symbols and end with the same d symbols.

Definition 1 (Block-switch transformation). *For increasing indices $b = (i, j, k, l) \in \{0, \dots, M\}^4$ such that $j - i \geq d$ and $l - k \geq d$, the **d-block-switch transformation** $T_b^{(d)}$ is defined as follows: if $y_{i:(i+d)} = y_{k:(k+d)}$ and $y_{(j-d):j} = y_{(l-d):l}$ then*

$$y = y_{1:i-1}\, y_{i:j}\, y_{(j+1):(k-1)}\, y_{k:l}\, y_{(l+1):M} \tag{3}$$

$$T_b^{(d)}(y) = y_{1:i-1}\, y_{k:l}\, y_{(j+1):(k-1)}\, y_{i:j}\, y_{(l+1):M}. \tag{4}$$

If $y_{i:(i+d)} \neq y_{k:(k+d)}$ or $y_{(j-d):j} \neq y_{(l-d):l}$ then the block-switch transformation leaves y unchanged: $T_b^{(d)}(y) = y$.

Definition 2 (Partial exchangeability). *Let A be a metric space. A function $F : \mathcal{Y}^M \to A$ is said to be d-**block-switch invariant** if $F(y) = F(T_b^{(d)}(y))$ for all $y \in \mathcal{Y}$ and for all d-block-switch transformations $T_b^{(d)}$. Similarly, a model $p(y)$ is d-**partially exchangeable** if for all d-block-switch transformations $T_b^{(d)}$ we have $p(y) = p(T_b^{(d)}(y))$.*

Note that 0-partial exchangeability reduces to exchangeability and that all permutations are 0-block-switch transformations.

It is rather easy to see that, if $p(y|\theta)$ is a Markov chain of order d, then $p(y)$ is partially exchangeable (and therefore $y \mapsto E(\theta|y)$ is d-block-switch invariant). In the limit of infinite data sets, Diaconis & Freedman (1980) showed that the converse was also true: any partially exchangeable distribution is conditionally Markovian. This result, which is an analogue of de Finetti's theorem for Markov chains, justifies that *partial exchangeability is the right symmetry to invoke when dealing with Markov models*.

3.2. From model invariance to network architecture

When dealing with Markovian data, we therefore wish to model a regression function $y \mapsto E(\theta|y)$ that is d-block-switch invariant. Next theorem gives a general functional representation of such functions, in the case where $\mathcal{Y}$ is countable.

Theorem 1. *Let $F : \mathcal{Y}^M \to A$ be d-block-switch invariant. If $\mathcal{Y}$ is countable, then there exist two functions $\phi : \mathcal{Y}^{d+1} \to \mathbb{R}$ and $\rho : \mathcal{Y}^d \times \mathbb{R} \to A$ such that*

$$\forall y \in \mathcal{Y}^M,\ F(y) = \rho\left(y_{1:d}, \sum_{i=1}^{M-d} \phi\left(y_{i:(i+d)}\right)\right). \qquad (5)$$

Proof. Let $\sim$ be the equivalence relation over $\mathcal{Y}^M$ defined by

$$x \sim y \iff \exists b_1, \dots, b_k,\ \ y = T_{b_1}^{(d)} \circ \cdots \circ T_{b_k}^{(d)}(x).$$

Let $\mathrm{cl} : \mathcal{Y}^M \to \mathcal{Y}^M/\!\sim$ be the projection over the quotient set. According to the properties of the quotient set, since F is d-block-switch invariant, there exists a unique function $g : \mathcal{Y}^M/\!\sim \to A$ such that $F = g \circ \mathrm{cl}$.

Since $\mathcal{Y}$ is countable, $\mathcal{Y}^{d+1}$ is also countable and there exists an injective function $c : \mathcal{Y}^{d+1} \to \mathbb{N}$. Consider then the function

$$\nu : y \mapsto \left(y_{1:d}, \sum_{i=1}^{M-d} 2^{-c(y_{i:(i+d)})}\right),$$

which is clearly d-block-switch invariant. There exists a unique function $h : \mathcal{Y}^M/\!\sim \to \nu(\mathcal{Y}^M)$ such that $\nu = h \circ \mathrm{cl}$.

We will now show that h is a bijection. By construction, h is clearly surjective. Let us now prove its injectivity. We thus have to show that, for all $x, y \in \mathcal{Y}^M$, $\nu(x) = \nu(y)$ implies $x \sim y$. Let $x, y \in \mathcal{Y}^M$ such that $\nu(x) = \nu(y)$. We have therefore $x_{1:d} = y_{1:d}$ and

$$\sum_{i=1}^{M-d} 2^{-c(x_{i:(i+d)})} = \sum_{i=1}^{M-d} 2^{-c(y_{i:(i+d)})}.$$

The uniqueness of finite binary representations then implies that $\{x_{i:(i+d)}\}_{i \leq M-d} = \{y_{i:(i+d)}\}_{i \leq M-d}$. According to Diaconis & Freedman (1980, Proposition 27), those two conditions imply that $x \sim y$, which shows that h is indeed injective.

Since h is a bijection, $\nu = h \circ \mathrm{cl}$ implies that $\mathrm{cl} = h^{-1} \circ \nu$ which leads to $F = g \circ h^{-1} \circ \nu$. Finally, expanding this gives

$$\forall y \in \mathcal{Y}^M,\ F(y) = g \circ h^{-1}\left(y_{1:d}, \sum_{i=1}^{M-d} 2^{-c(y_{i:(i+d)})}\right),$$

which is the desired form with $\phi(y) = 2^{-c(y)}$ and $\rho = g \circ h^{-1}$. □

When $d = 0$, the representation reduces to

$$F(y) = \rho\left(\sum_{i=1}^{M} \phi\left(y_i\right)\right), \qquad (6)$$

and we exactly recover Theorem 2 from Zaheer et al. (2017)—which also assumes countability of $\mathcal{Y}$—and the DeepSets representation. While an extension of our theorem to the uncountable case is not straightforward, we conjecture that a similar result holds even with uncountable $\mathcal{Y}$. A possible way to approach this conjecture is to study the very recent and fairly general result of Bloem-Reddy & Teh (2019). We note that the experiments on an autoregressive time series model in Section 4.3, which is a Markovian process, support this conjecture.

Partially exchangeable networks The result in Theorem 1 suggests how to build d-block-switch invariant neural networks: we replace the functions ρ and ϕ in Equation (5) by feed forward neural networks and denote this construction a d-partially exchangeable network (PEN-d or PEN of order d). In this construction, we will call ϕ the *inner network*, which maps a d-length subsequence $y_{i:i+d}$ into some representation $\phi(y_{i:i+d})$, and ρ is the *outer network* that maps the first d symbols of the input, and the sum of the representations of all d-length subsequences of the input, to the output. We note that DeepSets networks are a special case of the PENs that corresponds to PEN-0.

3.3. Using partially exchangeable networks for learning summary statistics for ABC

While PENs can by used for any exchangeable data, in this paper we use it for learning summary statistics in ABC. In particular, we propose the following regression model for learning the posterior mean

$$\theta^i = E(\theta|y^i) + \xi^i = \rho_{\beta_\rho}\left(y^i_{1:d}, \sum_{l=1}^{M-d} \phi_{\beta_\phi}(y^i_{l:l+d})\right) + \xi^i.$$

Here β_ϕ are the weights for the inner network, and β_ρ are the weights for the outer network that maps its arguments into the posterior mean of the unknown parameters, which is the ABC summary we seek. When using PENs to learn the summary statistics we obtain the weights for the networks using the same criterion as in Equation (2), except that instead of using the MLP network we use a PEN network for the underlying regression problem.

When targeting static models we employ a PEN-0, i.e. a DeepSets network, since a static model can be viewed as a zero-order Markov model. For time series models we use a PEN-d, where $d > 0$ is the order of the assumed data generating Markov process.

4. Experiments

We present four experiments: two static models (g-and-k and α-stable distributions), and two time series models (autoregressive and moving average models). Full specification of the experimental settings is provided as supplementary material. The code was written in Julia 1.0.0 (Bezanson et al., 2017) and the framework Knet (Yuret, 2016) was used to build the deep learning models. The code can be found at https://github.com/SamuelWiqvist/PENs-and-ABC. All experiments are simulation studies and the data used can be generated from the provided code. We compare approximate posteriors to the true posteriors using the Wasserstein distance, which we compute via the POT package (Flamary & Courty, 2017). This distance can be sensitive to the number of posterior samples used, however, we observed that our results are fairly robust to variations in the number of samples. In all experiments we used 100 posterior samples to estimate the Wasserstein distance, except for the AR2 model where we used 500 samples. We also employ two different MLP networks: "MLP small", where we use approximately the same number of weights as for the PEN-d network; and "MLP large", which has a larger number of weights than PEN-d.

4.1. g-and-k distribution

The g-and-k distribution is defined by its quantile function via four parameters, and not by its probability density function since the latter is unavailable in closed form. This means that the likelihood function is "intractable" and as such exact inference is not possible. However, it is very simple to simulate draws from said distribution (see the supplementary material), which means that g-and-k models are often used to test ABC algorithms (Prangle, 2017).

The unknown parameters are $\theta = [A, B, g, k]$ (for full specification of the g-and-k distribution, see the supplementary material). The prior distributions are set to $p(A) \sim \Gamma(2, 1)$, $p(B) \sim \Gamma(2, 1)$, $p(g) \sim \Gamma(2, 0.5)$, and $p(k) \sim \Gamma(2, 1)$ ($\Gamma(\alpha, \beta)$ is the Gamma distribution with shape parameter α and rate parameter β). We perform a simulation study with ground-truth parameters $A = 3$, $B = 1$, $g = 2$, $k = 0.5$ (same ground-truth parameter values as in Allingham et al., 2009, Picchini & Anderson, 2017, Fearnhead & Prangle, 2012). Our data set comprises $M = 1,000$ realizations from a g-and-k distribution.

We compare five different methods of constructing the summary statistics for ABC: (i) the handpicked summary statistics in Picchini & Anderson (2017), i.e. $S(y) = [P_{20}, P_{40}, P_{60}, P_{80}, \text{skew}(y)]$ (P_i is the ith percentile and skew(y) is the skewness); (ii) "MLP small"; (iii) "MLP large"; (iv) a MLP network with a preprocessing step, denoted "MLP pre", where we feed the network with the empirical distribution function of the data instead of feeding it with the actual data; and (v) PEN-0 (DeepSets) since the data is i.i.d. the order of the Markov model is 0).

The probability density function for the g-and-k distribution can be approximated via finite differences, as implemented in the `gk` R package (Prangle, 2017). This allow us to sample from an almost exact posterior distribution using standard Markov chain Monte Carlo (MCMC). We evaluate the inference produced using summaries constructed from the five methods (i–v) by comparing the resulting ABC posteriors to the "almost exact" posterior (computed using MCMC). ABC inferences are repeated over 100 independent data sets, and for a different number of training data observations for DNN models. The results are presented in Figure 1 and we can conclude that PEN-0 generates the best results. Furthermore, PEN-0 is also more data efficient since it performs considerably better than other methods with limited number of training observations. It seems in fact that PEN-0 requires 10 times less training data than "MLP pre" to achieve the same inference accuracy. However all methods performed poorly when too few training observations are used. The results also show that when MLP is fed with the observations it generates poor results, but if we instead use "MLP pre" and send in the empirical distribution function, in the spirit of Creel (2017), we obtain considerably better results.

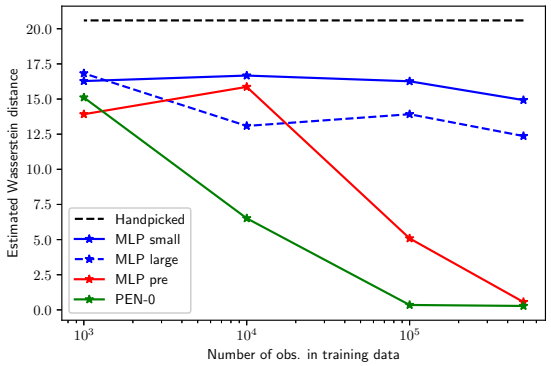


Figure 1. Results for g-and-k distribution: The estimated Wasserstein distances (mean over 100 repetitions) when comparing the MCMC posterior with ABC posteriors.

4.2. α-stable distribution

The α-stable is a heavy-tailed distribution defined by its characteristic function (see supplementary material). Its probability density function is intractable and inference is therefore challenging. Bayesian methods for the parameters can be found in e.g. Peters et al. (2012) and Ong et al. (2018). Unknown parameters are $\theta = [\alpha, \beta, \gamma, \delta]$. We follow Ong et al. (2018) and transform the parameters:

$$\tilde{\alpha} = \log \frac{\alpha - 1.1}{2 - \alpha}, \ \tilde{\beta} = \log \frac{\beta + 1}{1 - \beta}, \ \tilde{\gamma} = \log \gamma, \text{and } \tilde{\delta} = \delta.$$

This constraints the original parameters to $\alpha \in [1.1, 2]$, $\beta \in [-1, 1]$, and $\gamma > 0$. Independent Gaussian priors and ground-truth parameters are as in Ong et al. (2018): $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}, \tilde{\delta} \sim N(0, 1)$; ground-truth values for the untransformed parameters are: $\alpha = 1.5$, $\beta = 0.5$, $\gamma = 1$, and $\delta = 0$. Observations consist of $M = 1,000$ samples.

We compare methods for computing summary statistics as we did in Section 4.1 for the g-and-k distribution. However, since here the true posterior distribution is unavailable, we evaluate the different methods by comparing the root-mean square error (RMSE) between ground-truth parameter values and the ABC posterior means, see Table 1. From Table 1 we conclude that PEN-0 performs best in terms of RMSE. Similarly to the g-and-k example we also see that "MLP pre" (see Section 4.1 for details) performs considerably better than MLP. We now look at the resulting posteriors. In Figure 2 five posteriors from five independent experiments are presented (here we have used $5 \cdot 10^5$ training data observations). Inference results when using handpicked summary statistics are poor and for $\tilde{\gamma}$ the posterior resembles the prior. Posterior inference is worst for "MLP large". Results for "MLP pre" and PEN-0 are similar, at least in the case depicted in Figure 2 where we use $5 \cdot 10^5$ training data observations. However, in terms of RMSE, PEN-0 returns the best results when we reduce the number of training data observations.

Table 1. Results for α-stable distribution. Root-mean square error (RMSE) when comparing posterior means to the ground-truth parameters (over 25 repetitions), for different methods of computing the summary statistics, and different number of training observations (between brackets).

	HANDPICKED	MLP (SMALL)	MLP (LARGE)	MLP PRE	PEN-0
RMSE ($5 \cdot 10^5$)	0.64	0.18	0.15	0.07	0.05
RMSE (10^5)	0.64	0.19	0.17	0.07	0.06
RMSE (10^4)	0.64	0.21	0.37	0.07	0.06
RMSE (10^3)	0.64	0.72	0.62	0.40	0.07

4.3. Autoregressive time series model

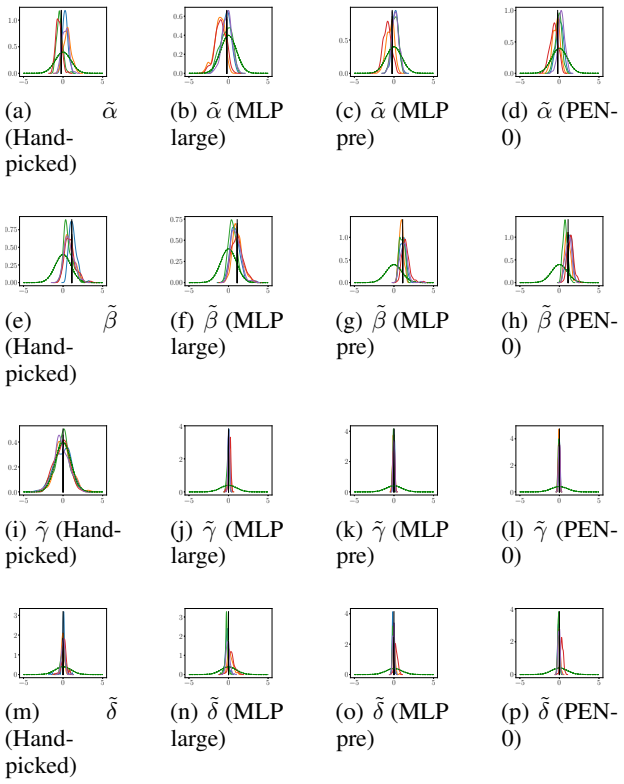
An autoregressive time series model of order two (AR(2)) follows:

$$y_l = \theta_1 y_{l-1} + \theta_2 y_{l-2} + \xi_l, \qquad \xi_l \sim N(0, 1).$$

The AR(2) model is identifiable if the following are fulfilled: $\theta_2 < 1 + \theta_1, \theta_2 < 1 - \theta_1, \theta_2 > -1$ (Fuller, 1976). We let the resulting triangle define the uniform prior for the model. The ground-truth parameters for this simulation study are set to $\theta = [0.2, -0.13]$, and the data size is $M = 100$. AR(2) is a Markov model, hence and the requirement for PEN-d with $d > 0$ is fulfilled.

We compare five methods for computing the summaries: (i) handpicked summary statistics, i.e. $S(y) = [\gamma(y, 1), \gamma(y, 2), \gamma(y, 3), \gamma(y, 4), \gamma(y, 5)]$ ($\gamma(y, i)$ is autocovariance at lag i), which are reasonable summary statistics since autocovariances are normally employed in parameter estimation for autoregressive models, for instance when using the YuleWalker equations; (ii) "MLP small" network; (iii) "MLP large"; (iv) PEN-0 (DeepSets); and (v) PEN-2. Since AR(2) is a time series model it makes sense to use PEN-2, and PEN-0 results are reported only in the interest of comparison. Here we do not consider the "MLP pre" method used in Section 4.1 and 4.2, since the empirical distribution function does not have any reasonable meaning for time series data. The likelihood function for AR(2) is known and we can therefore sample from the true posterior using MCMC.

Results are in Figure 3. PEN-2 outperforms MLP, for example we can see that the precision achieved when PEN-2 is trained on 10^3 training observations can be achieved by MLP when trained on 10^5 observations, implying an improvement of a 10^2 factor. Approximate and exact posteriors are in Figure 4 and we conclude that posteriors for both



(a) $\tilde{\alpha}$ (Hand-picked) (b) $\tilde{\alpha}$ (MLP large) (c) $\tilde{\alpha}$ (MLP pre) (d) $\tilde{\alpha}$ (PEN-0)

(e) $\tilde{\beta}$ (Hand-picked) (f) $\tilde{\beta}$ (MLP large) (g) $\tilde{\beta}$ (MLP pre) (h) $\tilde{\beta}$ (PEN-0)

(i) $\tilde{\gamma}$ (Hand-picked) (j) $\tilde{\gamma}$ (MLP large) (k) $\tilde{\gamma}$ (MLP pre) (l) $\tilde{\gamma}$ (PEN-0)

(m) $\tilde{\delta}$ (Hand-picked) (n) $\tilde{\delta}$ (MLP large) (o) $\tilde{\delta}$ (MLP pre) (p) $\tilde{\delta}$ (PEN-0)

Figure 2. Results for α-stable distribution: Approximate marginal ABC posteriors. Results obtained using $5 \cdot 10^5$ training data observations. The green dashed line is the prior distribution. The colored lines show posteriors from 5 independent experiments. These posteriors are not cherry-picked.

MLP and PEN-2 are similar to the true posterior when many training observations are used. However, the approximate posterior for MLP degrades significantly when the number of training observations is reduced and is very uninformative with 10^3 and even with 10^4 observations, while for PEN-2 the quality of the approximate posterior distribution is only marginally reduced.

4.4. Moving average time series with observational noise model

We consider a partially observed time series, with latent dynamics given by a moving average MA(2) model and observations perturbed with Gaussian noise:

$$\begin{cases} y_l = x_l + \xi_l^y, & \xi_l^y \sim N(0, \sigma_\epsilon = 0.3), \\ x_l = \xi_l + \theta_1 \xi_{l-1}^x + \theta_2 \xi_{l-2}^x, & \xi_l^x \sim N(0,1), \end{cases}$$

where the ξ_l^x and ξ_l^y are all independent. An MA(2) process without observational noise is identifiable if $\theta_1 \in [-2, 2]$, $\theta_2 \in [-1, 1]$, and $\theta_2 \pm \theta_1 \geq -1$. Same as in Jiang et al. (2017), we define a uniform prior over this triangle. We use

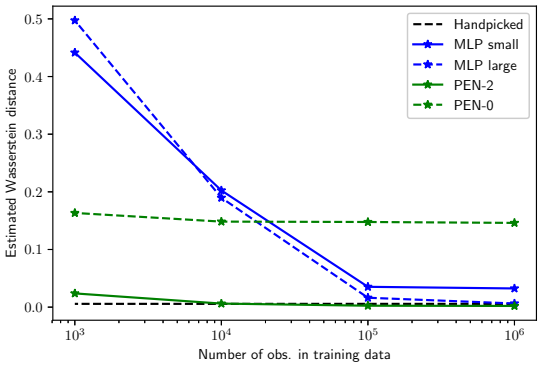


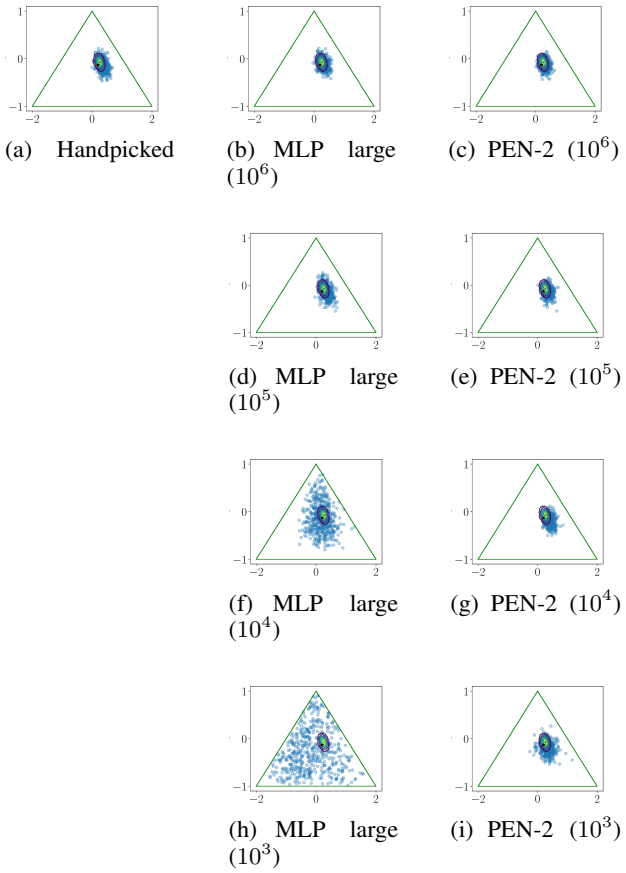
Figure 3. Results for AR(2) model: Estimated Wasserstein distances (mean over 100 data sets) when comparing the true posterior with ABC posteriors, for varying sizes of training data when using DNN models.

the same setting as in Jiang et al. (2017) and set the ground-truth parameters for the simulation study to $\theta = [0.6, 0.2]$. We only observe $\{y_l\}$ and the number of observations is $M = 100$.

The latent dynamics are not Markovian, hence the Markov property required for PEN of order larger than 0 is not fulfilled, however, the quasi-Markov structure of the data might still allow us to successfully use PEN-d with an order d larger than 0. An additional complication is given by the observational noise ξ_l^y, further perturbing the dynamics. Once more, we compare five methods for computing the summary statistics: (i) handpicked summaries $S(y) = [\gamma(y, 1), \gamma(y, 2)]$, i.e. we follow Jiang et al. (2017); (ii) "MLP small"; (iii) "MLP large"; (iv) PEN-0 (DeepSets); and (v) PEN-10. Same as for the AR(2) example, here PEN-0 results are reported only in the interest of a comparison with PEN-10, as for a time-series model it is expected from PEN-0 to be suboptimal. Also in this case the likelihood function is available, and we can compute the true posterior distribution. Once more, we compare the approximate posteriors to the true posterior over 100 different data sets, see Figure 5. We conclude that PEN-10 performs slightly better than MLP when the training data set is large, and that PEN-10 outperforms MLP when we restrict the size of the training data. Once more, we notice that PEN-10 implies a factor ≥ 10 in terms of savings on the size of the training data.

5. Discussion

Simulation experiments show that our partially exchangeable networks (PENs) achieve competitive results in learning summary statistics for use in ABC algorithms, outper-



(a) Handpicked (b) MLP large (10^6) (c) PEN-2 (10^6)

(d) MLP large (10^5) (e) PEN-2 (10^5)

(f) MLP large (10^4) (g) PEN-2 (10^4)

(h) MLP large (10^3) (i) PEN-2 (10^3)

Figure 4. Results for AR(2) model. The green line indicates the prior distribution, the contour plot is from the exact posterior and the blue dots are 100 samples from the several ABC posteriors. The number in parenthesis indicates number of observations in the training data set. These posteriors are not cherry-picked.

forming the other deep learning methods that we have considered. Moreover, PENs require much smaller training data to achieve the same inference accuracy of competitors: in our experiments a reduction factor of order 10 to 10^2 was observed.

As mentioned in Section 2, in this work we were not focused on the specific ABC algorithm used for sampling, but only on learning summary statistics for ABC. However, in future work we plan to use our approach for constructing summary statistics alongside more sophisticated variants of ABC methods, such as those which combine ABC with Markov chain Monte Carlo (Sisson & Fan, 2011) or sequential techniques (Beaumont et al., 2009).

Murphy et al. (2019) recently shed light on some limitations of the DeepSets architecture, and proposed to improve it by replacing the sum fed to the outer network by another pooling techinque called *Janossy pooling*. Since the drawbacks they inspect are also likely to affect our architectures, extending Janossy pooling to the PEN framework might constitute a valuable improvement.

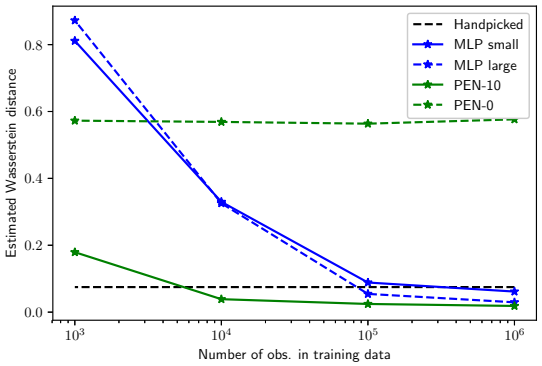


Figure 5. Results for MA(2) model: Estimated Wasserstein distances (mean over 100 data sets) when comparing the true posterior with ABC posteriors.

Our experiments show that the performance of the MLP networks using different choices for the number of weights is quite similar, and that PEN outperforms MLP even when MLP has access to a larger number of weights compared to PEN. The main insight is that PENs by design incorporate the (partial) exchangeability property of the data, whereas the MLPs have to learn this property. Exchangeability and partial exchangeability can in principle be expressed in an MLP, but for small data sets these properties will be difficult to learn, and we expect that the model will overfit to the training data. One approach to alleviate this problem for MLPs is to perform data augmentation. However, it is not straightforward to perform data augmentation for continuous Markovian data, unless we have access to the underlying data generating process. In ABC the assumption is that we do have access to this process, but data generation may be computational expensive, and in a more general application we may not have access to the process.

Although we have applied the PEN architecture to the problem of learning summary statistics for ABC, notice that PEN is a general architecture and could be used for other applications. One example would be time series classification.

The main limitation for PEN is that it is designed for Markovian data or, when considering the special case of DeepSets (i.e. PEN-0), for exchangeable data. However, in the MA(2) example we achieve good inference results even though the MA(2) model is itself non-Markovian *and* observations are perturbed with measurement noise.

Acknowledgements

Research was partially supported by the Swedish Research Council (VR grant 2013-05167). We would also like to thank Joachim Hein and colleagues at LUNARC, Lund University, for helping out on setting up the GPU environment used for the simulations.

References

Allingham, D., King, R., and Mengersen, K. L. Bayesian estimation of quantile distributions. *Statistics and Computing*, 19(2):189–201, 2009.

Beaumont, M. A., Zhang, W., and Balding, D. J. Approximate Bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. Adaptive approximate Bayesian computation. *Biometrika*, 96(4):983–990, 2009.

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

Bloem-Reddy, B. and Teh, Y. W. Probabilistic symmetry and invariant neural networks. *arXiv:1901.06082*, 2019.

Blum, M. G., Nunes, M. A., Prangle, D., Sisson, S. A., et al. A comparative review of dimension reduction methods in approximate Bayesian computation. *Statistical Science*, 28(2):189–208, 2013.

Chan, J., Perrone, V., Spence, J., Jenkins, P., Mathieson, S., and Song, Y. A likelihood-free inference framework for population genetic data using exchangeable neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 8603–8614. Curran Associates, Inc., 2018.

Cornuet, J.-M., Santos, F., Beaumont, M. A., Robert, C. P., Marin, J.-M., Balding, D. J., Guillemaud, T., and Estoup, A. Inferring population history with DIY ABC: a user-friendly approach to approximate Bayesian computation. *Bioinformatics*, 24(23):2713–2719, 2008.

Creel, M. Neural nets for indirect inference. *Econometrics and Statistics*, 2:36–49, 2017.

de Finetti, B. Funzione caratteristica di un fenomeno aleatorio. In *Atti del Congresso Internazionale dei Matematici: Bologna dal 3 al 10 di settembre 1928*, pp. 179–190, 1929.

Diaconis, P. Recent progress on de Finetti's notions of exchangeability. *Bayesian statistics*, 3:111–125, 1988.

Diaconis, P. and Freedman, D. de Finetti's theorem for Markov chains. *The Annals of Probability*, pp. 115–130, 1980.

Fearnhead, P. and Prangle, D. Constructing summary statistics for approximate bayesian computation: semi-automatic approximate Bayesian computation. *Journal of the Royal Statistical Society: Series B*, 74(3):419–474, 2012.

Flamary, R. and Courty, N. POT Python optimal transport library, 2017. URL https://github.com/rflamary/POT.

Fuller, W. A. *Introduction to time series analysis*. New York: John Wiley & Sons, 1976.

Jasra, A. Approximate Bayesian computation for a class of time series models. *International Statistical Review*, 83 (3):405–435, 2015.

Jiang, B., Wu, T.-y., Zheng, C., and Wong, W. H. Learning summary statistic for approximate Bayesian computation via deep neural network. *Statistica Sinica*, pp. 1595–1618, 2017.

Marin, J.-M., Pudlo, P., Robert, C. P., and Ryder, R. J. Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

Minsky, M. and Papert, S. *Perceptrons (expanded edition) MIT Press*. 1988.

Murphy, R. L., Srinivasan, B., Rao, V., and Ribeiro, B. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. In *International Conference on Learning Representations*, 2019.

Ong, V. M., Nott, D. J., Tran, M.-N., Sisson, S. A., and Drovandi, C. C. Variational bayes with synthetic likelihood. *Statistics and Computing*, 28(4):971–988, 2018.

Peters, G. W., Sisson, S. A., and Fan, Y. Likelihood-free bayesian inference for α-stable models. *Computational Statistics & Data Analysis*, 56(11):3743–3756, 2012.

Picchini, U. Inference for SDE models via approximate Bayesian computation. *Journal of Computational and Graphical Statistics*, 23(4):1080–1100, 2014.

Picchini, U. and Anderson, R. Approximate maximum likelihood estimation using data-cloning ABC. *Computational Statistics & Data Analysis*, 105:166–183, 2017.

Prangle, D. Summary statistics in approximate Bayesian computation. *arXiv:1512.05633*, 2015.

Prangle, D. gk: An R package for the g-and-k and generalised g-and-h distributions. *arXiv:1706.06889*, 2017.

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., and Feldman, M. W. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular biology and evolution*, 16(12):1791–1798, 1999.

Ravanbakhsh, S., Schneider, J., and Poczos, B. Deep learning with sets and point clouds. *International Conference on Learning Representations (ICLR) - workshop track*, 2017.

Shawe-Taylor, J. Building symmetries into feedforward networks. In *Artificial Neural Networks, 1989., First IEE International Conference on (Conf. Publ. No. 313)*, pp. 158–162. IET, 1989.

Sisson, S. A. and Fan, Y. *Handbook of Markov Chain Monte Carlo*, chapter Likelihood-free Markov chain Monte Carlo. Chapman and Hall, 2011.

Sisson, S. A., Fan, Y., and Beaumont, M. *Handbook of Approximate Bayesian Computation*. Chapman and Hall/CRC, 2018.

Yuret, D. Knet: beginning deep learning with 100 lines of julia. In *Machine Learning Systems Workshop at NIPS*, volume 2016, pp. 5, 2016.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In *Advances in Neural Information Processing Systems*, pp. 3391–3401, 2017.